Transformation of Graphical Models to Support Knowledge Transfer

Vom Fachbereich Elektrotechnik und Informatik der Universität Siegen

zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertation

von

Dipl.-Inform. Alexander Holland

1. Gutachter:	Prof. Dr.-Ing. Madjid Fathi
2. Gutachter:	Prof. Dr. rer. nat. Rainer Brück
Vorsitzender:	Prof. Dr. rer. nat. Udo Kelter

Tag der mündlichen Prüfung: 25. November 2008

gedruckt auf alterungsbeständigem holz- und säurefreiem Papier

Contents

Notation

Lists of Symbols

Spaces

$\mathbb{B}$	Space of binary numbers
$\mathbb{R}$	Space of real numbers
$\mathbb{R}^+$	Space of the positive real numbers, $(0, \infty)$
$\mathbb{R}_0^+$	Space of the positive real numbers including zero, $[0, \infty)$
$\mathbb{Z}$	Space of integer-valued numbers

Relations, Operators, and Functions

$\sim$	Distributed according to
$\preceq$	Preference relation
$\Delta(\cdot)$	Decision function
$\Delta^*(\cdot)$	Optimal decision function
$argmax_D \rho$	An action from D maximizing ρ
$Bel(X)$	Posterior distribution over X
$\mathrm{erf}(\cdot)$	Error function, also called Gauss error function
$\mathrm{erf}^{-1}(\cdot)$	Inverse error function
$E(\cdot)$	Expectation
$EU(\cdot)$	Expected utility
$\inf(\cdot)$	Infimum
$\mathcal{I}$	Indifference relation
$\mathcal{J}$	Incomparability relation
$\mathcal{L}$	Set of labels
$\min(\cdot)$	Minimum

$\max(\cdot)$	Maximum
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with expectation μ and standard deviation σ
$\mathcal{P}$	Preference relation
$P(a)$	Probability of event a
$P(a\|b)$	Probability of event a given b
$P(A)$	Probability distribution for variable A
$P(A\|B)$	Probability distributions for variable A given the states of variable B
$P(A, B)$	Joint probability distribution for variables A and B
$pa(A)$	The parent set for variable A
ξ	Probability measure
$\mathcal{S}$	Set of possible world states
$\mathcal{T}$	σ-field
$\mathcal{U}(\cdot)$	Utility function
$\mathrm{Var}(\cdot)$	Variance
$\mathcal{W}$	Linear probability information

List of Acronyms

AI	Artificial Intelligence
AIC	Akaike Information Criterion
BC	Belt Conveyor
BN	Bayesian Network
CBN	Causal Bayesian Network
cdf	Cumulative distribution function
CF	Collaborative Filtering
CI	Computational Intelligence
CM	Condition Monitoring
CPT	Conditional Probability Table

DAG	Direct Acyclic Graph
DBN	Dynamic Bayesian Network
DDN	Dynamic Decision Network
DN	Decision Network
DS	Dempster Shafer
DT	Decision Tree
EA	Evolutionary Algorithm
EDM	Electrical Discharge Machining
EM	Expected Maximization
ETA	Event Tree Analysis
EU	Expected Utility
EUT	Expected Utility Theory
FL	Fuzzy Logic
FMEA	Failure Mode and Effects Analysis
FTA	Fault Tree Analysis
GC	Granular Computing
GHC	Greedy Hill Climbing
ID	Influence Diagram
ITSM	IT Service Management
KEBN	Knowledge Engineering with Bayesian Networks
KBS	Knowledge Based System
KD	Knowledge Desk
KL	Kullback Leibler divergence
KM	Knowledge Management
LAGD	Look ahead in good directions
LS	Log Score
MAP	Maximum-a-posterior
MCDM	Multiple criteria decision making
MDL	Minimum Description Length

MEU	Maximal Expected Utility
ML	Machine Learning
NN	Neural Network
pdf	Probability density function
OOBN	Object-Oriented Bayesian Network
PDM	Product Data Management
PLM	Product Lifecycle Management
PSS	Product Service System
RCA	Root Cause Analysis
RM	Risk Management
RS	Rotary Spindle
rv	Random variable
SA	Simulated Annealing
TE	Teamcenter Engineering
WEDM	Wire Electrical Discharge Machining
WEDG	Wire Electrical Discharge Grinding
WEKA	Waikato Environment for Knowledge Analysis

Acknowledgements

This thesis is a product of my work in the department of Electrical Engineering and Computer Science at the Institute of Knowledge Based Systems and Knowldge Management in Siegen, Germany. At first I would like to thank Madjid Fathi for letting me be part of his group and for providing a great environment for research.

I wish to thank my advisors Madjid Fathi and Rainer Brück for advice and assistance. They guided me in this endeavor with good ideas, insightful comments, friendly support and just the right mixture of affirmation and criticism.

Many thanks go to Manuel Neubach from the chair IT in Mechanical Engineering at the University of Bochum. We have discussed for hours together and exchanged our ideas in the field of applications based on product lifecycle management and condition monitoring.

Finally, i owe a lot to my colleagues from the Institute of Knowledge Based Systems and Knowldge Management for proofreading parts of this work.

Zusammenfassung

Menschliche Experten verfügen über die Fähigkeit, ihr Entscheidungsverhalten flexibel auf die jeweilige Situation abzustimmen. Diese Fähigkeit zahlt sich insbesondere dann aus, wenn Entscheidungen unter beschränkten Ressourcen wie Zeitrestriktionen getroffen werden müssen. In solchen Situationen ist es besonders vorteilhaft, die Repräsentation des zugrunde liegenden Wissens anpassen und Entscheidungsmodelle auf unterschiedlichen Abstraktionsebenen verwenden zu können. Weiterhin zeichnen sich menschliche Experten durch die Fähigkeit aus, neben unsicheren Informationen auch unscharfe Wahrnehmungen in die Entscheidungsfindung einzubeziehen.
Klassische entscheidungstheoretische Modelle basieren auf dem Konzept der Rationalität, wobei in jeder Situation die nutzenmaximale Entscheidung einer Entscheidungsfunktion zugeordnet wird. Neuere graphbasierte Modelle wie Bayes'sche Netze oder Entscheidungsnetze machen entscheidungstheoretische Methoden unter dem Aspekt der Modellbildung interessant. Als Hauptnachteil lässt sich die Komplexität nennen, wobei Inferenz in Entscheidungsnetzen NP-hart ist. Zielsetzung dieser Dissertation ist die Transformation entscheidungstheoretischer Modelle in Fuzzy-Regelbasen als Zielsprache. Fuzzy-Regelbasen lassen sich effizient auswerten, eignen sich zur Approximation nichtlinearer funktionaler Beziehungen und garantieren die Interpretierbarkeit des resultierenden Handlungsmodells. Die Übersetzung eines Entscheidungsmodells in eine Fuzzy-Regelbasis wird durch einen neuen Transformationsprozess unterstützt. Ein Agent kann zunächst ein Bayes'sches Netz durch Anwendung eines in dieser Arbeit neu vorgestellten parametrisierten Strukturlernalgorithmus generieren lassen. Anschließend lässt sich durch Anwendung von Präferenzlernverfahren und durch Präzisierung der Wahrscheinlichkeitsinformation ein entscheidungstheoretisches Modell erstellen. Ein Transformationsalgorithmus kompiliert daraus eine Regelbasis, wobei ein Approximationsmaß den erwarteten Nutzenverlust als Gütekriterium berechnet. Anhand eines Beispiels zur Zustandsüberwachung einer Rotationsspindel wird die Praxistauglichkeit des Konzeptes gezeigt.

1. Introduction

During recent years, several types of decision support systems have been proposed in literature. Among these systems, decision-theoretic models and rule-based systems have gained considerable attraction. Expert systems, for instance, are often implemented as rule-based systems, since human expert knowledge can generally be expressed quite well in terms of rules. Fuzzy rule-based systems extend classical rule-based systems in a reasonable way, since expert knowledge is usually involved with vagueness and imprecision.
Decision-theoretic models are based directly on concepts from mathematical (statistical) decision theory. During recent years, these concepts have been developed further in artificial intelligence [RN03]. Particularly, the mathematical framework has been extended by graphical models, notably Bayesian networks and Decision networks ([Avo95], [CF01], [BL04], [Jen01]). In fact, the decision-theoretic approach is now recognized as the most important formal foundation of modeling rational behavior. Moreover, decision networks are widely accepted as important tools for both, the design of intelligent agents ([San96],[Wei00]) as well as the realization of (decision-theoretic) expert systems ([FHMV95], [FDS93]).
Our aim in this thesis is to combine the advantages of both approaches. More precisely, we outline a knowledge transformation framework which allows one to transform a decision theoretic model into a fuzzy rule base. Thus, we consider fuzzy rule-based systems as a special type of condensed decision model [DGL97]. The idea is to maintain a decision theoretic model *offline*, which appears reasonable from a knowledge engineering point of view. For reasons of efficiency, rule-based systems derived from that model are then used *online*, e.g. for real-time decision making ([Kor90], [KD01]) like collaborative agent-based knowledge support [FMS05]. This transformation framework support decision making activities in a wide range of domains in which decisions are made. Decision-theoretic models dispose of adequate concepts for the modeling of uncertainty ([Pea88], [Pea00], [Hol04b]) and take the decision maker's *preferences* [FR94a] into account in an explicit way. Worth mentioning in this connection is the *declarative* character of decision-theoretic models: What the approach requires is only a *description* of the problem and the decision maker's preferences. Given this, the optimal decision behavior is already determined by means of an underlying rationality principle [Joy99]. This way, the declarative approach avoids systematic faults typically made by human experts in rule-based modeling [FH97], e.g. caused by an inconsistent handling of uncertainty [Ban98]. The declarative approach also facilitates the adaptation and extension of a model, e.g. the consideration of a new variable. One disadvantage of decision-theoretic methods concerns the issue of complexity: Finding an optimal decision might become very expensive. In fact, inference in Bayesian networks is known to

be NP-hard [PP02]. Therefore, decision networks are not always suitable for time-critical applications. Rule-based systems are much more efficient in this respect. This is mainly due to the fact that decision knowledge is represented in a very compressed form. For example, so called *condition-action* rules of the form 'IF *condition* THEN *action*' define a kind of action model that assigns an action (a decision) to each situation in an immediate way. As opposed to decision-theoretic models, the situation itself is actually not analyzed before a decision is made. Particularly, aspects such as preference and uncertainty do no longer appear explicitly.
One possibility to combine the advantages from both approaches is to provide methods for transforming a decision-theoretic model into a rule-based system [BK02]. Such a transformation can be seen as a special type of knowledge compilation, that is the transformation of a formal representation of knowledge into an alternative form which is more suitable for the purpose at hand.
Here, we are concerned with the representation of decision knowledge. A decision-theoretic model is to be *approximated* by means of an efficient action model in the form of a fuzzy rule base. Transformation can be *approximate* in the sense that the input-output relation of the original model is not exactly reproduced. We pursue as transformation framework a *data-driven* approach. As decisive measure, a decision network is represented as set of situations with optimal decisions as specification of an optimal decision function. The objective is to approximate this function by means of a function induced by a fuzzy rule base. In this connection, the idea of *information granulation* plays an important role. Information granulation refers to the partitioning of an object into a set of granules, where each granule is a collection of basic entities drawn together, e.g., by indistinguishability, similarity, proximity, or functionality. More generally, granular computing [Yao00] is concerned with the systematic study of abstracting information and of processing information at different levels of abstraction [Sme96]. Information granulation provides a tool for modulating the level of abstraction and, hence, the complexity of a (decision) model. To sum up, the research focus of this thesis provides a decision maker an instrument to represent decision models on different levels of complexity, understandability and efficiency. Using rule bases for representing decision functions allows a decision maker to facilitate knowledge transfer based on the intelligible model property. Hence, the transformation framework introduces an adaptable tool to support reasonable decision making.

The thesis is structured in four parts. Part I gives a survey on Knowledge Based Systems. Knowledge-based techniques and methods to facilitate and support human behaviour in various disciplines like medicine or finance are introduced and different solutions are provided. Further, the strong interaction with uncertainty will be recapitulated. In Part II Graphical Models are presented. Two approaches of graphical representations are discussed and the underlying network technology, causal discovery, learning probabilities from data along with examples of using these technologies to develop probabilistic systems, and aggregation of models are presented. In Part III the transformation framework including a process schema and analysis criteria like granulation

is provided. In Part IV the concept realization and validation based on two industrial application fields are given. Part V closes this thesis with conclusions and gives an outlook and suggestions for future work.

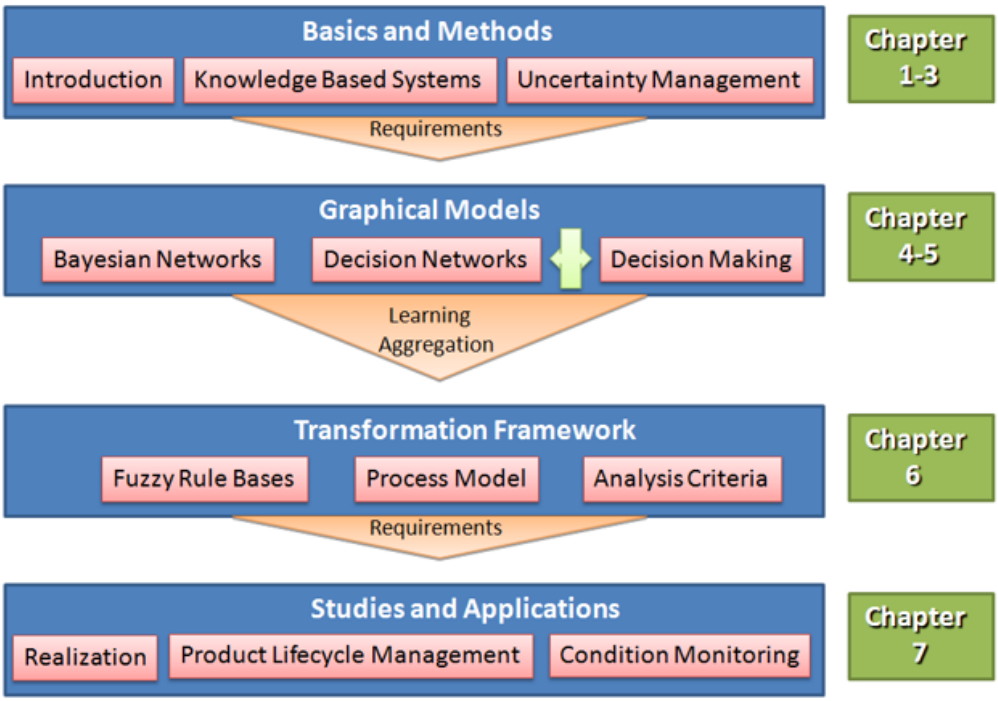


Figure 1.1.: Approach and structure of this thesis

Figure 1.1 above illustrates the approach and structure of the main parts of this thesis.

In the first chapter of Part I, i.e. Chapter 2, we introduce what Knowledge Based Systems are composed of and specify which fields of application are of broad interest. In a nutshell, Knowledge Based Systems take over the role of an expert to arrive at a solution in a problem-solving situation in a specific domain. Further, the knowledge based process is explained in this Chapter. The intersection with uncertainty is discussed in the second Chapter of Part I. Humans as system users plan, act, or handle in situations where facts are often partly missing, unknown or vague. Reasoning under uncertainty is associated with the handling of uncertain knowledge. Probability notations and foundations of the probability theory are given. One can see that different techniques for dealing with uncertainty exist and various techniques to solve them. Several of the most widely used approaches are sketched and classified.

Part II is about graphical models which are a special tool for handling uncertainty in complex domains. The first Chapter of Part II, i.e. Chapter 4, points out the strengths and weaknesses of Bayesian networks including the network technology and probabilistic inference. Also their area of application and methods for learning graphical models from data that aid a user in the task to develop an appropriate model for the domain under consideration are given. We have devised a new parameterized structure learning algorithm which is

introduced and evaluated applying a benchmark case.

Chapter 5 is dedicated to Decision networks as extension of Bayesian networks including decision and utility nodes based on the decision theoretic concept. Sequential and dynamic decision making aspects point out situations in which a decision maker has to select a sequence of action or a plan. This chapter also describes the possibility to order consequences by preference by using utility functions and methods for learning of preference models. Last, a method to facilitate the decision making process by adding specified information which enables to make a reasonable decision is presented.

Since Part III is about the transformation framework, in Chapter 6 the aforementioned concept is introduced. The principle of compiling decision networks is sketched and some optimization aspects are rendered. A strategy including process steps of turning a decision network into a fuzzy rule base to derive the latter directly from the formal specification of the network is presented as data-driven solution.

In the next Part IV the transformation framework concept is utilized. In Chapter 7, first, transformation results in the field of mechanical engineering are given. A specialized product lifecycle management approach with operation monitoring and maintenance of the product during its usage based on a condition monitoring method framework illustrates the applicability of this concept. In general, the derived rule base application supports quality control and fault management by quickly identifying problematic situations in order to prevent faults from causing too much damage and to improve the product quality.

Finally, Chapter 8 concludes this thesis and points to future work, respectively.

Part I.

Basics and Methods

2. Knowledge Based Systems

Knowledge-Based Systems concentrate on system solutions that enable the usage and further development of knowledge based techniques and methods to support human behaviour like decision making, learning, acting or planning. Artificial Intelligence (AI) researchers recognized that general problem-solving methods were insufficient to solve daily problems (i.e. medical diagnosis) accurately and comprehensibly. It was necessary to make knowledge available in specific well defined problem domains rather than general knowledge across many domains. Knowledge-Based Systems (KBS) are appropriate to cooperate with human users because quality and presentation issues are fundamental for interacting with the system. KBS applications are broadly diversified in many fields of applications like medical diagnosis (e.g. MYCIN[1] [BS84]), financial services (e.g. AUTHORIZER'S[2] as assistant system to support the credit authorization staff determine the credit level for credit card users), car assistant solutions or mechanical engineering (e.g. DELTA/CATS[3] as rule-based diagnostic system for troubleshooting electrical diesel locomotives, GENAID[4] remotely monitors and diagnoses the status of large electrical generators in real time). The construction of KBS can be subdivided in separate parts. Knowledge acquisition plays a fundamental and therefore critical part. It covers the activities of gathering, processing, and transforming knowledge into a form that can be handled by hardware systems like personal computers. This acquisition part is really time consuming and illustrates the bottleneck of construction and development of KBS [GD93]. Research results show two common approaches to resolve the acquisition problem. Acquire the knowledge directly from the domain expert or through historical records, i.e. by rule induction. The domain expert has considerable expertise in a specific domain, while a knowledge engineer is responsible for the construction of the system. KBS cover data and logic derived from experts in a field or from various sources within a knowledge base and applies this knowledge base to provide that kind of problem analysis that the domain expert might provide. KBS take over the role of an expert to arrive at a solution to a problem scenario from a specific domain. Distinguish the fundamental concept of KBS from general search-based problems found in the artificial intelligence researcher's group in the early 1970s (e.g. searching a problem space represented by a network or a tree). The first point is a separation of the knowledge from knowledgeable issues about the domain of the

[1] MYCIN was developed at Stanford University as medical diagnostic system which uses expert medical knowledge to determine the infectious agent in a patient's blood and prescribes a treatment for this infection

[2] developed by the Inference Corporation and the American Express Company

[3] developed by the General Electric Company

[4] developed by Westinghouse Electrical Corporation with assistance from Texas Utilities Generating Company

problem and how to use this knowledge. Second, dismiss application domains with a broad general knowledge by applying highly specific well defined domain knowledge and finally the heuristic rather than the algorithmic nature to achieve a solution to a request.

2.1. KBS Structure

Humans try to solve problems in general by classifying the problem into particular domains (e.g. start car engine scenario with battery, starter problem assumption). More specific knowledge about troubleshooting batteries or starters deals with individual circumstances of a request (e.g. battery charged, battery flat). A question is how to realize human problem solving with knowledge-based systems. From the structural point of view they consist of a *knowledge base* containing all necessary knowledge to solve a problem in a domain, an *inference engine* that manipulates the knowledge represented in the knowledge base activated when the user initiates the consultation session for the development of a solution to the initial issue described by the information in the database (see Figure 2.1). The inference engine responds to the question how to achieve the solution whereas the knowledge base contains knowledge about the particular problem describing how a car engine works or how to accomplish a diagnosis or maintenance task. A user interface interacts and exchanges with the knowledge base and inference engine of the KBS to produce and allocate the appropriate information to the user.

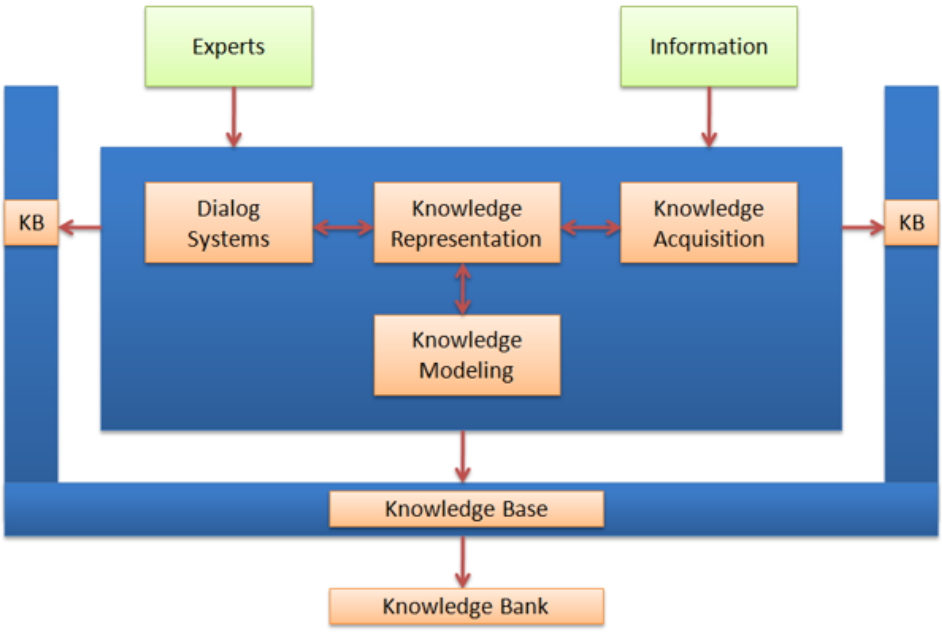


Figure 2.1.: General structure of a knowledge-based system [GD93]

The separation effect in differentiating knowledge from it's use allows to develop different applications by having to create only a new knowledge base

for each application. The generic reasoning technique is not modified (e.g. troubleshooting in medical diagnosis is similar to the diagnosis of a fault in a mechanical environment). Rule-based reasoning plays a fundamental role in describing the reasoning process and also in understanding human thinking and acting caused by their simplicity and similarity. Rules are an important knowledge representation paradigm often using the *IF-THEN* format, where *IF* stands for the premise as *condition* to test the trueness of a set of facts. In the positive case the action or *conclusion* as *THEN* portion is inferred as a new set of facts. Rules can be used to express many associations (e.g. IF ambient temperature is above 36 degree THEN weather is hot) for situations and actions, for premises and conclusions or for antecedents and their consequences. Rule-based systems use the modus ponens rule of inference (X is true and $X \mapsto Y$ is true, then Y is true, see for details also chapter 4) to handle and manipulate rules in providing the logical progression from initial data to the desired conclusion. This approach causes new facts to be derived and leads to a solution of the initial problem. A series of inference steps explains an inference chain for the progression steps. The way to obtain a comprehensible solution can differ and vary significantly. In the case of starting with all the known data and progress to the conclusion little data is required and many possible solutions are reachable. This data driven approach is well known as *forward chaining*. The opposite case is goal driven and selects a possible conclusion first and tries to prove its validity by looking for supporting evidence. This second inference mechanism is typically known as *backward chaining* and works similar to the depth-first search algorithm. Backward chaining decides if the existing facts support the derivation of a value for this starting conclusion. Disadvantages of such rule-based systems are the avoidance of infinite forward or backward chaining, the case of handle with additional new knowledge which can cause a contradiction and in the case of modifications to existing rules.

2.2. KBS Processes

2.2.1. Knowledge Acquisition

The knowledge acquisition process is the extraction of knowledge from sources of expertise and its transfer to the knowledge base in a formalized structured form. It is a difficult procedure to create the knowledge base which requires knowledge to be captured from people's heads [Mil07]. As possible systems for knowledge sources come into consideration experts, reports, books, guidelines, electronic information or end users. To overcome the knowledge acquisition bottleneck different stages have to be fulfilled. After knowledge sources are identified, conceptualization, formalization, implementation and test steps follow to organize appropriate knowledge, to formulate rules and to validate them accurately. Different applications were developed to support and facilitate the knowledge acquisition process. One of the systems, KADS[5] and the follower

[5]Knowledge Acquisition Documentation and Structuring Knowledge Analysis and Design System

CommonKADS (KADS II) [Sch00], was a structured, model-based development of KBS realized at the University of Amsterdam. It includes a methodology to manage knowledge engineering projects, a methodology for performing knowledge elicitation and an achievement perspective which includes a set of models from different aspects of a KBS and their environment to adapt continuously during life cycle. ACUDES[6] was designed and developed to improve the quality of the services provided in intensive care units. ACUDES manages the information regarding patient evolution in terms of the temporal sequence diseases suffered. Functionalities are supported by an ontology which facilitates knowledge acquisition while guaranteeing the semantic consistency of patient's evolution data [CPL+03].

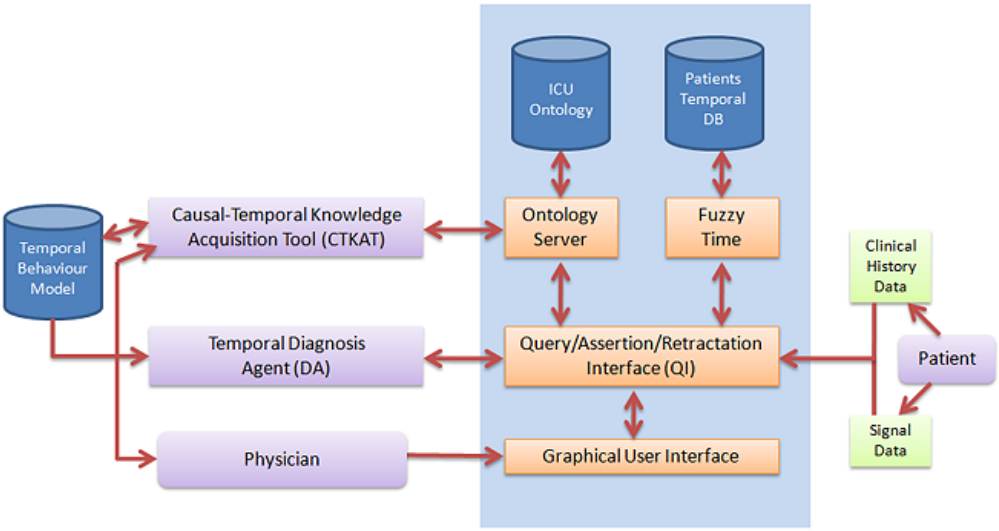


Figure 2.2.: ACUDES general architecture [CPL+03]

Figure 2.2 illustrates the generic ACUDES architecture for decision support systems in intensive care units. The knowledge acquisition tool CTKAT allows physicians to browse, modify, and manage the temporal behaviour model which is structured and integrated as causal network for the connection of each disease with abnormal manifestations and other diseases. In order to explain and describe a specific disease in detail, CTKAT interacts with the ICU ontology assuring the semantic consistency. The knowledge acquisition tool architecture supports users and guides the model building process with a side effect in reducing expert cognitive load. The temporal and causal model of diseases is adapted to the requirements imposed by the diagnosis agent. A fuzzy time component is based on a three-layered architecture which allows separating the interface for querying and updating temporal information from the layer where temporal entities and relations are managed, and from their low level representation. The system architecture can benefit from the major features of the temporal reasoner such as dealing with qualitative and quantitative temporal constraints and the efficient query answering process. Guided knowledge acquisition tools

[6] Architecture of Intensive Care Units Decision Support

offer more than only a knowledge editor to integrate the acquired knowledge from an expert. Locating bugs and belief revision tasks to compare existing knowledge with new knowledge should state more precision and exactness.

2.2.2. Knowledge Representation

Different representation techniques originate from human information processing. The main reason is the representation of knowledge in a form to facilitate inference mechanisms. The main representation schemes are (predicate) logic, rules, associative networks like semantic networks, frames, and object structures. Each technique uses different types of reasoning techniques to understand, interpret, and apply its knowledge. It is also possible to differentiate between declarative and procedural knowledge representations. In the first case knowledge is presented and stored as facts that must be interpreted, accessible for a variety of purposes but also inefficient for problem solving tasks. Procedural knowledge gets stored as algorithms in the program code and is therefore usable and efficient in specialized problem solving contexts.
The main representation schemes are illustrated and explained in the following:

- **Logic:** The role of logic based on the knowledge representation language, the study of truth conditions, and rules of inference. A logic consists of a formal system for describing states of affairs consisting of the syntax and semantics of the language [RN03]. The syntax consisting how to make sentences, the semantics determines the facts in the world from which the sentences refer. Including semantics means that each sentence makes a claim about the world. First-order logic constitutes a widespread representation technique in terms of objects and predicates on objects describing properties of objects or relations between objects. Among objects various relations hold (e.g. owner of, part of), partly realized as functions. Based on an ontological commitment properties of real-world sceanrios can be exemplified through causal (e.g. reflect the assumed direction of causality) or diagnostic rules (to infer the presence of hidden properties).

- **Rules:** Rules in production systems are a simulation of the cognitive behaviour of human experts. A production system keeps an ongoing memory of assertions as working memory. A production rule is a two-part structure comprising an antecedent set of conditions and a consequent set of actions. In a recognition step we have to find the rules which are applicable and satisfying the working memory. Among the rules partly conflict sets can occur, which must be resolved before executing rules. The working memory is updated with the conclusions of executed rules which are added as new facts.

- **Semantic networks:** Semantic networks are graphical notations for representing knowledge in patterns of interconnected nodes and arcs as links between nodes. A node stands for the representation of objects or concepts, whereas links represent relations between nodes. Links are labeled (e.g. is-a, has-a) and connected as a directed graph. In common sense

a semantic network is a declarative graphic representation that can be used either to represent knowledge or to support automated systems for reasoning about knowledge. Brachman [Bra83] initialized *definitional networks* emphasize the subtype as *is-a* relation as concept type and newly described subtype resulting in a general hierarchy with inheritance character for referring properties. The term *associative networks* is more precisely to express the understanding to represent physical and/or causal associations between various concepts or objects. This representation technique has the advantages to allow declarations of necessary and important associations explicitly and succinctly. A negative effect is the missing interpretation standard for the knowledge expressed within networks.

- **Objects and Frames:** Grouping items is a very natural way to think of knowledge itself not as a mere collection of sentences, but rather as structured and organized in terms of what the knowledge is about, the objects of the knowledge [BL04]. Conceptual objects deal to reify abstractions like events or relations. Each of this objects have its own parts constrained in different ways. The constraints between the parts might be expressed procedurally (e.g. reserving an airline seat). Minsky[7] suggested the idea of using object-oriented groups of procedures to recognize and deal with new situations. The idea adopts a rather more structured approach in collecting together facts about particular object and event types, and arranging the types into a large taxonomic hierarchy.

After introducing knowledge representation schemes the question of selecting the appropriate paradigm occurs. The general differentiation comprised two approaches. *Rules* have the opportunity causing inferences if the supporting facts are present. Diagnostic, classification, and interpretative applications are suited for this kind of paradigm. We pick *rule-based* techniques for characteristic knowledge representations which are not essential to represent the structure and relationships with concepts or objects. In contrast *structure* techniques have the feasibility for representing deeper and more structurally related knowledge. The selection decision depends on the necessity to provide inferential capability and whether the relationship is between facts or between structured entities.

2.2.3. Knowledge Modeling

Before starting the implementation phase of the system a model must be formulated and designed as simplification of the real task that bridges the gap between knowledge acquisition and its use. In business and economics, a model is a construct that represents business processes for process analyses by a set of variables and a set of quantitative relationships between them. This economic model represents a knowledge management strategy, and makes decision making feasible based on the simplification of an abstraction from observed data.

[7] Marvin Minsky is an American cognitive scientist in the field of artificial intelligence and co-founder of MIT's AI laboratory

Models are commonly used in scenarios where the expert is highly involved, e.g. knowledge elicitation, validation or cross-validation with other experts or knowledge publications. Pohlmann [Poh82] has introduced modeling steps for technical objects including abstraction (e.g. 3D model), formalization by using a model scheme (e.g. volumes, surfaces) and representation of the information model as internal representation. Particular the type of abstraction and the choice of a model scheme affects the modeling process. Kinds of geometric modeling for three dimensional models are such as of trees or boundary representations. The essential input to conceptual modeling is a knowledge intense task separated in the stages knowledge identification (e.g. identify information sources and key decision points), knowledge specification (e.g. choose task template, construct domain schema), and knowledge refinement (e.g. validate the model). Well-established knowledge modeling techniques are the concept ladder (shows classes of concepts and their sub-types), composition ladder (shows the way a knowledge object is composed of its constituent parts), decision ladder (shows the alternative courses of action for a particular decision), attribute ladder (shows attributes and values), and process ladder (shows processes consisting of the parts tasks and activities useful for representing process knowledge) [Reh06]. A concept of a map-based knowledge modeling approach is the PreSERVe method refined in work at the NASA Glenn research center[8], containing an effective elicitation method and a useful representation scheme based upon concept maps. A concept map is a type of diagram that shows knowledge objects as nodes and the relationships between them as links or labeled arrows. PreSERVe describes an iterative method of eliciting expert knowledge based on the steps preparation, scoping, elicitation, rendering and verification as a principle way to combine various knowledge elicitation strategies. Scoping is necessary based on the potentially changing scope and direction, elicitation can be categorized broadly into those that are direct or indirect, and rendering separates resource rendering to create or edit elements that will be included in the knowledge model from model rendering focuses on the assembly of these elements themselves. Centralized knowledge modeling facilitates to bridge the gap between knowledge acquisition and knowledge use based on simplification and enabling reasoning within an idealized framework.

2.2.4. Dialogue System

A dialogue system is responsible for the communication with the user being able to respond properly to a variety of requests. Such a system must facilitate the dialogue ability between the users and the system for different types of interaction like text, speech, graphics, haptics, gestures and other modes. It utilizes a variety of knowledge sources and models (e.g. knowledge of the dialogue, the task at hand, and the domain). The architecture contains different processing modules necessary to understand various involved reasoning mechanisms, for instance interpretation or dialogue management. The separation of different processing modules facilitates the portability of the dialogue system

[8] J.W. Coffey, Institutional memory preservation at the NASA Glenn Research Center, 1999

to new domains. Requests can be difficult to handle in the case of vagueness or ambiguity or requests which are related to the structures and properties of the application and also requests outside the scope of the application. It is a challenge to be able to access, gather, and integrate knowledge from various knowledge sources and application systems. From the users' point of view it is fundamental to use dialogue systems which are able to cognize the precise meaning of a request and to generate an appropriate answer. Dialogue systems have been optimized based on years of development and experience and numerous proven installations in the fields of airline and railway information, weather forecasts, ticket booking, help desk, multimedia services or routing systems. An occurring question in each dialogue system solution is how different user tasks can be performed. It is also important to handle factual information acts beside dialogue control acts. The former is used for requests where the response includes domain and task specific information or pointers to other information sources.

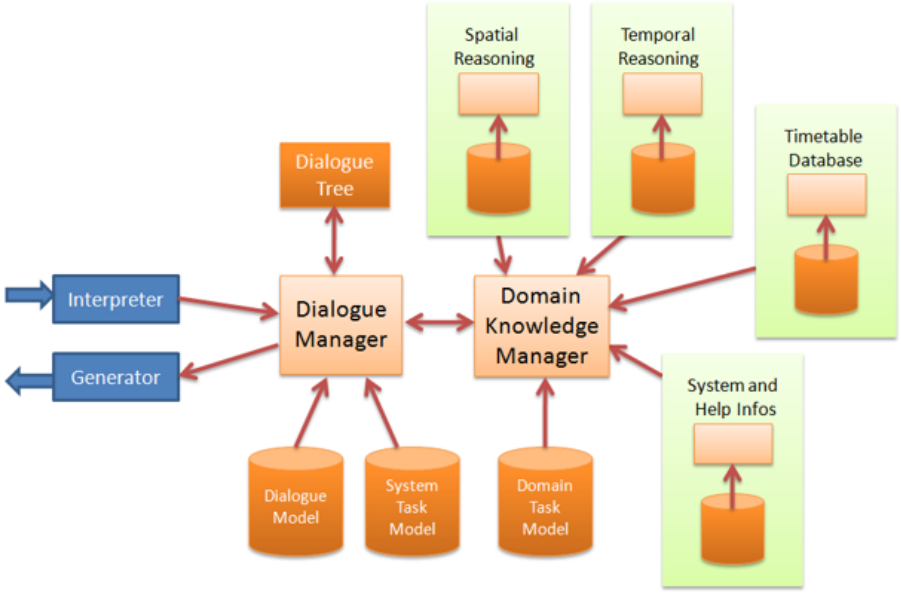


Figure 2.3.: MALIN dialogue system architecture [FEJ00]

To assemble dialogue systems a modular architecture is preferred including processing modules for interpretation, dialogue management, and domain management. Processing modules utilize various knowledge sources, such as grammar, dialogue model, domain model, and task model. The dialogue management is primarily responsible to control the flow of the dialogue by deciding how the system should respond to a user request. In the case of mistakes or missing information, clarification requests are formulated and posed to the system user. A general description is indispensable to decide which action has to be taken in a specific situation. A system task model deals with the information required for complex requests where the response includes information on what can be done with the system. The domain manager has an aligning function and

is responsible for retrieving and coordinating knowledge from different domain knowledge sources and application systems. The dialogue manager interacts with the domain manager in delivering a request and expecting an answer from the background system.
A representative for this kind of dialogue systems is MALIN[9] (see Figure 2.3), an application for time-table information for local bus traffic in a swedish town [FEJ00]. MALIN uses a combination approach where the dialogue manager is the central controller of the interaction and the domain knowledge manager acts and is based on an agent-related architecture. A dialogue history used for dialogue control, disambiguation of context dependent utterances, and context sensitive interpretation is maintained by the dialogue manager. Domain knowledge sources and application systems are implemented as (domain) agents, which provide different services to retrieve and reason about some information given some parameters. Agent communication and interaction is realized by passing messages. A domain knowledge miner is functional applying a spatial reasoner for the sub areas of the swedish town and also a temporal reasoner. A timetable agent retrieves trip information from the current internet based timetables.

2.3. KBS Review

To summarize, a knowledge-based system is capable of cooperating with human users to support decision making, learning, planning or acting. A KBS reflects the problem solving abilities of a human expert within a highly specific domain, separates the domain knowledge from how it is used, and comprises a heuristic rather than algorithmic way of proceeding. With this in mind, a KBS solves problems where algorithmic solutions either do not exist or are barely adequate to implement. An advantage is the ability to reproduce the knowledge and skills possessed by experts which allow wide distribution of expertise with access at any time. The separation of domain knowledge from the reasoning mechanism simplifies the modification of the knowledge base. The system user gets uniform answers to requests capable of solving problems whereby incomplete or vague data exist. Tracking the knowledge used to generate a solution assist the user by clarifying and justifying the results which serve as explanation. Disadvantages of knowledge-based systems are their possible lack of correctness and their limitation to a specific domain of expertise. They always try to deduce a solution, regardless of whether or not the problem is within the boundaries of expertise. Experts make mistakes which may be integrated in the KBS. Proving errors may thereafter occasion consequential costs. As a result of knowledge limited to specific domains, misleading or incorrect answers may be generated. A human user knows his limitation of knowledge and is able to qualify answers or does not attempt to solve problems related to other domains. However, a KBS always endeavors to infer a solution regardless of the field of expertise. A complex point of view of the KBS comes from the perspective

[9] MALIN: Multi-modal Application of LINLIN is a refinement of LINLIN to handle also multi-modal interaction and more advanced application

of the system developer. Developing a KBS includes the inference engine as well as the development shell. Involved in this development effort are decisions concerning how general the tool should be for the system user. A knowledge gap is the missing personalization alignment. For instance, system users have different degrees of prior domain knowledge, preferences or a partial order of the outcomes which should be pre-processed by the system. It would be interesting for research purposes to consider the problem of KBS decision parameter determination. For this purpose, methods based on experts and methods based on data are considerable. In the first case, one alternative is that the expert directly supplies the required decision parameters knowing the system user's point of view or supplies relevant information that is later used for decision parameter determination. Another interesting approach is based on data, where decision parameters are learned from examples almost automatically without conducting an expert interview. One alternative consists in having preferences or a partial order of the examples. Another alternative consists in having examples where each example is defined in terms of the inputs of the KBS as well as the intended output for this inputs.

3. Uncertainty Management

3.1. Introduction

Humans plan, act, or reason in situations where facts are often unknown or uncertain [VH06]. They do not have access to the complete environment to evaluate each scenario. Conditions and facts are unknown, incomplete or only crudely summarized. Supposedly, for example, a situation in which a car driver wants to drive a colleague to the next airport to catch a flight and considering the time leaving from home before the flight. In the case of planning 90 minutes leaving time before the estimated time of departure and a distance of 10 kilometers to the airport and an average rate of 60 kilometers/hour the car driver can only act under uncertainty. He is not able to follow if he gets into an accident, if a flat tire occurs or if he is caught up in a traffic jam during the driving time. But the handling of uncertain knowledge is calculated and well considered to avoid unproductive waiting time at the airport or speeding tickets along the driving way. Another situation is the codification of knowledge available in rules such as "dogs bark" or "much alcohol suggests long-term liver diseases". These rules include many exceptions which are not common to enumerate or ambiguously created or difficult to fulfill in real life situations. Reasoning in realistic domains requires some kind of simplification or adaption to deal with exceptions or to increase the degree of belief in decision making situations [Nea90]. The car driver can calculate a longer leaving time but also a longer waiting time at the airport, it depends on the importance of different goals (e.g. relaxed journey, natural environment) the car driver wants to achieve. Reasoning under uncertainty is interchangeably associated with the handling of uncertain knowledge. The **probability theory** has an outstanding position and serves as basis for human's behaviour in decision situations where they reason under uncertainty. Before introducing probability foundations and formulas have a closer look at the nature of uncertain knowledge. In many application fields like medicine uncertainty is all-embracing involved. If a doctor tries to build a diagnosis (e.g. cardiac trouble) he might infer from a symptom (e.g. pain in the left arm). But not all patients with a specific symptom have cardiac problems (e.g. bronchia). The enumeration of exceptions is the *OR* combination of several other problems with the same symptom as a long list of possible causes. In the case of specifying a causal (rule) dependence not all arm pains cause cardiac diseases. It is also possible that both disease states are unconnected. In practice, it is impossible to list the complete set of antecedents or consequents needed to ensure an exceptionless handling and the complete medical domain is too capacious to process all these sets. For a particular patient it would be necessary to make tests and analysis to avoid the handling of uncertain knowledge. A decision maker bypasses the enumer-

ation of exceptions by summarizing them in assigning each proposition to a numerical measure of uncertainty. Connect these measures according to a uniform syntactic principle based on truth values combined in logic. Truth values characterize formulas, uncertainty measures characterize the facts not visible or not covered in the formulas. Logical visible interactions enable to calculate the impact of each new fact in stages, propagate the impact of new facts on a set of sentences. Justifying this advancement under uncertainty includes restrictive assumptions of independence. But it is impossible to calculate impacts based on the complete set of previous observations to the complete set of sentences in a global step. Uncertainty solutions must solve occurring questions in how to represent uncertain data, how to combine pieces of uncertain data and how to draw inference with uncertain data. Artificial Intelligence researchers tackle these problems in dealing with uncertainty using different techniques, mainly separated in *quantitative* and *symbolic* approaches. Quantitative approaches using certainty factors [Voo96], fuzzy logic, belief functions, probabilities [Pea88] and possibilities [DLP94]. Symbolic approaches dealing for instance with circumscriptions or default logic. In real-world decision scenarios, the best choice is providing a degree of belief based on the probability theory assigning numerical degrees in the interval [0,1]. A probability with a value of 0.7 represents a 70% chance that a specific symptom causes a disease, it summarized all cases in which the factors needed a specific disease for a detailed symptom cause. The remaining 30% stands for all other possible causes. Probability values correspond with a degree of belief that facts do or do not hold. The probability of a statement depends on the percepts a human has received to date. In uncertain reasoning, probability statements must indicate the evidence with respect to which the probability is being assigned. If a human receives new or updates percepts, its probability assessments are updated to represent the new evidence. Based on probability theory a foundation for non-monotonic reasoning is given, whereas probabilistic networks allow the computation of quantified uncertainty. Statements in the form *IF X then Y* are quantifiable using probabilities.
Fuzzy logic (FL) is dealing with degrees of truth representing vagueness. Express and specify how well an object satisfies a vague description based on the fuzzy set theory. In the case of the statement "Henry is young" it is difficult to separate the true and false cases. Is it true, when Henry is 32 years old? Uncertainty about the world state or missing information to conclude is not the problem, the age of the evaluated person is well-known. In this case the meaning of the term *young* is realized as fuzzy predicate with a degree of true as number in [0,1]. In practice, FL is a truth-functional system in determining its truth value as a function of the truth values of its components.
Another interesting question is how to act in "I don't know?" situations like flipping a coin. In the fair case the probability for head (tail) is exactly 0.5. If the coin is manipulated and we have no additional information, then 0.5 is only a reasonable probability [RN03]. Probability theory can't differentiate both cases with the same probability value.
The Dempster-Shafer theory of evidence [Sha76] was developed as mathematical theory of evidence using interval-valued degrees of belief to represent the decision maker's knowledge of the probability of a proposition. Rather than

computing directly the probability of a proposition, it calculates the probability that the evidence supports the proposition measuring as belief function. In general, the belief value and the negation does not necessarily summarize to one as in the probability theory. If no information is accessible or justifiable, both values can also be zero (*Bel(head)=0 and Bel(¬head)=0*) and express the undecidable situation. If we have tested with 90 per cent certainty that the coin is fair, the believe value changes from *Bel(head)=0.5 and Bel(¬head)=0.5* to 0.45 ($0.9 \cdot 0.5$). Dempster-Shafer detects the gap not accounted by the evidence, but for a decision maker there is no adequate definition to support his decision process. The resulting probability interval can be a good aid in discussing about the necessity to acquire more evidence.

3.2. Basic Definitions

This chapter presents basic probability notations and foundations of probability theory which are essential for representing and reasoning with uncertain knowledge. The meaning of probability based on the syntax and semantics of probability theory provides everything to build an uncertain reasoning system. The probability theory constitutes the formal language of uncertainty and randomness. It also provides the basis for statistical inference. The basic problem studied in statistics is in inverse to the question considered in probability. Given the outcomes, what can be said about the process that generated the data? The axioms of probability including set theory, probability space and conditional probability were introduced in detail as fundamentals to capture uncertain knowledge in an efficient and natural way. Conditional independent relationships among variables can simplify the calculation and computation of query results used in a data structure to represent the dependence between variables and to provide a concise specification. The following basic definitions and notations of probability theory do not claim completeness. The interested reader can engross the thoughts and achieve comprehensive background information of probability and random variables in Papoulis & Pillai [PP02], Grimmett & Stirzaker [GS04], or [Ros00], where the presented material is gathered from.

3.2.1. Set Theory

Many everyday statements take the form "the probability of A is p", where A is some event like "tomorrow is rainy weather" and p is a number describing quantity. The occurrence or non-occurrence of A depends upon the chain (e.g. experiment) of involved circumstances. A *set* in general is a collection of well-defined and well-distinguished objects considered as a single whole. The objects are also called elements (e.g. car, pear, ballpen) and the set is also said to be the aggregate of these elements. In a set the order of its elements has no significance and multiplicity is generally also ignored. We think of the set Ω as a universal set if every set we consider is a subset of Ω. The set of all possible outcomes of an experiment is called the sample space denoted as Ω. A set containing no elements is called an empty set, denoted by $\emptyset$, while a set whose elements are

themselves sets is called a class. To indicate that ω is an element of the set Ω (also called sample outcome) write $\omega \in \Omega$ and $\omega \notin \Omega$ otherwise.

Example 3.1. A coin is tossed with two possible outcomes head (H) and tail (T), so that $\Omega = \{H, T\}$. The possible occurrences of the following events would be interesting, where each example can be specified as a subset A of the sample space Ω: outcome is a head ($A = \{H\}$), outcome is either head or tail ($A = \{H\} \cup \{T\}$), outcome is both head and tail ($A = \{H\} \cap \{T\}$), and outcome is not head ($A = \{H\}^c$).

If A and B are sets and every element of A is likewise an element of B, then A is called a subset of B, denoted by $A \subseteq B$. An axiom of set theory called extensionality says that two sets are equal if and only if they contain exactly the same elements. Then both $A \subseteq B$ and $B \subseteq A$ and we write $A = B$.

Definition 3.1. Let Ω be a universal set.

1. The *union* or *sum* of the sets $A_i;\ i = 1, \ldots, n$ is the set

$$A_1 \cup A_2 \cup \cdots \cup A_n = \bigcup_{i=1}^{n} A_i = \{\omega \in \Omega \mid \omega \in A_i \text{ for at least one } i\}.$$

   Similarly, the union of the infinite sequence of sets $A_i;\ i = 1, \ldots$ is the set

$$A_1 \cup A_2 \cup \cdots = \bigcup_{i=1}^{\infty} A_i = \{\omega \in \Omega \mid \omega \in A_i \text{ for at least one } i\}.$$

2. The *intersection* of the sets $A_i;\ i = 1, \ldots, n$ is the set

$$A_1 \cap A_2 \cap \cdots \cap A_n = \bigcap_{i=1}^{n} A_i = \{\omega \in \Omega \mid \omega \in A_i \text{ for every } i\}.$$

   Similarly, the intersection of the infinite sequence of sets $A_i;\ i = 1, \ldots$ is the set

$$A_1 \cap A_2 \cap \cdots = \bigcap_{i=1}^{\infty} A_i = \{\omega \in \Omega \mid \omega \in A_i \text{ for every } i\}.$$

   Notice that two sets are disjoint if and only if $A \cap B = \emptyset$.

3. The *difference* $A \setminus B$ of two sets A and B is defined by

$$A \setminus B = \{\omega \in \Omega \mid \omega \in A,\ \omega \notin B\}.$$

   Notice the definition of the *complement*

$$A^c = \{\omega \in \Omega \mid \omega \notin A\}$$

   as special case of the difference.

4. The *symmetric difference* $A \,\Delta\, B$ of two sets A and B is defined by

$$A \,\Delta\, B = \{\omega \in \Omega \mid (\omega \in A) \,\mathsf{XOR}\, (\omega \in B),\ \omega \notin B\}.$$

Union, intersection, difference, and symmetric difference are referred to as set operations.

Definition 3.2. The collection $A_1, A_2, \ldots$ of sets is *mutually* (or *pairwise*) *disjoint* if $A_i \cap A_j = \emptyset$ for every $i \neq j$. Then $\bigcup_{i=1}^{n} A_i$ sometimes is written as $\sum_{i=1}^{n} A_i$, and similarly for infinite sums.

Lemma 3.1. *For any sets $A, B, C \in \Omega$ the following identities hold:*

1. Associative laws

$$A \cup (B \cup C) = (A \cup B) \cup C, \qquad A \cap (B \cap C) = (A \cap B) \cap C$$

2. Commutative laws

$$A \cup B = B \cup A, \qquad A \cap B = B \cap A$$

3. Distributive Laws

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C), \qquad A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

4. Idempotent laws *and* complement laws

$$\begin{aligned} A \cup A &= A, & A \cap A &= A \\ A \cup A^c &= \Omega, & A \cap A^c &= \emptyset \end{aligned}$$

5. Identity laws *and* domination laws

$$\begin{aligned} A \cup \emptyset &= A, & A \cap \Omega &= A \\ A \cup \Omega &= \Omega, & A \cap \emptyset &= \emptyset \end{aligned}$$

6. De Morgan's laws

$$(A \cup B)^c = A^c \cap B^c, \qquad (A \cap B)^c = A^c \cup B^c$$

Proof. We only proof 1 here. The other identities are derived similarly.

$$\begin{aligned} A \cup (B \cup C) &= \{\omega \in \Omega \mid \omega \in A \vee (\omega \in B \cup C)\} \\ &= \{\omega \in \Omega \mid \omega \in A \vee (\omega \in B \vee \omega \in C)\} \\ &= \{\omega \in \Omega \mid (\omega \in A \vee \omega \in B) \vee \omega \in C\} \\ &= \{\omega \in \Omega \mid (\omega \in A \cup B) \vee \omega \in C)\} \\ &= (A \cup B) \cup C \end{aligned}$$

□

3.2.2. σ-Algebra and Measure Space

When thinking of a *non-deterministic experiment* it is intuitive to think of it as an experiment with an uncertain outcome. The set Ω of all conceivable outcomes associated with such an experiment is referred to as the *sample space*. Often it is not appropriate to consider the universal set Ω with all possible subsets on the whole. Instead, it is sufficient to think only of the collection of sets as a subcollection $\mathcal{A}$ of the set of all subsets of Ω. This subcollection should have certain properties in accordance with the earlier remarks.

Definition 3.3. Let Ω be a universal set. An *algebra* $\mathcal{A}$ is a nonempty collection of subsets of Ω, satisfying the axioms:

(A1) $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$

(A2) $A_1, A_2 \in \mathcal{A} \Rightarrow A_1 \cup A_2 \in \mathcal{A}$

Example 3.2. Let $\Omega = \{a, b, c, d\}$. Then $\mathcal{A}_1 = \{\emptyset, \{a, b, c\}, \{d\}, \Omega\}$ is an algebra while $\mathcal{A}_2 = \{\emptyset, \{a\}, \{b, c\}, \Omega\}$ is not.

The sets of an algebra $\mathcal{A}$ are called *events*. Mutually disjoint sets in $\mathcal{A}$ are called *mutually exclusive events*. Thus an algebra $\mathcal{A}$ can be thought of as a collection of events associated with an experiment whose sample space is Ω. It follows that algebras are closed under unions, intersections, and differences.

Lemma 3.2. *Let $\mathcal{A}$ be an algebra and $A_1, A_2, \ldots, A_n \in \mathcal{A}$. Then the following holds:*

1. $\emptyset, \Omega \in \mathcal{A}$
2. $A_1 \cap A_2 \in \mathcal{A}$
3. $A_1 \setminus A_2 \in \mathcal{A}$
4. $\bigcup_{i=1}^{n} A_i \in \mathcal{A}, \bigcap_{i=1}^{n} A_i \in \mathcal{A}$

Proof.

1. Because $\mathcal{A}$ is not empty, an $A \in \mathcal{A}$ exists, with $A \subset \Omega$. It then follows from (A1) that $A^c \in \mathcal{A}$, from (A2) that $A \cup A^c = \Omega \in \mathcal{A}$, and last from (A1) that $\Omega^c = \emptyset \in \mathcal{A}$.
2. $A_1 \cap A_2 = (A_1^c \cup A_2^c)^c \in \mathcal{A}$
3. $A_1 \setminus A_2 = A_1 \cap A_2^c = (A_1^c \cup A_2)^c \in \mathcal{A}$
4. By induction.

□

To investigate limits it is necessary that the system of sets $\mathcal{A}$ is not only closed under unions and intersections of *finite* many sets but also under unions and intersections of *countably infinite* many sets. This can be reached by adding the following axiom.

Definition 3.4. A *σ-algebra* is an algebra satisfying the additional axiom

(A3) $A_1, A_2, \ldots \in \mathcal{A} \Rightarrow \bigcup_{i=1}^{\infty} A_i \in \mathcal{A}$.

Definition 3.5. The pair or tuple $(\Omega, \mathcal{A})$ is called a *measurable space*, if $\mathcal{A}$ is a σ-algebra or σ-field relative to Ω. The elements of $\mathcal{A}$ are called *measurable sets*.

The smallest σ-algebra associated with Ω is the collection $\mathcal{A} = \{\emptyset, \Omega\}$. If A is any subset of Ω then $\mathcal{A} = \{\emptyset, A, A^c, \Omega\}$ is a σ-algebra. For every Ω the power set $\mathcal{P}$ is always also a σ-algebra. If Ω is finite or countably infinite, $\mathcal{A} = \mathcal{P}$ can be chosen. If Ω is uncountably (e.g. $\Omega = \mathbb{R}$ or $\Omega = [0, 1]$), a smaller σ-algebra has to be chosen.

To sum it up, with any experiment a pair is associated $(\Omega, \mathcal{A})$, where Ω is the set of all possible outcomes or elementary events and $\mathcal{A}$ is a σ-algebra of subsets of Ω which contains all the events in whose occurrences we are interested in.

3.2.3. Probability

Repeating an experiment with a large number N of times and keeping the initial conditions as equal as possible, and also suppose that A is some event which may or may not occur on each repetition. Most scientific experiments show that the proportion of times in which A occurs settles down to some value as N becomes larger and larger. Consider the value of the ratio $N(A)/N$ with $N(A)$ for the number of occurrences of A in N trials as *probability* that A occurs on any particular trial.

Definition 3.6 (Kolmogorov Axioms of Probability)**.** Let $(\Omega, \mathcal{A})$ be a measurable space. A *probability function* (or *probability measure*) *on* Ω is a function $P : \mathcal{A} \to [0, 1]$ such that

(P1) P is non-negative: $P(A) \geq 0$ for each $A \in \mathcal{A}$

(P2) P is normed: $P(\Omega) = 1$

(P3) P is σ-additive: $P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i)$ for pairwise disjoint $A_1, A_2, \ldots \in \mathcal{A}$

$P(A)$ is termed *probability of the event* $A \in \mathcal{A}$. Let $(\Omega, \mathcal{A})$ be a measurable space and P be a measure on $\mathcal{A}$. Then the triple $(\Omega, \mathcal{A}, P)$ is called a *measure space*. If $P(\Omega) = 1$, then it is called a *probability space*. Note that in the case of countable Ω, assume $\mathcal{A} = 2^{\Omega}$ without loss of generality, and the probability of an event A is the sum of the probabilities of its elements:

$$P(A) = \sum_{\omega \in A} P(\{\omega\})$$

A probability $P(A)$ can be interpreted in the two main ways as frequencies and degrees of belief. Frequency interpretation estimates $P(A)$ as long as run proportion of times that A is true in repetitions. The degree of belief $P(A)$ is the strength of belief of an individuum in the truth of A. In economic sense, it is the price that a person is willing to pay for a bet in which this person wins 1 monetary unit in case A is true and zero otherwise [Sor00]. In statistical inference, the two interpretations lead to different attitudes, the *Frequentist* and the *Bayesianist*.

Theorem 3.1. *Let $(\Omega, \mathcal{A}, P)$ be a probability space and the event $A \in \mathcal{A}$, then the following properties are verified:*

1. $P(\emptyset) = 0$
2. $P(A) \leq 1$
3. $P(A^c) = 1 - P(A)$

Proof.

1. Since Ω and $\emptyset$ are pairwise disjoint, by axiom (P3) then $P(\Omega) = P(\Omega \cup \emptyset) = P(\Omega) + P(\emptyset)$. Also, $P(\Omega) = 1$ by (P1), hence $1 = 1 + P(\emptyset)$, so that $P(\emptyset) = 0$.
2. $1 = P(\Omega) = P(A \cup A^c) = P(A) + P(A^c)$ by (P2) and (P3). Since $P(A)$ and $P(A^c)$ are non-negative by P(1), then $P(A) \leq 1$.
3. This follows immediately from $1 = P(\Omega) = P(A) + P(A^c)$.

□

Theorem 3.2. *Let $(\Omega, \mathcal{A}, P)$ be a probability space and $A, B \in \mathcal{A}$ be two events, then*

1. $A \subseteq B \Rightarrow P(A) \leq P(B)$
2. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
3. $P(A \cup B) \leq P(A) + P(B)$

Proof.

1. $P(B) = P(A \cup (B \setminus A)) = P(A) + P(B \setminus A)$. Since $P(B \setminus A) \geq 0$ by (P1), then $P(B) \geq P(A)$.
2. Since $P(A \setminus B) = P(A) - P(B)$ for $A, B \in \mathcal{A}$ with $A \supset B$ and

$$A \cup B = (A \setminus (A \cap B)) \cup (A \cap B) \cup (B \setminus (A \cap B))$$

it holds

$$\begin{aligned} P(A \cup B) &= P(A \setminus (A \cap B)) + P(A \cap B) + P(B \setminus (A \cap B)) \\ &= P(A) - P(A \cap B) + P(A \cap B) + P(B) - P(A \cap B) \\ &= P(A) + P(B) - P(A \cap B). \end{aligned}$$

3. This follows immediately from 2.

□

With Theorem 3.2 immediately further properties of probability measures follow:

1. P is finitely additive:

$$P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i) \tag{3.1}$$

for every finite sequence $\{A_i\}_{i=1}^n$ of mutually disjoint events in $\mathcal{A}$.

2. Subadditivity of P:

$$P(\bigcup_{i=1}^{n} A_i) \leq \sum_{i=1}^{n} P(A_i) \tag{3.2}$$

for every collection $\{A_i\}$ of events in $\mathcal{A}$.

As a generalization of the second part of Theorem 3.2, also Poincaré-Sylvester's inclusion-exclusion formula arises.

Corollary 3.1. *For every $n = 1, 2, \ldots$ and every sequence $A_1, \ldots, A_n \in \mathcal{A}$ holds*

$$P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} (-1)^{i-1} \sum_{1 \leq k_1 < \ldots < k_i \leq n} P(A_{k_1} \cap \ldots \cap A_{k_i}) \,. \tag{3.3}$$

Proof. By induction.

Furthermore, it is possible to show with Theorem 3.2 that probability measures are continuous with respect to the monotone convergence of sets.

Corollary 3.2. *Let $A_1, A_2, \ldots \in \mathcal{A}$. Then it holds*

$$P(\bigcup_{i=1}^{\infty} A_i) = \lim_{i \to \infty} P(A_i), \quad \text{if } A_1 \subset A_2 \subset \ldots, \tag{3.4}$$

and

$$P(\bigcap_{i=1}^{\infty} A_i) = \lim_{i \to \infty} P(A_i), \quad \text{if } A_1 \supset A_2 \supset \ldots. \tag{3.5}$$

Proof. Omitted.

The subadditivity of probability measures is not only valid for two events as discussed in part 3 of Theorem 3.2 or for finite many events as shown in (3.2). It is also valid for infinite collections.

Theorem 3.3. *Let $(\Omega, \mathcal{A}, P)$ be a probability space and $\{A_i\}_{i=1}^{\infty}$ be a sequence of events from $\mathcal{A}$. Then it holds*

$$P(\bigcup_{i=1}^{\infty} A_i) \leq \sum_{i=1}^{\infty} P(A_i). \tag{3.6}$$

Proof. Instead of the sequence $\{A_i\}_{i=1}^{\infty}$, define the sequence $\{A'_i\}_{i=1}^{\infty}$ as

$$A'_1 = A_1, \qquad A'_i = A_i \setminus \bigcup_{k=1}^{i-1} A_k.$$

Now $A'_i \cap A'_j = \emptyset$ for $i < j$ so $\{A'_i\}_{i=1}^{\infty}$ is a sequence of disjoint sets. Since $\bigcup_{i=1}^{\infty} A'_i = \bigcup_{i=1}^{\infty} A_i$, and since $A'_i \subset A_i$, it holds

$$P(\bigcup_{i=1}^{\infty} A_i) = P(\bigcup_{i=1}^{\infty} A'_i) = \sum_{i=1}^{\infty} P(A'_i) \leq \sum_{i=1}^{\infty} P(A_i),$$

and the claim follows. □

3.2.4. Conditional Probability

Statements about changes take the form "if B occurs, then the probability of A is p", where A and B are events. Considering an experiment repeated N times, and on each occasion we observe whether or not the two events A and B occur. Suppose only an interest in cases with outcomes for which B occurs and all other experiments are disregarded. Given that an event B occurs in the case that A occurs if and only if $A \cap B$ occurs.

Definition 3.7 (Conditional Probability)**.** For two events $A, B \subseteq \Omega$ such that $P(B) > 0$, the *conditional probability* of A given B is defined by

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}.$$

Example 3.3. Two fair dice are thrown. Given that the first shows 3, what is the probability that the total exceeds 6? The answer is 1/2, since the second shows 4, 5, or 6 ($A \cap B = \{(3,4),(3,5),(3,6)\}$).

Example 3.4. A family has two children. What is the probability that both are boys, given that at least one is a boy? Represent the sample space as $\Omega = \{GG, GB, BG, BB\}$ with $P(BB) = 1/4$. Applying the definition of conditional probability, it follows $P(BB \mid \text{one boy at least}) = P(BB) \;/\; P(GB \cup BG \cup BB) = 1/3$.

Theorem 3.4 (Total Probability)**.** *Let $A_1, ..., A_k$ be a partition of Ω, i.e., $A_1 \cup ... \cup A_k = \Omega$ and $A_i \cap A_j = \emptyset$ for $1 \leq i \neq j \leq k$. Then for any arbitrary event B holds*

$$P(B) = \sum_{i=1}^{k} P(B \mid A_i) P(A_i). \tag{3.7}$$

Proof. Obvious $B = B\Omega = B(A_1 \cup ... \cup A_k) = BA_1 \cup ... \cup BA_k$. But the events BA_i and BA_j are mutually exclusive because the events A_i and A_j are as well mutually exclusive. Hence $P(B) = P(BA_1) + ... + P(BA_k)$ and (3.7) follows based on the conditional probability. □

Since $P(BA_i) = P(A_i \mid B)P(B)$ conclude that

$$P(A_i \mid B) = P(B \mid A_i)\frac{P(A_i)}{P(B)}.$$

and obtain with (3.7) the *Bayes theorem.*

Theorem 3.5 (Bayes' Theorem or Bayes' Rule)**.** *Let $A_1, ..., A_k$ be a partition of Ω such that $P(A_i) > 0$, $i = 1, ..., k$. Furthermore let $B \subseteq \Omega$ such that $P(B) > 0$. Then*

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{P(B)} = \frac{P(B \mid A_i)P(A_i)}{\sum_{j=1}^{k} P(B \mid A_j)P(A_j)} \tag{3.8}$$

for all $i = 1, ..., k$. The terms a priori and a posteriori are often used for the probabilities $P(A_i)$ and $P(A_i \mid B)$.

Proof. Definition 3.7 can be rewritten as $P(A \cap B) = P(A \mid B)P(B)$ which is also called *product rule*. The conjunction can also be expressed as $P(B \cap A) = P(B \mid A)P(A)$. Equating the two right-hand sides and dividing by $P(B)$ we get (3.8). □

The Bayes' Rule is a suitable means by updating uncertain belief, represented in terms of a probability measure $P(\cdot)$ in the view of new information. Given this information, which is represented by the event B, the measure $P(\cdot)$ is replaced by the measure $P_B(\cdot \mid B)$. Bayesian inference provides the basis of probabilistic expert systems in the field of artificial intelligence.

Example 3.5. The Bayes' rule is very useful in practical fields like medical diagnostics because there are many cases where we have good probability estimates to compute the missing conditional probability. A doctor knows for instance that the disease meningitis causes the patient to have a stiff neck in 50% of the time [RN03]. The doctor also knows the prior probability of a patient having meningitis is 1/50000, and the prior probability of any patient having a stiff neck is 1/20. Calculating with this required three terms the missing probability of meningitis given a stiff neck.

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002. \tag{3.9}$$

The doctor expects only one in 5000 patients with a stiff neck having meningitis, even though a stiff neck is strongly indicated by meningitis in 50% of the cases. We come in a tenuous situation with crop up epidemic of meningitis. This has a direct influence to the unconditional probability of $P(M)$. Fortunately, the causal information $P(S \mid M)$ is unaffected by the epidemic, it compiles the way how meningitis works. This kind of causal knowledge provides the robustness needed to make probabilistic systems feasible in real world scenarios.

3.2.5. Independence

The occurrence of some event B changes the probability that another event A occurs in general by replacing $P(A)$ by $P(A \mid B)$. If the probability remains

unchanged ($P(A \mid B) = P(A)$, then we call A and B *independent*.

Definition 3.8 (Independent Events)**.** Two events $A, B \subseteq \Omega$ are called *independent* if

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}.$$

More generally, a set of events $\{A_i\} : i \in I\}$ is called independent if

$$P(\bigcap_{i \in J} A_i) = \prod_{i \in J} P(A_i).$$

for all finite subsets J of I.

For a given probability measure, the independence of events can be checked by verifying the definition in (3.8). Independence is often assumed in order to construct a probability measure. For example, in order to construct a probability measure for the experiment "tossing two dice", the outcome of the two dice is usually considered as independent.
Conditional independence is used in the multi-valued case. To say that A and B are independent with a given C can be expressed as $P(A \mid B, C) = P(A \mid C)$. The corresponding simplification of Bayes' rule for multiple events is $P(C \mid A, B) = \alpha P(C)P(A \mid C)P(B \mid C)$, where α is a *normalized* constant such that the entries in $P(C \mid A, B)$ sum to one.

3.2.6. Random Variables

The quantities governed by randomness are corresponding with functions on the probability space. In practice, it is often interesting to achieve some consequence of an experiment's random outcome. For instance, a fair coin is tossed twice. Many gamblers are more concerned with their losses than with the games which give rise to them. Such consequences, when really valued, may be thought of as functions which map Ω into the real line $\mathbb{R}$, and these functions are called random variables. The value taken by a random variable is subject to chance, and the associated likelihoods are described by a function called the distribution function.

Definition 3.9. Let $(\Omega, \mathcal{A}, P)$ be a probability space. A *random variable* is a mapping $X : \Omega \to \mathbb{R}$ such that

$$\{\omega \mid \omega \in \Omega,\, X(\omega) \leq x\} \in \mathcal{A}, \quad \forall\, x \in \mathbb{R}\,. \tag{3.10}$$

The regularity condition (3.10) is called *measurability* of the mapping X with respect to the σ-algebra $\mathcal{A}$. Such a function is also said to be $\mathcal{A}$-measurable. Often we are not only interested in the probability that the values $X(\omega)$ of a random variable X do not exceed a given threshold x, i.e. take values in the interval $B = (-\infty, x]$. Rather we are interested in the probability that X takes values in a more general subset $B \subset \mathbb{R}$, where B might be the union of disjoint intervals. Therefore not only in the sample space, but also in the event space a system of subsets is considered that is closed under the set operations

$\cup, \cap,$ and $\setminus$. Usually the Borel σ-algebra $\mathcal{B}(\mathbb{R})$ is considered, that is defined as the minimal σ-algebra of subsets of $\mathbb{R}$ containing all open sets (a, b), with $-\infty < a < b < \infty$. Hence, $\mathcal{B}(\mathbb{R}) = \sigma(\{(a, b), -\infty < a < b < \infty\})$ is a generating system. In particular, $\mathcal{B}(\mathbb{R})$ also contains all half-open resp. closed intervals, since e.g. $(a, b] = \bigcap_{n=1}^{\infty}(a, b + n^{-1}) \in \mathcal{B}(\mathbb{R})$ holds.

Thus, we can give an equivalent to the regularity condition (3.10) as follows:

Theorem 3.6. *Let $(\Omega, \mathcal{A}, P)$ be a probability space. A mapping $X : \Omega \to \mathbb{R}$ is a random variable if*

$$\{\omega \mid \omega \in \Omega,\, X(\omega) \in B\} \in \mathcal{A}, \quad \forall\, B \in \mathcal{B}(\mathbb{R}) \tag{3.11}$$

where B is the Borel σ-algebra over $\mathbb{R}$.

Proof. Omitted.

Example 3.6. When rolling two dice, the sample space is given by $\Omega = \{\omega = (\omega_1; \omega_2),\, \omega_i \in \{1, \ldots, 6\}\}$. If we are interested in the sum of the two dice, then $\omega \to X(\omega) = \omega_1 + \omega_2$. The event that we get a sum of 4, for example, is given by $A = \{\omega \mid X(\omega) = 4\} = \{(1, 3), (2, 2), (3, 1)\}$, or in general $A = \{\omega \mid X(\omega) = k\}$ with $k \in \{2, \ldots, 12\}$. Questionable is the probability $P(A)$, thus it is necessary that $A \in \mathcal{A}$. It has to hold $\{\omega \mid \omega \in \Omega,\, X(\omega) = k\} \in \mathcal{A}$ for every $k = 2, \ldots, 12$. In this example this is equivalent to $\{\omega \mid \omega \in \Omega, X(\omega) \leq x\} \in \mathcal{A}$ for every $x \in \mathbb{R}$.

Note that a random variable X induces a probability measure $P_X(\cdot)$ on the measurable space, defined by $P_X(B) = P(B^{(-1)})$, for all $B \in \mathcal{B}$, where $B^{(-1)} = \{\omega \mid X(\omega) \in \mathcal{B}\}$.
A random variable can be considered as connection between sample spaces and data. The data that can be observed corresponds to realizations of a random variable X, the probability law governing the data generating process is characterized by an underlying probability space. In practical situations, the underlying space is often not mentioned explicitly, rather we work with the random variable and its distribution directly.

3.2.7. Cumulative Distribution Function

Theorem 3.6 leads to the definition of the *distribution* and the *cumulative distribution function* of a random variable X.

Definition 3.10. Let $(\Omega, \mathcal{A}, P)$ be a probability space and $X : \Omega \to \mathbb{R}$ an arbitarily random variable. The *distribution* of X is the set function $P_X : \mathcal{B}(\mathbb{R}) \to [0, 1]$ with

$$P_X(B) = P(\{\omega \mid \omega \in \Omega,\, X(\omega) \in B\}), \quad \forall\, B \in \mathcal{B}(\mathbb{R}). \tag{3.12}$$

The set function defined in (3.12) is a *probability measure* over the measure space $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$, since P_X is σ-additive and normed due to $P_X(\mathbb{R}) = P(\Omega) = 1$. Usually, (3.12) is written in its abbreviated form,

$$P(X \in B) = P(\{\omega \mid \omega \in \Omega,\, X(\omega) \in B\}), \quad \forall\, B \in \mathcal{B}(\mathbb{R})$$

and especially

$$P(X \leq x) = P\left(\{\omega \mid \omega \in \Omega,\, X(\omega) \leq x\}\right), \quad \forall\, x \in \mathbb{R}.$$

With the definition of the distribution (3.12), the *cumulative distribution function* of a random variable X can be defined.

Definition 3.11. The function $F_X : \mathbb{R} \to [0,1]$ with $F_X(x) = P(X \leq x)$ is called *cumulative distribution function* (cdf) of the random variable X.

Next, some properties of cdfs are discussed.

Theorem 3.7. *Let $X : \Omega \to \mathbb{R}$ be an arbitrary random variable and $F_X : \mathbb{R} \to [0,1]$ its cdf. Then it holds*

1. *Asymptotic behavior at infinity:*

$$F_X(-\infty) := \lim_{x \to -\infty} F_X(x) = 0, \qquad F_X(\infty) := \lim_{x \to \infty} F_X(x) = 1, \quad (3.13)$$

2. *Monotony:*

$$F_X(x) \leq F_X(x+h), \quad \forall\, x \in \mathbb{R} \text{ and } h \geq 0, \quad (3.14)$$

3. *Continuity from the right: $F_X(x)$ is continuous from the right, i.e. for every sequence $\{h_n\}$ with $h_n \geq 0$ and $\lim_{n\to\infty} h_n = 0$ it holds*

$$\lim_{n \to \infty} F_X(x + h_n) = F_X(x), \quad \forall\, x \in \mathbb{R}\,. \quad (3.15)$$

Proof.

1. Only the first part of (3.13) is shown. Since F_X is monotone we can w.l.o.g. assume that x converges monotone against $-\infty$. With Corollary 3.2 then follows

$$\lim_{x \to -\infty} F_X(x) = \lim_{x \to -\infty} P_X((-\infty, x]) = P_X\Big(\bigcap_{x \leq 0} (-\infty, x]\Big) = P_X(\emptyset) = 0.$$

   The proof of the second part of (3.13) is analogical.

2. Since $(-\infty, x] \subset (-\infty, x+h]$, it follows from the first part of Theorem 3.2 that

$$F_X(x) = P_X((-\infty, x]) \leq P_X((-\infty, x+h]) = F_X(x+h).$$

3. Analogue to the proof of the first part, from Corollary 3.2 follows that

$$\begin{aligned} \lim_{n \to \infty} F_X(x + h_n) &= \lim_{n \to \infty} P_X((-\infty, x + h_n]) \\ &= P_X\Big(\bigcap_{n \geq 1} (-\infty, x + h_n]\Big) \\ &= P_X((-\infty, x]) \\ &= F_X(x). \end{aligned}$$

□

With the distribution function F_X also the following probabilities can be described

$$P(a \leq X \leq b), \quad P(a < X \leq b), \quad P(a < X < b), \quad P(a \leq X < b),$$

because for example

$$\begin{aligned} P(a \leq X \leq b) &= P(\{X \leq b\} \setminus \{X < a\}) \\ &= P(X \leq b) - P(X < a) \\ &= F_X(b) - \lim_{h \to 0} F_X(a - h). \end{aligned}$$

However, in general $F_X(a) = \lim_{h \to 0} F_X(a - h)$ does not hold, but

$$F_X(a) = \lim_{h \to 0} F_X(a - h) + P(X = a). \tag{3.16}$$

In Theorem 3.7 it is shown that distribution functions are monotone and bounded. Hence, they can have for every $\varepsilon > 0$ only countably many jump discontinuities with jumps higher then ε; and thus only countably many jump discontinuities can exist.

Theorem 3.8. *Let $X : \Omega \to \mathbb{R}$ be an arbitrarily random variable. Then the distribution P_X of X is uniquely determined by the cdf F_X of X.*

Proof. Omitted.
The cdf of a random variable completely determines the distribution of that random variable. More specifically, if two random variables X and Y have the same cdf, then $P_X(A) = P_Y(A)$ for all measurable events A. A mapping $F_X : \mathbb{R} \to [0, 1]$ is a cdf for some probability $P(\cdot)$ if and only if F satisfies the following three properties:

- F is non-decreasing, i.e., $F(x_1) \leq F(x_2)$ for x_1, x_2,
- F is normalized, i.e., $lim_{x \to -\infty} = 0$ and $lim_{x \to +\infty} = 1$,
- F is right continuous, i.e., $lim_{y \searrow x} F(y) = F(x)$ for all x.

3.2.8. Discrete and Continuous Variables

Much of study in random variables is devoted to distribution functions. The general theory of distribution functions and their applications is quite difficult and abstract relying on a treatment of the construction of the Lebesque integral. Consider certain subclasses of random variables in depth, here the collection of discrete and continuous random variables.

Definition 3.12. A random variable X is *discrete* if it takes countably many values $x_1, x_2, ...$ In this case, the *(probability) mass function* for X is defined by

$$f_X(x) = P(\{x\}) = P(X = x), \tag{3.17}$$

and the cdf is simply given by

$$F_X(x) = P(X \leq x) = \sum_{i:x_i \leq x} f_X(x_i). \tag{3.18}$$

Lemma 3.3. *The (probability) mass function $f : \mathbb{R} \to [0,1]$ satisfies*

1. *the set of x such that $f(x) \neq 0$ is countable,*
2. *$\sum_i f(x_i) = 1$, where $x_1, x_2, ...$ are the values of x such that $f(x) \neq 0$.*

Proof. Omitted.

Example 3.7. A coin is tossed n times, and head turns up each time with probability $p = 1 - q$. Then $\Omega = \{H, T\}^n$. The total number X of heads takes values in the set $\{0, 1, 2, ..., n\}$ and is a discrete random variable. Its (probability) mass function satisfies $f(x) = 0$ if $x \notin \{0, 1, 2, ..., n\}$. Let $0 \leq x \leq n$, and consider $f(x)$. Exactly $\binom{n}{x}$ points in Ω give a total of k heads, each of these points occurs with probability $p^k q^{n-k}$, and so

$$f(x) = \binom{n}{x} p_x (1-p)^{n-x}. \tag{3.19}$$

The random variable X is said to have the *binomial distribution* with parameters n and p.

Definition 3.13. The random variable $X : \Omega \to \mathbb{R}$ (resp. its distribution) is called *absolutely continuous*, if the cdf F_X of X has the following integral representation

$$F_X(x) = \int_{-\infty}^{x} f_X(y) dy, \quad \forall\, x \in \mathbb{R} \tag{3.20}$$

where $f_X : \mathbb{R} \to [0, \infty)$ is a non-negative (Lebesgue-integrable) function, referred to as *probability density function* (pdf) or *density* of X.

The integral in (3.20) is usually regarded as Lebesgue integral.

The cdf F_X (and thus also the distribution P_X) of an absolutely continuous random variable X is in the following sense completely determined by a pdf f_X.

Theorem 3.9.

1. *The random variable $X : \Omega \to \mathbb{R}$ is absolutely continuous, if the distribution P_X of X can be written in the following form:*

$$P_X(B) = \int_B f_X(y)\, dy, \quad \forall\, B \in \mathcal{B}(\mathbb{R}) \tag{3.21}$$

2. *Let $X, Y : \Omega \to \mathbb{R}$ be absolutely continuous random variables. Then $P_X = P_Y$, if and only if*

$$f_X(x) = f_Y(x) \tag{3.22}$$

for almost all $x \in \mathbb{R}$, i.e. (3.22) holds for all $x \in \mathbb{R} \setminus B$, where the set of exceptions $B \subset \mathbb{R}$ has Lebesgue measure zero.

Proof. Omitted.

Often the density f_X is an (at least piecewise) absolutely continuous function. If X is absolutely continuous, then the cdf F_X has no jumps. With (3.16) follows especially

$$P(X = x) = 0, \quad \forall\, x \in \mathbb{R}. \tag{3.23}$$

To describe an absolutely continuous random variable X it is sufficient to regard its pdf f_X, since f_X uniquely determines the cdf F_X and with this also the distribution P_X of X. Some examples of pdfs are given in the next definition.

Definition 3.14. Let X be a continuous random variable. It is said to possess

(a) *uniform distribution*, $\mathcal{U}(a, b)$, with support (a, b) if the pdf is

$$f_X(x; a, b) = \begin{cases} \frac{1}{b-a} & \text{if } a < x < b \\ 0, & \text{otherwise} \end{cases} \quad \forall\, x \in \mathbb{R}, \tag{3.24}$$

(b) *normal (gaussian) distribution*, $\mathcal{N}(\mu, \sigma^2)$, with parameters $\mu \in \mathbb{R}$ and $\sigma > 0$ if the pdf is

$$f_X(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \quad \forall\, x \in \mathbb{R}. \tag{3.25}$$

Many interesting probabilistic statements about a pair X, Y of variables concern the way X and Y vary together as functions on the same domain Ω.

Definition 3.15. The *joint distribution function* of X and Y is the function $F : \mathbb{R}^2 \to [0, 1]$ given by

$$F_{X,Y}(x, y) = P(X \le x, Y \le y). \tag{3.26}$$

Their *joint mass function* $f : \mathbb{R}^2 \to [0, 1]$ is given by

$$f_{X,Y}(x, y) = P(X = x, Y = y). \tag{3.27}$$

If X and Y are continuous, it is essential to use another density function called probability density function for the random variables X and Y.

Definition 3.16. The random variables X and Y are jointly continuous with *joint (probability) density function* $f : \mathbb{R}^2 \to [0, \infty)$ if

$$F(x, y) = \int_{v=-\infty}^{y} \int_{u=-\infty}^{x} f(u, v)\, du\, dv, \quad \forall\, x \in \mathbb{R}. \tag{3.28}$$

Definition 3.17. If (X, Y) have a joint distribution with a mass function $f_{X,Y}$, then the *marginal mass function* for X is defined by

$$f_X(x) = P(X = x) = \sum_y P(X = x, Y = y) = \sum_y f_{X,Y}(x, y) \tag{3.29}$$

and the marginal mass function for Y by

$$f_Y(x) = P(Y = y) = \sum_x P(X = x, Y = y) = \sum_x f_{X,Y}(x, y). \tag{3.30}$$

For continuous random variables, the marginal density is

$$f_X(x) = \int f(x, y)\, dy \;\; and \;\; f_Y(y) = \int f(x, y)\, dx \tag{3.31}$$

Definition 3.18. Two random variables X and Y are *independent* if, for every A and B,

$$P(X \in A, Y \in B) = P(X \in A)P(Y \in B). \tag{3.32}$$

Random variables with joint pdf $f_{X,Y}$ are independent if and only if

$$f_{X,Y} = f_X(x)\, f_Y(y) \quad \forall x, y. \tag{3.33}$$

If $X_1, ..., X_n$ are independent random variables and each has the same marginal distribution with cdf F, we say that $X_1, ..., X_n$ are independent and identically distributed: $X_1, ..., X_n \sim F$.
$X_1, ..., X_n$ are called a random sample of size n from F. It holds that

$$P(X_1 \in A_1, ..., X_n \in A_n) = \prod_{i=1}^{n} P(X_i \in A_i)$$

and

$$f(x_1, ..., x_n) = \prod_{i=1}^{n} f_{X_i}(x_i).$$

3.3. Dempster-Shafer Theory of Evidence

The *Dempster-Shafer theory* is a mathematical theory of evidence based on *belief functions* and plausible reasoning, which is deployed to combine pieces of information to calculate the probability of an event. In a first step, Dempster generalized a method for computing probabilities for statistical parameters from observations by a rule of combination explained later in detail. Shafer's most influential contributions were simplifications as mathematical theory of evidence [Sha76] based on non-Bayesian weights of evidence. Dempster-Shafer theory is well-known as mass theory to model imperfect data. Smets [Sme96] introduces the perception of imperfection, if imprecision, uncertainty and inconsistency are existent. To consider the different meanings of imprecision and uncertainty regard the following situation. The information "Peter has at least one or two

children“ contains that the number of children is imprecise but certain. In the case “Peter has two children, but its not sure“ the number of children is precise but uncertain. Inconsistency means conflicting or contradictory information.
Several mathematical models have been proposed to model degrees of belief. If there exists a probability measure with known values, the Bayesian approach finds favour. If there exists a probability measure but with some unknown values, the usage of belief models based on other premises and belief functions is necessary. If the existence of a probability measure is not known, use the transferable belief model that sets out to represent someone's beliefs. In such representation, it uses a belief function where $bel(A)$ is *true*. In probability theory, the component consists of the assessment of a probability density p on the elements of Ω in a way that $p : \Omega \rightarrow [0,1]$ with $\sum_{\omega \in \Omega} p(\omega) = 1$. Probability-based evidential reasoning represent the certainty in an attribute value as a point probability. In Dempster Shafer (DS), the relevant characteristics of the world are represented as a finite set of mutually exclusive statements or propositions called the *frame of discernment* (Θ). In other words, each piece of evidence leads to the allocation of parts of some initial amount of belief to subsets of the universe of discourse. If all these parts were allocated only to the singletons of Θ, the resulting model will correspond directly to the Bayesian approach. In general, parts may be allocated to subsets ($X \subseteq \Theta$) representing that part of our belief that supports some subset of the frame of discernment. Because of the lack of information, it does not support any more specific subsets. DS assigns a belief mass to each subset of the power set called *basic belief assignment*. A basic belief assignment m on the universe Θ is a function

$$m : 2^{\Theta} \rightarrow [0,1]$$

with

$$m(\emptyset) = 0 \quad and \quad \sum_{X \subseteq \Theta} m(X) = 1.$$

In contrast to the probability theory with point probabilities $P(X) \in [0,1]$ DS allocates interval valued degrees $[Bel(X), Pl(X)] \subseteq [0,1]$. The *degree of belief* $Bel(X)$ given to the set X of Θ is defined as the sum of all masses that support the set X

$$Bel(X) = \sum_{Y \subseteq X} m(Y).$$

The *degree of plausibility* $Pl(X)$ quantifies the total amount of belief that might support X:

$$Pl(X) = 1 - Bel(\neg X) = 1 - \sum_{Y \subseteq \neg X} m(Y) = \sum_{X \cap Y \neq \emptyset} m(Y).$$

In any modeling of uncertain situations, it is necessary that a method is introduced and provided for making decisions used as a decision support tool. In the case where we must *bet* on the elements of Ω, build on Ω a probability distribution as $Bet\ P$ derived from the belief function that describes the creedal state. For all elements $x \in \Omega$ follows $Bet\ P(x) = \sum_{x \in A \subseteq \Omega} m(A)|A|$, where $|A|$

is the number of elements of Ω in A.
Sugeno has introduced a confidence measure [Sug77] and proposed a set of axioms to interpret a degree of confidence in decision situations. A function $g : 2^{\Omega} \rightarrow [0, 1]$ is a sugeno measure, if the conditions (1.)-(3.) hold:

1. boundary: $g(\emptyset) = 0$ and $g(\Omega) = 1$
2. monotony: $X, Y \in 2^{\Theta}$ and $X \subseteq Y \Rightarrow g(X) \leq g(Y)$
3. continuity: if $X_i \in 2^{\Theta}$ either $X_1 \subseteq X_2 \subseteq ...$ or $X_1 \supseteq X_2 \supseteq ...$ then $lim_{i\rightarrow\infty}\, g(A_i) = g(lim_{i\rightarrow\infty}\, A_i)$

Interpret A as degree of confidence in a decision scenario to follow A as correct answer for considered questions. The boundary settings cover the complete ignorance ($g = 0$) and certainty cases ($g = 1$).
In DS, a belief or plausibility measure becomes a probability measure P, when each set $A \in 2^{\Theta}$ with focal elements $m(A) \neq 0$ is represented as singletons. If this case occurs, it follows $P(A) = Bel(A) = Pl(A)$, which follows from the Bel and Pl equations and we obtain the *additivity property* of probability measures. Determine any probability measure P on a finite set X by a probability distribution function $p : X \rightarrow [0, 1]$ via the formula $P(A) = \sum_{x\in A} p(x)$. From the point of DS theory[KF88], the function p is equivalent to the function m restricted to singletons.

Example 3.8. Suppose a doctor who suspects pneumonia in a specific patient in the medical practice. The doctor takes an accurate medical history and performs a physical examination. Bacterial pneumonias often break out suddenly by creating a cough that produces mucous, fever, and pain along the chest wall. Some of the bacteria that can cause pneumonia are legionella (L), pneumococcus (P), mycoplasma (M), chlamydia (C), and klebsiella (K). The frame of discernment consists of $\Theta = \{P, L, K, C, M\}$. The basic probability assignment m as degree of evidential support is given as $m(\{P\}) = 0.3$, $m(\{L\}) = 0.2$, $m(\{P, L\}) = 0.4$, $m(\{P, K, C\}) = 0.1$, and $m = 0$ in all other cases. The calculation for the belief measure of the pair $\{P, L\}$ follows in applying the $Bel(P, L)$ equation.

$$Bel(\{P, L\}) = \sum_{B\subseteq(P,L)} m(B) =$$
$$m(\{P\}) + m(\{L\}) + m(\{P, L\}) = 0.3 + 0.4 + 0.2 = 0.9$$

The quantification of the total amount of belief that might support the pair $\{P, L\}$ results as follows:

$$Pl(\{P, L\}) = \sum_{B\cap(P,L)\neq\emptyset} m(B) =$$
$$m(\{P\}) + m(\{L\}) + m(\{P, L\}) + m(\{P, K, C\}) =$$
$$0.3 + 0.2 + 0.4 + 0.1 = 1.0.$$

Dempster's rule allows the combination of different basic probability assignments by aggregation of two different bodies of evidence (F_1, m_1) and (F_2, m_2) on the same reference set. F denotes the set of all focal elements of m within the body of evidence. Dempster's rule of combination provides a function for combining evidential information provided by different sources. The addressed problem concerns how to combine two independent sets of mass assignments. The information provided by two evidential sources, represented as basic probability assignments m_1 and m_2 over a common universe Θ may be combined to a joint basic probability assignment *bpa* over the same universe denoted by $m_1 \otimes m_2$.
The product of two bpa's $m_1, m_2 : 2^\Theta \rightarrow [0, 1]$ over the same universe Θ is defined as follows:

$$(m_1 \otimes m_2)(X) : \sum_{Y \cap Z = X} m_1(Y) m_2(Z)$$

It is clear to verify that this scheme yields a basic probability assignment due to the fact that it may assign a non-zero weight to the empty set. Therefore, the weight of the empty set is explicitly set to zero, and the rest of the weights is normalized by a factor of K^{-1}, where

$$K = \sum_{Y \cap Z = \emptyset} m_1(Y) m_2(Z)$$

K is a measure of the amount of conflict between the two mass sets as $(m_1 \otimes m_2)$. The normalization factor $1 - K$ has the effect of completely ignoring conflict and attributing any mass associated with conflict to the null set. The combination

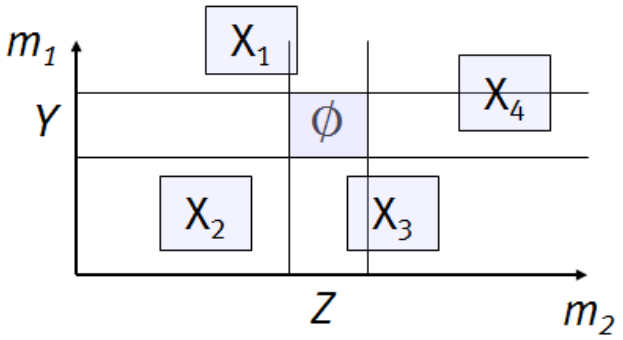


Figure 3.1.: Measure of the amount of conflict between the two mass sets m_1 and m_2

m of the bpa's m_1 and m_2 is defined as

$$m(X) = \frac{(m_1 \otimes m_2)(X)}{1 - (m_1 \otimes m_2)(\emptyset)} \quad \forall X \neq \emptyset, m(\emptyset) = 0.$$

The appliance of Dempster's rule is problematical in the case of high conflict measure between m_1 and m_2, and also caused by the implicit bpa's independence based on the rule definition.

Example 3.9. Suppose now two doctors who suspect pneumonia in a specific patient in the medical practice. Include the bacteria that can cause pneumonia as in the previous example: legionella (L), pneumococcus (P), mycoplasma (M), chlamydia (C), and klebsiella (K). The frame of discernment consists of $\Theta = \{P, L, K, C, M\}$. The basic probability assignments m_1 and m_2 are given as $m_1(\{P\}) = 0.1$, $m_1(\{L\}) = 0.2$, $m_1(\{P, L\}) = 0.4$, $m_1(\{P, K, C\}) = 0.1$, $m_2(\{L\}) = 0.5$, $m_2(\{P, C\}) = 0.3$, $m_2(\{P, L, K\}) = 0.2$, $m_2(\{P, L, K, C\}) = 0.1$, and $m_1, m_2 = 0$ in all other cases. The calculation of the measure of the amount of conflict between m_1 and m_2 delivers $\sum_{Y \cap Z = \emptyset} m_1(Y)m_2(Z)$:

$$m_1(\{P\})m_2(\{L\}) + m_1(\{L\})m_2(\{P, C\}) + m_1(\{P, K, C\})m_2(\{L\}) = 0.24.$$

The following intersections of m_1 and m_2 are non-empty subsets of Θ:
$\{P\}, \{L\}, \{P, L\}, \{P, C\}, \{P, K\}, \{P, K, C\}$. Exemplify the determination of Dempster's rule of combination for $m_{12}(\{L\})$:

$$m_{12}(L) = \frac{(m_1 \otimes m_2)(L)}{1 - (m_1 \otimes m_2)(\emptyset)} = \frac{(m_1 \otimes m_2)(L)}{0.76}.$$

$$(m_1 \otimes m_2)(L) = m_1(\{L\})m_2(\{L\}) + m_1(\{L\})m_2(\{P, L, K\}) + m_1(\{L\})m_2(\{P, L, K, C\}) + m_1(\{P, L\})m_2(\{L\}) = 0.36.$$

Substitute the numerator and calculate the final bpa as $m_{12}(L) = 0.4737$.

Condensed DS is an extension of the Bayesian theory. It allows a decision maker to propagate probabilities through logical links and it combines items of evidence by using Dempster's rule as generalization of the Bayes' rule. In practice, it allows to derive beliefs for a question of interest from probabilities for a related demand. In the classical decision theory under uncertainty, an agent has only access to the set S of possible environment states (in the risk case represented as probability distribution $\pi : S \to [0, 1]$). Typify the agent's behaviour through the belief function Bel specifies as $Bel(S) = 1$ and $Bel(X) = 0$ for all $X \subset S$ as *completely vacuous belief function.* Modeling the knowledge of an agent as belief function $Bel : 2^S \to [0, 1]$ in place of $\pi : S \to [0, 1]$ omit the distinction between risk and uncertainty decision scenarios based on the generalization character of the evidence theory.

3.4. Possibility Theory

An agent deals with imprecise knowledge and uncertainty, which concerns the state of knowledge of an agent about the relation between the world and the statement about the world. In a certain case, the agent has full knowledge of the true value of the available data. Uncertainty in a problem scenario emerges

whenever information pertaining to the situation is deficient in some dependency. Uncertainty is partial knowledge and results in ignorance or not knowing situations, where it may be incomplete, imprecise, contradictory, vague, unreliable, or deficient in some other form and way. These various information inadequacies result in different types of uncertainties an agent has to deal with. When an appropriate amount of uncertainty is allowed in dealing with a specific request, the associated complexity may often be substantially reduced while the reliability of the solution obtained is increased. It is an important aspect to have the capability to quantify the uncertainty involved in a problem-solving situation. This requires to measure in an adequate justified way the amount of uncertainty involved in each possible characterization of uncertainty within a mathematical theory. Given a specific measurement unit, the value of uncertainty in each situation should be unique. L.A. Zadeh [Zad78] has introduced in connection with the fuzzy set theory the *possibility theory* to establish a reasoning to be carried out on imprecise or vague knowledge, and making it possible to deal with uncertainties on this knowledge. Only possible situations can be probable. The possibility theory concerns the ability of the situation to occur. In other words, it is necessary to develop a method to formalize non-probabilistic uncertainties of situations. This provides a means of assessing to what extent the occurrence of a situation is possible and to what extent a decision maker is certain regarding its occurrence without knowing the evaluation of the probability of this occurrence. This situation occurs, if no similar or comparable situation is available to refer. The possibility theory can be applied in situations to concern the ability of a proposition to be true. For instance, the question "Will Mary be at the railway station on Monday 25th February?" comprises additional subsequent questions to be taken into account, e.g., what Mary did every previous Monday. The uncertainty within this question is described by the degree of possibility and also the degree of *necessity* as dual or impossibility of the contrary question. The statement "Mary's height is larger than 1.65 meters" implies that any height above 1.65 meters is possible, all other cases are impossible. The possibility value equal to 1 implies, that an event is completely possible, and impossible if it is equal to zero.
Possibility allows the quantification of uncertainty through a possibility measure Π determined by a possibility distribution function

$$\pi : \Omega \rightarrow [0, 1]$$

with a possibility measure

$$\Pi : 2^{\Omega} \rightarrow [0, 1], \;\; \pi(x) = \Pi(\{X\}) \;\; \forall x \in \Omega.$$

The possibility measure Π is defined on the universe of discourse Ω with $\Pi(A)$ for $A \subseteq \Omega$ being the degree of possibility that A is true. An important axiom corresponds to the additivity axiom of probability concerning the possibility $\Pi(A \cup B)$ of the disjunction of two propositions A and B as the maximum of

the possibility of the individual propositions $\Pi(A)$ and $\Pi(B)$

$$\Pi(A \cup B) = max(\Pi(A), \Pi(B)).$$

It follows with $\Pi(\Omega) = 1$ and $\Pi(\emptyset) = 0$ that the possibility measure on finite set is determined as

$$\Pi(A) = max_{x \in A}\, \pi(x) \;\; \forall A \in \Omega.$$

The necessity of a proposition is the negation of the possibility of its negation case. Define the necessity measure $Nec(A)$ to a given proposition A by $Nec(A) = 1 - \Pi(\neg A)$. In that case, express the intersection of $Nec(A)$ and $Nec(B)$ as

$$Nec(A \cap B) = min(Nec(A), Nec(B)).$$

It follows for the possibility intersection operator and necessity union operator

$$\Pi(A \cap B) \leq min(\Pi(A), \Pi(B))$$

and

$$Nec(A \cup B) \geq max(Nec(A), Nec(B)).$$

Possibility and necessity are dual, so that the occurrence of an event is certain, if and only if the occurrence of its complement is impossible. $\Pi(A)$ measures the degree to which the event A is likely to happen, whereas $N(A)$ indicates the degree of certainty which can be assigned to this occurrence.

Example 3.10. Considering the distinction of possibility measure versus probability measure the number of eggs X that Mary eats at breakfast and is going to order for tomorrow morning (see Figure 3.2). Consider the case where Mary

n	1	2	3	4	5	6	7
$p(n)$	0.1	0.8	0.1	0	0	0	0
$\pi(n)$	1	1	1	0.8	0.6	0.4	0.2

Figure 3.2.: Possibility and probability distributions associated with X

eats 3 eggs. The possibility value is equal to one, the probability is quite small, e.g., 0.1. A high degree of possibility does not imply a high degree of probability, nor in the opposite case. If an event is impossible, it is bound to be improbable. This heuristic connection may be called consistency principle.

From the epistemic point of view, possibility is related to our state of knowledge and must not deal with actual abilities independently of our knowledge about them. Epistemic possibility says something about a statement and our knowledge about the actual world. A statement is epistemically possible if it

may be true for all we know and impossible, if it can't be true given what we know. In the exact same manner a statement is epistemically necessary if it is *certain* given what we know.
Summarized possibility and probability are prescriptions to represent uncertainties. Possibility measures reflect vague, but coherent knowledge, whereas probability measures summarize more precise and varying knowledge. The relation between probability and possibility can be described in the subjective context. Subjective properties of uncertainty are linked to the individual opinion about the true value of the data as derived from the available information. Possibility and necessity are the epistemic properties that reflect the individual opinion about the truth statement. Only possible statements can be believed.

3.5. Fuzzy Logic

Fuzzy logic is derived from fuzzy set theory dealing with approximate reasoning for the quantification of imprecision. It provides an approximate and effective means of describing the behaviour of linguistic declarations, complex systems and applications, that are incompletely defined, not easy described precisely, or not easy analyzed mathematically. Fuzzy logic simplifies the way human beings reason in supporting graded statements rather than ones that are strictly true or false and therefore less error-prone to noisy information sources. Fuzzy logic techniques have been successfully applied in a number of application fields like medical diagnosis, controller systems or decision making systems. Fuzzy logic controllers are capable of making intelligent control decisions in often volatile and rapidly changing problem environments.
The following example characterizes the need for a modified understanding in defining a set as collection of objects. If the age of a man is close to 25 and we declare this man as being *young*, the question is, it is more or less correct for a man with the age of 22 and another man with the age of 28. The decidability of *being young* is decreasing with this kind of information pieces. Applying the classical set theory causes a sharp age value for describing the exact meaning of the linguistic expression *young*. The assignment of fuzzy logic allows dealing with degrees of truth represented as membership in vaguely defined sets. L.A. Zadeh has introduced 1965[Zad65] the idea of non-crisp sets as fuzzy sets. Set membership values range between zero and one as real number related to the use of vague predicates like "James is young" or "Mary is tall". For instance, the predicate *tall* is vague, imprecise and not clear defined. But humans have an idea what the expression means and agree that there occurs no sharp cutoff between *tall* and *not tall* without using a folding rule. For a fuzzy set *tall* we may define a degree of membership 0.8, if Mary stands 1.72 meters high. Belonging to a set admits here a degree that is not necessarily just one or zero as in the classical set theory case. Dealing with linguistic forms and declarations, it is often impossible to say that some elements of the universe of discourse belong to the set or not. In probability, define a scalar variable for the example *tall* and also describe a conditional distribution for the tall level. Partial membership in a set can express as degree of membership $\mu_A(x)$ of the element x to a fuzzy

set A. Zadeh replaces the range in the classical indicator function ($I_A(x)$ with 1, if $x \in A$ and 0, if $x \notin A$) from $\{0, 1\}$ to the real-valued interval $[0, 1]$ as generalization. In the classical case, a subset A of a set X can be defined as mapping from the elements of X to the elements of $\{0, 1\}$ to represent truth or falsity of a statement. A fuzzy set extends the binary membership to a spectrum in the interval of $[0, 1]$. All elements of the universal set X are members of a given set A and for each element $x \in X$ $0 \leq \mu_A \leq 1$ hold. The mathematical fuzzy set theory generalizes the concept of sets and can be used wherever sets can be used, and therefore is not restricted to any particular form of imperfect data. One domain of application is the modeling of imprecision and vagueness. Fuzziness creates an order among the possible values in which the actual value is known to belong. New concepts like fuzzy probability (likely), fuzzy quantifiers (most), fuzzy predicated (small), or the impact of linguistic hedges (very) are explained in detail in [And96]. The fuzzy logic approach is able to distinguish between ambiguity of the knowledge and uncertainty in generating errors and lack of complete knowledge. Compared to probabilistic reasoning the fuzzy set approach concerns the belonging of a well defined individual to an imprecisely defined set. Probability deals with uncertainty, probabilistic reasoning offers a mechanism to evaluate the outcome of a system concerned by probabilistic uncertainty and using conditioning to update the probability values and perform probabilistic inference. The advantages of fuzzy logic come into operation, when the evidence is uncertain and it is unusual for human reasoning to follow the probability axioms. A tolerance to imprecision is given, however, fuzzy logic is not useful for the development of systems where we have not enough *a priori* knowledge about the system. Next outline the relation between fuzzy sets and possibility theory. Zadeh has introduced both concepts and proposed an idea how to interpret a membership function in some cases as a possibility distribution. The possibility theory has been widely used in real-world scenarios of modeling, control and decision making for which the measurements don't allow precise structured information or in other cases the information is obtained from a human expert. Possibility allows therefore both, the modeling of imprecision and qualitative characterization of uncertainty. It is essential to differentiate between imprecise and uncertain information. Imprecision belongs to the context of the considered information and uncertainty belongs to the truth level at which it corresponds to reality. The representation of imprecise information can be combined with methods of qualitative description of imprecision, when possibility distributions are built on fuzzy sets [DP93]. For instance, $\mu_{young}(y)$ quantifies the membership of a person with age y to the set of *young* men and $\pi_{young}(y)$ quantifies the possibility that the age of a person belongs to the set of *young* men. Explain this relationship as equality $\mu_{young}(y) = \pi_{young}(y)$ for all $y \in Y$, where Y is the set of age. Possibility distribution functions can therefore be interpreted as a membership function of the fuzzy set. The degree of truth of a proposition "Peter is young" knowing that Peter's age is y is equated numerically to the grade of membership of a person with age y to the set of *young* men and therefore to the possibility that the age of a *young* person is y.

The strengths of fuzzy sets are the plausibility of fuzzy sets, their easy in-

terpretability and plausibility and ability for approximate reasoning. We know from human reasoning, that it is approximate in most cases and involves different uncertainties. As an example remember the vague predicate *tall*. Given that Mary is tall and Evelyn is a little bit shorter than Mary leads to the conclusion that Evelyn is tall as well. For the development of a production machine dealing with such propositions we need a set of rules and an inference engine for the interpretation of the rules. The inference engine generates new inferences as reasoning mechanism. Different forms of fuzzy reasoning with single and multiple antecedent clauses will be discussed in Chapter 6.

3.6. Uncertainty Management Review

Uncertainty management plays a critical role in the development of knowledge-based systems. Human tasks require intelligent behaviour with some degree of uncertainty. A knowledge-based system exhibits such intelligent behaviour by modeling the empirical associations and heuristic relationships that (domain) experts have built over time. Types of uncertainty that can occur may be caused by different problems with the data, which might be missing, imprecise, inconsistent or unavailable. Uncertainty may also be caused by the represented knowledge since it might not be appropriate in all situations. This chapter has presented different numerically oriented methods which have been developed to deal with uncertainty. They answer different questions such as how to represent, how to combine pieces or how to draw inference using uncertain data. The probability theory is the most mature of all uncertainty reasoning methods based on the mathematical and theoretical foundation. The probability approach requires a significant amount of probability data to construct a knowledge base. Certainty factor values relying on values obtained from domain experts are able to compensate for this disadvantage. Dempster-Shafer theory of evidence is able to represent a decision maker's certainty about certainty. In other words, the commitment to a belief in some fact does not imply that the remaining belief holds for the fact's complement. A decision maker has the advantage to express and compute the degree of ignorance concerning to some fact. The major difficulty with Dempster-Shafer is its complexity requiring an exhaustive enumeration of all possible subsets in the frame of discernment in nearly all cases. Additionally, it offers no guidance on how the mass probability assignments should be calculated or how a decision maker takes a decision from the result. Fuzzy methods have been applied across a range of domains including medical diagnosis, process control or fault detection. Fuzzy logic is able to deal with numerous vague and imprecise concepts and terms which are used, but the development of the membership functions is nontrivial and needs expert guidance. The selected method which is able to mimic the heuristic knowledge of a problem domain will likely depend on the nature and complexity of the problem domain, the nature of that uncertainty, and the amount of data available to support the uncertainty probabilities. In the case of probability theory it is desirable to install a graphical component to facilitate causal relationship understanding. Competing goals exist when building the graphical structure. On

the one hand, we would like to minimize the number of parameters to make the probability elicitation task easier and to simplify belief updating. Otherwise, we would like to maximize the feasibility of the model, which sometimes requires more nodes, arcs, and states. A tradeoff must be made between building a more accurate graphical model and the cost of additional modeling.

Part II.

Graphical Models

4. Bayesian Networks

Bayesian networks have become increasingly popular for reasoning under uncertainty practiced in many fields of application such as diagnosis, prediction, decision making, or data mining [HE04]. Many real-life situations can be modeled as a domain of entities represented as random variables in a network structure [FH00]. A Bayesian network is a proper graphical representation of dependence and independence relations between such random variables. This chapter introduces the key concepts of (conditional) dependence and independence in a network structure, the different varieties of Bayesian networks, methods for making inference, algorithms for learning network structures from data, aggregation methods to gather and fuse models from different and distributed sources and a hierarchical specification of such networks allowing a knowledge engineer to work on different levels of abstraction based on the underlying object-oriented programming paradigm. We have conceptualized and developed a new structure learning algorithm named LAGD, which offers a new class of parameterized algorithms. In the meanwhile, the algorithmic implementation is integrated in the well-known WEKA environment. The LAGD derivation and concept description is also given in this chapter.

4.1. Introduction

Knowledge representation applications need a powerful instrument as formal graphical language requiring reasoning under uncertainty [LGS07]. Bayesian networks (BN) are graphical models to represent knowledge under conditions of uncertainty, where nodes represent discrete or continuous variables and arcs represent direct connections between them. BNs model the quantitative strength of the connections between variables allowing probabilistic beliefs about them to be updated automatically as new information becomes available. Contending views about how to understand probability derive from the physical and Bayesian interpretation. Physical probability is referred to some definite physical process, whereas the traditional alternative is to think of probabilities as reporting our subjective degrees of belief. Thomas Bayes [Bay58] has expressed this view as more general account of probability in that we have subjective belief in a huge variety of propositions. The Bayes' Theorem (see equation 3.8) serves as basis of the probability calculus, which is applied in BNs signifying graphical structures that allow to represent uncertain domains and to reason about them. BNs have been used successfully in many fields like logistic applications ([Hol03] [Hol04a]), expert systems [Kle92] or classification systems as powerful tools for the knowledge representation and inference under uncertainty. A reason for this broad field of applications is the notion of modularity, where a complex system is built by combining simpler parts. Probability theory ensures that the system

as a whole is consistent, and provides ways to interface models to data. The graph theoretic side provides an intuitively appealing interface by which knowledge engineers can model highly interacting sets of variables. The following Figure 4.1 motivates the use of probabilistic graphical models in which nodes represent random variables, and arcs represent conditional dependence assumptions. Consider a BN for diagnosing faults in a car [PW02] like VW Golf IV.

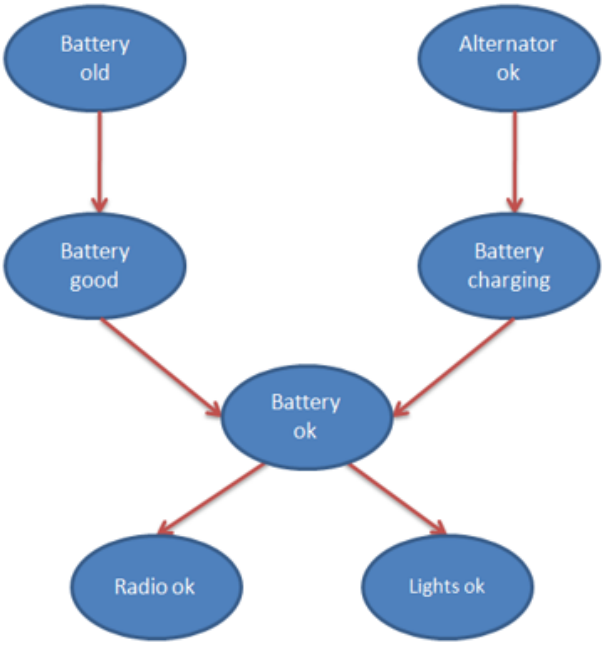


Figure 4.1.: Diagnosing faults in a car

Figure 4.1 represents the fact that the age of the battery, which is represented by the node *Battery old*, has a probabilistic influence on how good the battery status is. This in turn has an influence on whether the battery is operational (node *Battery ok*), which is also affected by whether the alternator is working (node *Alternator ok*) and, as a result, whether the battery is recharged (node *Battery charging*) when the Golf IV moves. The operational state of the battery (node *Battery ok*) affects whether the radio and lights will work (nodes *Radio ok* and *Lights ok*). It is expected in this BN that the observations that can be carried out are those relating to the lights, the radio and possibly the age of the battery. The result of these observations can be propagated through the network to establish the probability of the alternator being in the status okay and the battery being good. These latter variables are the ones a car mechanic employee is interested in since they relate to fixing the car.
BNs and the use of probabilistic models is based on directed acyclic graphs (DAG) with a probability table associated with each node. The nodes in a Bayesian network represent propositional variables in a domain, the edges between the nodes represent the dependency relationship among the variables. Assuming discrete variables, the strength of the relationship between variables is quantified by conditional probability distributions associated with the nodes.

Each node has a conditional probability table (CPT) $P(X|X_1,\ldots,X_n)$ attached that quantifies the effects that the parents $X_1,\ldots,X_n$ have on the node. We can say that the conditional probabilities encode the strength of dependencies among the variables. For each variable a conditional probability distribution is defined that specifies the probability of node X being in a certain state given the values of the parents of X.
A decision maker makes decisions by combining his own knowledge, experience and intuition with that available from other sources. Given a learned network structure like BNs the decision maker can derive additional information by applying an inference algorithm. Use the learned BN to calculate new probabilities when particular information is achieved. For instance let A have n states with $P(A) = (x_1, ..., x_n)$ and assume that we get the information e that A can only be in state i or j. This statement expresses that all states except for i and j are impossible, so next illustrate the probability distribution as $P(A, e) = (0, ..., 0, x_i, 0, ..., 0, x_j, 0, ..., 0)$ [Jen01]. Note that $P(e)$ is the sum of $P(a, e)$. Assume a joint probability table $P(U)$ where e is the preceding finding (n-dimensional table of zeros and ones). Using the chain rule for a BN [RN03] over the universe U and let e_i be findings express the following

$$P(U, e) = \prod_{A \in U} P(A|pa(A)) \cdot \sum_i e_i$$

and for all $A \in U$ we have

$$P(A, e) = \frac{\sum_{U\setminus\{A\}} P(U, e)}{P(e)}.$$

A knowledge engineer must undertake different steps when building a BN. First have a closer look at the following medical diagnosis problem [KN04].

Example 4.1. Lung cancer: A forty-some years old man visits his doctor suffering from shortness of breath called dyspnoea. The doctor knows that different diseases such as tuberculosis, bronchitis, and lung cancer are possible causes. The doctor also knows that other relevant information should receive attention, whether or not the patient is a smoker and what sort of air pollution he has been exposed to. A positive X-ray can indicate either tuberculosis or lung cancer. The question is the calculation of the probability of the lung cancer disease.

A knowledge engineer has to identify the variables of interest for the building of the structure of the appropriate BN. This involves answering the question to identify the nodes to represent and what values they can take. Consider only nodes that take discrete values, which should be both mutually exclusive and exhaustive, which means that the variable must take on exactly one of these values at a time. Common types of discrete nodes include boolean nodes (e.g. represent proposition *has cancer* by taking binary values), ordered nodes (e.g. represent *patient's pollution exposure* with the values *low, medium, high*), and integer values (e.g. represent *age of a patient*) in an interval from 1 to 100. It is important to choose values that represent the domain efficiently with enough

level of detail.

Table 4.1.: Preliminary choices of nodes and values for lung cancer

Node name	Type	Values
Pollution	Binary	low,high
Smoker	Boolean	T,F
Cancer	Boolean	T,F
Dyspnoea	Boolean	T,F
X-ray	Binary	pos,neg

The structure of the network should capture qualitative relationships between the variables. Two nodes are connected directly if one affects or causes the other, with the edge indicating the direction of effect. In the introduced lung cancer example we might ask what factors affect the patient's chance of having cancer? If pollution and smoking are the answer, then two edges from *Pollution* and *Smoker* to *Cancer* should be added to the network structure (see Figure 4.2). If the topology of a BN is specified, it is essential to quantify the relationships between the connected nodes by specifying a conditional probability distribution for each node. For each node look at all the possible combinations of values of those parent nodes. Each of such combination is an *instantiation* of the parent set. For each distinct instantiation of parent node values specify the probability that the child will take each of its values. Consider the *Cancer*

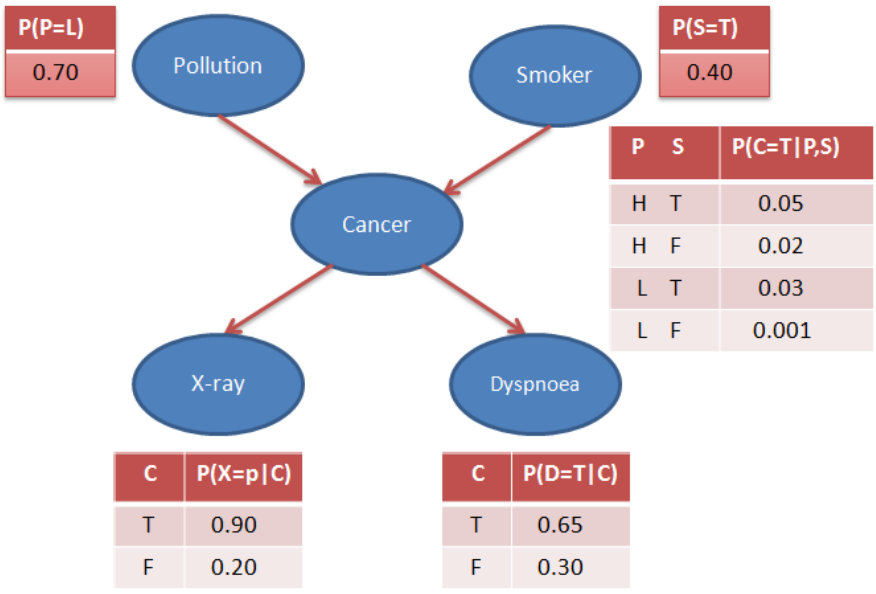


Figure 4.2.: Typical Bayesian network for the lung cancer scenario

node in Figure 4.2. Its parent nodes *Pollution* and *Smoking* take the possible joint values $\{< H, T >, < H, F >, < L, T >, < L, F >\}$. The CPT specifies the probability of cancer for each of these cases to be $< 0.05, 0.02, 0.03, 0.001 >$.

These probabilities must sum to the value *one* over all possible states of the *Cancer* variable, so that the probability of the opposite case *no cancer* is implicitly given as $< 0.95, 0.98, 0.97, 0.999 >$. Calculate the probability of the event that the disease is lung cancer but neither pollution nor smoking patient is existent, and both X-ray and dyspnoea is positive. Use single letter names for the variables:

$$\begin{aligned} &P(\neg P \wedge \neg S \wedge C \wedge X \wedge D) \\ &\quad = P(\neg P)P(\neg S)P(C|\neg P \wedge \neg S)P(X|C)P(D|C) \\ &\quad = 0.70 \times 0.60 \times 0.001 \times 0.90 \times 0.65 = 0.0002457 \ (\sim 0.025\%). \end{aligned}$$

The calculation of the probability of the discussed event is based on the BN property in providing a complete description of the problem domain. A generic entry in the joint probability distribution is the probability of a conjunction of particular assignments to each variable given by the formula

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} P(X_i|Parents(X_i)). \tag{4.1}$$

Equation 4.1 implies certain conditional independence relationships that can be used efficiently to guide a knowledge engineer in constructing the network topology. The specification of the joint probability distribution is equivalent to the general assertion that

$$P(X_i|X_{i-1}, ..., X_1) = P(X_i|Parents(X_i)) \tag{4.2}$$

provided that $Parents(X_i) \subseteq \{x_{i-1}, ..., x_1\}$. The last condition allows to construct a network from a given ordering of nodes using an algorithm for constructing BNs ([RN03],[Pea88]). The construction method processes each node in order, adding it to the existing network and adding arcs from a minimal set of parents such that the parent set renders the current node conditionally independent of every other node preceding it. The incremental network construction is in detail as follows:

1. Choose the set of relevant variables X_i that describe the problem domain D.

2. Choose an ordering for the variables $X_1, ..., X_n$.

3. For $i = 1$ to n
   (a) Pick the next variable X_i and add a node to the network.
   (b) Set $Parents(X_i)$ to some minimal set of nodes already in the net such that the conditional independence property (4.2) is satisfied.
   (c) Define the CPT for X_i.

The acyclic network property is fulfilled because each node is only connected to earlier nodes. The knowledge engineer creates a BN without redundant probability values, except perhaps for one entry in each row of each conditional probability table. It is obvious that the construction method with a different

node order may result in a different network structure even so representing the same joint probability distribution. There are domains in which each variable can be influenced directly by all the others, so that the network is then fully connected. Then specifying the CPTs requires the same amount of information as specifying the joint. In a locally structured domain constructing a BN is to rise to a challenge. It is required that each variable is directly influenced by a few others and that the network topology reflects those direct influences with the appropriate set of parents. The correct order to add nodes is to add root causes at first and then the variables they influence until reaching the leave nodes. The question now is what happens if the wrong order is chosen? Consider the cancer BN scenario and suppose we decide to add the nodes in the order $Dyspnoea, X-ray, Cancer, Pollution, Smoker$. $Dyspnoea$ is the root cause node having no parents. When adding $X-ray$ it is crucial to consider if $X-ray$ is independent of $Dyspnoea$? Since they have a common cause in $Cancer$, they will be dependent, exemplify as $P(X|D) \neq P(X)$. When adding the next nodes, the knowledge engineer must respond to the conditional independence request as illustrated in Table 4.2. The resulting network has two

Table 4.2.: Conditional independence given the node order $< D, X, C, P, S >$

Conditional independence	Satisfied
$P(X\|D) = P(X)$?	No
$P(C\|X, D) = P(C\|X)$? $P(C\|X, D) = P(C)$?	No
$P(P\|C, X, D) = P(P\|C)$?	Yes
$P(P\|C, X, D) = P(P)$?	No
$P(S\|P, C, X, D) = P(S\|C)$?	No
$P(S\|P, C, X, D) = P(S\|C, P)$?	Yes

additional arcs and three new probability values associated with them. It is generally desirable to build the most compact BN possible in reducing probability values requiring specification and in representing independencies explicitly or representing the causal dependencies in the considered problem domain.
To design inference algorithms it is essential to identify conditional independencies given a BN. Is a set of nodes X independent of another set Y given a set of evidence nodes E? As introduced in Section 3.2.5 a variable X is independent of another variable Y with respect to a probability distribution P if $P(x|y) = P(x), \forall x \in dom(X), \forall y \in dom(Y)$. It is feasible to express this property symbolically as $X \perp Y$. A variable X is conditionally independent of Y given Z with respect to a probability distribution P if $P(x|y, z) = P(x|z), \forall x \in dom(X), \forall y \in dom(Y), \forall z \in dom(Z)$ (symbolically $X \perp Y|Z$).
Let X, Y and Z be three variables for which $X \perp Y|Z$ is essential. Following the ordering (X, Y, Z) this can be simplified to

$$P(X, Y, Z) = P(X|Z)P(Y|Z)P(Z).$$

Similarly, following the ordering (X, Z, Y) we get

$$P(X, Y, Z) = P(X|Z)P(Z|Y)P(Y)$$

and following the ordering (Y, Z, X) we get

$$P(X, Y, Z) = P(Y|Z)P(Z|X)P(X).$$

Consider three significant scenarios *causal chain*, *common cause* and *common*

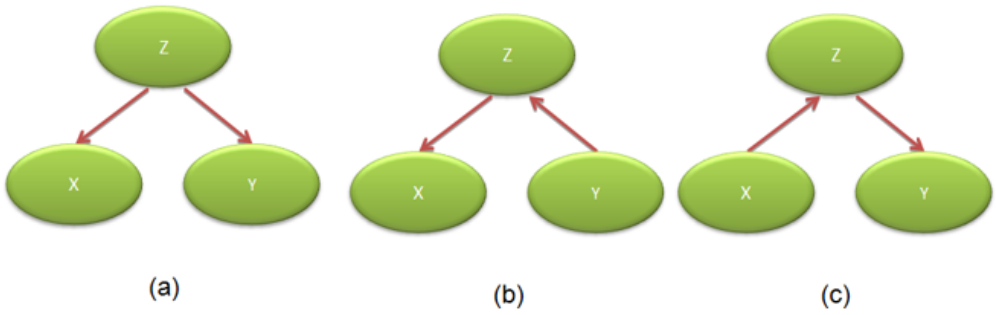


Figure 4.3.: Conditional independence with graphical representations of $X \perp Y|Z$

effect. Causal chains (see Figure 4.3 (b)-(c)) give rise to conditional independence. This means, that the probability of X given Z in (b) is exactly the same as the probability of X given both Z and Y. Knowing that Y has occurred does not make any difference to the belief about X if we already know that Z has occurred. In the *common cause* case (a) X and Y have a common cause Z, e.g. if a disease like cancer is a common cause of the two symptoms of a positive X-ray and dyspnoea. If a doctor has no evidential information about a specific disease, then learning that one symptom is present will increase the chances of this disease which in turn will increase the probability of the other symptom. Common effects are represented as *v-structure*. An effect node has two causes. If the doctor observes the effect of a disease and then finds out that one of the causes is absent, then this raises the probability of the other cause [WS06].
How BNs represent conditional independence and how these independencies affect belief change during an updating step is declared. If a BN is given, is it possible to identify whether a set of nodes X is independent of another set Y given evidence nodes E? *Direction dependent separation* or *d-separation* can solve this occurring problem. Given $X \perp Y|Z$, follow that knowing the value of Z *blocks* information about Y being relevant to X and vice versa. If every undirected path as path through the BN regardless of the direction of the arcs from a node in X to a node in Y is d-separated by E, then X and Y are conditionally independent given E. A path is *blocked* given a set of nodes E if there is a node Z on the path for which one of three conditions holds [RN03]:

1. Z is in E and Z has one arc on the path leading in and one arc out.
2. Z is in E and Z has both path arcs leading out.

3. Neither Z nor any descendent of Z is in E, and both path arcs lead into Z.

Recapitulating a set of nodes E d-separates two other sets of nodes X and Y if every path from a node in X to a node in Y is blocked given E.
Consider the introduced lung cancer scenario of Figure 4.2. With the blocking condition *Pollution* is d-separated from $X-ray$ and *Dyspnoea*. In the same way, *Smoker* is d-separated from $X-ray$ and *Dyspnoea*. In the common cause $X-ray$ is d-separated from *Dyspnoea*. If *Cancer* has not been observed, then *Smoker* is d-separated from *Pollution* in the common effect case.

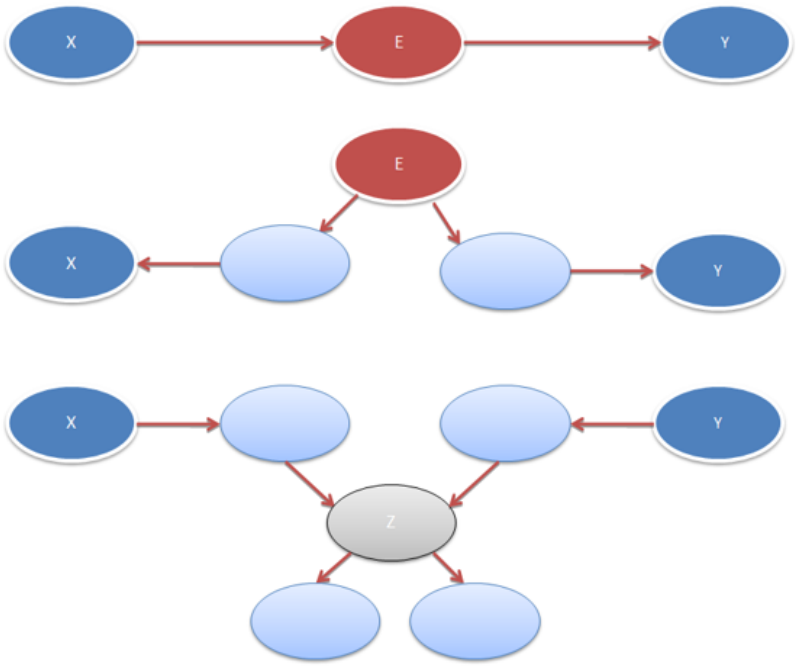


Figure 4.4.: Types of situations in which a path from X to Y can be blocked, given the evidence E

This introduction shows tellingly, that graphical structures provide a powerful instrument for expressing and reasoning about causal relationships among variables. The ability to perform inter-causal reasoning between causes of a common effect is unique for such models. The qualitative reasoning advantage lies in the fact that a DAG is a very compact representation of dependence and independence statements among a set of variables. Since the concept of causality though debatable from a philosophical point of view appears to be quite natural to human experts, BNs are indeed very attractive from a knowledge engineering point of view. The aforementioned independence relations are utilized in order to make probabilistic calculations more efficient. Particularly, the joint distribution of all variables in $\mathcal{V}$ is given by

$$\prod_{V \in \mathcal{V}} \rho_V(V|pa(V)) \tag{4.3}$$

where $pa(V)$ denotes the set of parents of the variable V. Note that the complete number of probabilities to be specified is now given by

$$\sum_{V \in \mathcal{V}} \prod_{W \in pa(V)} card(\mathfrak{D}_W), \tag{4.4}$$

which is generally much smaller than the size

$$\prod_{V \in \mathcal{V}} card(\mathfrak{D}_V) \tag{4.5}$$

of the complete joint distribution. It is clear, that the sum of each variable $V \in \mathcal{V}$ (4.4) might still be prohibitively large.
DAGs are also very intuitive and intelligible maps of causal and correlation interactions, and thus provide a powerful instrument for conceptualizing, formulating, communicating, and discussing qualitative interaction models in problem domains where causal or correlation mechanisms are at least partly known and also in domains where such mechanisms are really unknown but can be revealed through learning techniques of model structures from data (see also Section 4.3). The concept of probabilistic independence and advertise is seen, whenever a probabilistic model is specified in terms of a product of lower-dimensional conditional distributions. Probabilistic graphical models are almost descriptions of local causal phenomena, where a domain expert provides assessment of cause-effect relations and their associated conditional probability distributions standing for the strengths of the relations. From a philosophic point of view, differentiate the Bayesian and frequentist view. The Bayesian view of probabilities considers probabilities as expressing subjective assessments of belief rather than objective measures of frequencies of events. The use of mechanisms of causality on the one hand and subjective assessment of probabilities on the other hand to express strengths of causal relationships provides a strong paradigm for formulating models for reasoning under uncertainty. In Chapter 5 it is explained in detail, how such models can be augmented with decision variables and utility functions for specific representation and solution of sequential decision problems.

4.2. Inference in Bayesian Networks

After the BN representation of a knowledge domain and its uncertainty is given, we can discuss about different types of reasoning. The question is how to use a BN to reason about the domain. When observing the value of some variable, we would like to condition upon the new information [KN04]. The conditioning or belief updating process is performed as information flow through the network. This information flow is not limited to the direction of the arcs. This becomes the task of computing the posterior probability distribution for a set of query nodes given values for some evidence or observation nodes. In the case of probabilistic networks, there is a clear distinction between the underlying knowledge base and the inference engine. The knowledge base is the

BN, whereas the inference engine is a set of generic methods that applies the knowledge formulated in the knowledge base on task specific data sets known as evidence variables to compute solutions to queries against the knowledge base. It is therefore fundamental to compute the posterior probability distribution for a set of query nodes given values for some evidence nodes. The *belief updating* process is flexible, because evidence can be entered about any node while beliefs in any other nodes are updated. BNs can be conditioned upon any

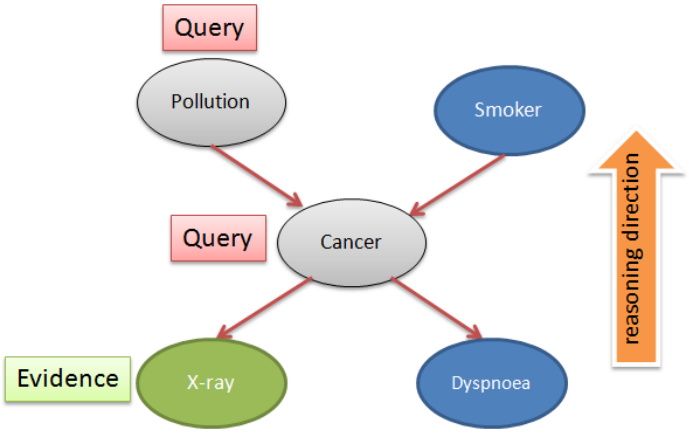


Figure 4.5.: Diagnostic reasoning in a Bayesian network

subset of their variables supporting any directions of reasoning based on the comprehensive representation of probability distributions over the variables of a BN. In diagnostic reasoning a doctor reasons from symptoms to causes, when he observes *X-ray* and then updates his belief about *Cancer* and whether the patient is a smoker (see Figure 4.5).
Predictive reasoning traces the opposite direction from new information about causes to new beliefs about effects in following the direction of the network arcs.

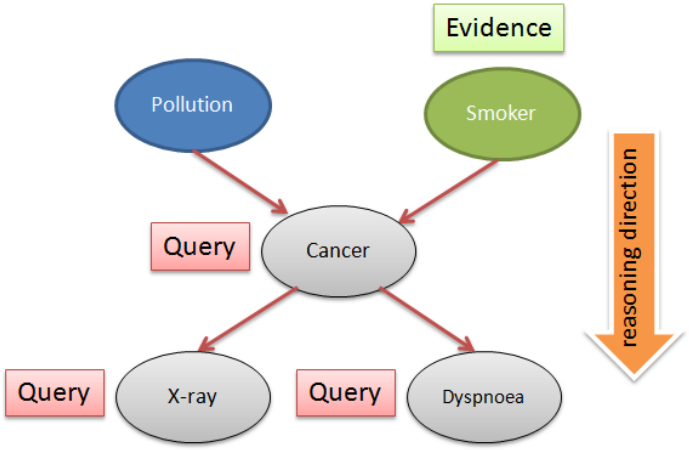


Figure 4.6.: Predictive reasoning in a Bayesian network

The patient informs the doctor about his current smoker characteristic even

before any symptoms have been assessed. This information piece changes the doctor's expectations that the patient might have cancer or a positive *X-ray* result (see Figure 4.6).
Another form of reasoning involves reasoning about the mutual causes of common effects as *inter-causal* reasoning. There are two or more possible causes of a particular effect represented as *v-structure* in a BN. Initially, these causes are independent of each other, but with knowledge of the effect the presence of one explanatory cause renders an alternative cause less likely. Combine the introduced reasoning types in different varieties of reasoning as illustrated using the cancer BN scenario. Besides calculating beliefs in query variables given definite values for evidence variables, BNs can also be used for a decision making process based on probabilities in the network and on agent's utilities (see Chapter 5) or for performing *sensitivity analysis* to understand which aspects of the BN have the greatest impact on the probabilities of the query variables. It is also fundamental to explain, describe, and illustrate the results of probabilistic inference to the system user applying probabilistic mechanisms.
Different inference algorithms are suited for different network structures and performance requirements. In BNs where nodes are connected by multiple paths inference algorithms become very complex, where exact inference mechanisms become computationally complex and in which approximate inference must be used. Under performance circumstances, the speed of inference depends on the structure of the network including the connection degree and the locations of evidence and query nodes. In general, (approximate) probabilistic inference is NP-hard [DL93]. For certain classes of BNs the complexity of probabilistic inference is polynomial or even linear in the number of variables in the network. The complexity is polynomial, if the graph of the BN is a *polytree* and linear in the case of a *tree*.
It is possible to derive an algorithm that works efficiently on single connected networks also known as polytrees. They have at most one undirected path between any two nodes in the network [RN03]. Assume X is a query node, and there is some set of evidence nodes E excluding X. The aim is to compute $P(X|E)$ and update $Bel(X)$. Denote $Bel(X)$ in a two node network $W \rightarrow X$ with evidence $W = w$ as posterior probability or belief of X, which can be extracted from the CPT as $P(X|W = w)$. In the case of evidence about the child node $X = x$, the inference task of updating the belief for W is done by using the Bayes Theorem 3.8 obtaining $Bel(W = w) = \alpha P(w)\lambda(w)$ with α as normalization constant, $P(w)$ as prior and $\lambda(w) = P(X = x|W = w)$ as likelihood.
Figure 4.7 shows a polytree and how local belief updating of node X is achieved through incorporation of evidence through its parents U_i and children Y_j. Evidence can be divided into predictive (from evidence nodes connected to X through its parents U_i as E_X^+) and diagnostic (from evidence nodes connected to X through its children Y_j as E_X^-) support for X. X d-separates E_X^+ from E_X^- in the network, so use conditional independence to simplify the calculation of $P(X|E) = \alpha P(E_X^-|X)P(X|E_X^+)$. The derivation of the major steps to compute $P(X|E)$ can be found in detail in [RN03], [Pea88], [Nea04].
More general, the BN structure is a DAG, in which two nodes are connected by

more than one path. Multiply connected networks occur when some variable can influence another through more than one causal mechanism. There are three basic classes of algorithms for evaluating multiply connected networks. The first *clustering* inference algorithm transforms the polytree into a probabilistically equivalent polytree by merging nodes, removing multiple paths between two nodes along which evidence may travel. It is possible, that all non-leaf nodes be merged into a single compound node.

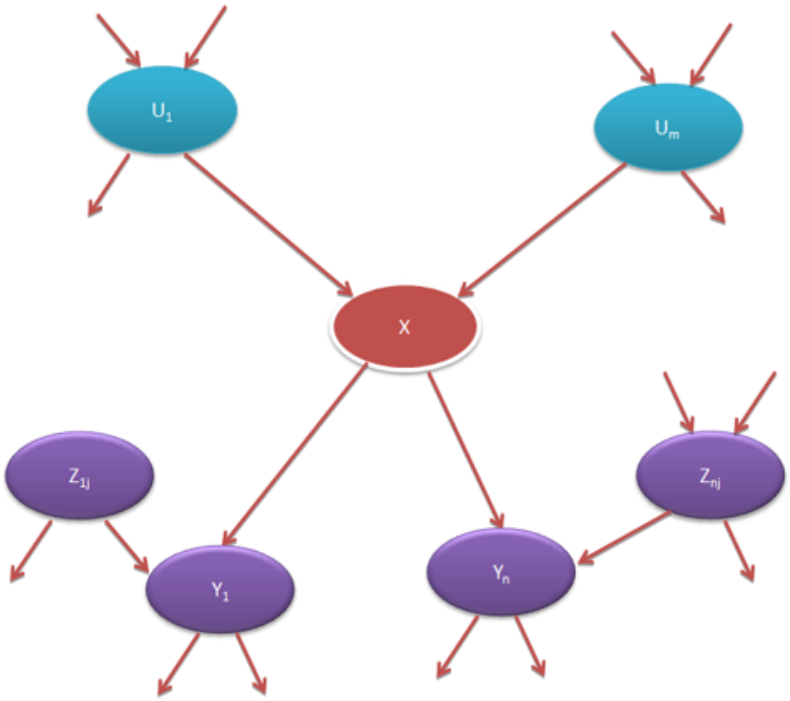


Figure 4.7.: A generic singly connected network partitioned according to the parents and children of the query variable X

It is easier to handle smaller clusters, since the CPT size for the cluster grows exponentially in the number of nodes merged into it. Highly connected original networks enclosed the requirement of rather larger clusters. Another approach of probabilistic inference operates on a structure known as *junction tree* or *join tree*. A junction tree representation is efficient when solving the inference task for multiple sets of different evidence and target variables. A junction tree representation $\mathcal{T}$ of a BN is a pair $\mathcal{T} = (\mathcal{C}, \mathcal{S})$, where $\mathcal{C}$ is the set of cliques and $\mathcal{S}$ is the set of separators [NPP06]. A clique of an undirected graph is defined as a set of nodes that are all pairwise linked. This means, for every pair of nodes in the set there is an arc between them. A maximal clique cannot be increased by adding any node. Each clique $C \in \mathcal{C}$ represents a maximal complete subset of pairwise connected variables of X, $C \subseteq X$, of an undirected graph. The separators $\mathcal{S}$ annotate the links of the tree. The process of creating a junction tree representation of a DAG is beyond the scope of this work and is engrossed in [CDL07]. Details of the inference process can be found in [Jen01].
In practice, for most small to medium sized networks, clustering mechanisms are

good enough. For larger or densely connected networks, approximate algorithms should be used. *Stochastic simulation* uses the network to generate a large number of cases from the network distribution. As more cases are generated, the estimate converges on the exact probability by the law of large numbers from statistics. As with exact inference, there is a computational complexity [DL93], but in practice, if the evidence being conditioned upon is not too unlikely, these approaches converge quickly enough. A popular sampling algorithm is *logic sampling* [Hen88]. The aim is to compute $P'(X|E = e)$ as an estimate of the posterior probability for node X given the evidential information $E = e$. The algorithm generates a case by randomly selecting values for each node, weighted by the probability of that value occurring. The nodes are traversed from the root node down to the leaves to ensure at each step the weighting probability is either the prior or the CPT entry for the sampled parent values. To estimate $P(X|E)$ with a sample value $P'(X|E)$, compute the ratio of cases where both X and E are true to the number of cases where just E is true. After a generation of each case, these combinations are counted as appropriate. For instance, someone will calculate $P(Pollution|X - ray \wedge \neg Cancer)$. With logic sampling, we generate random samples and count the following: N_C as samples with $X - ray$=true and $Cancer$=false, N_S as samples with $Pollution$=true and $X - ray$=true and $Cancer$=false, and N as number of samples. If N was chosen large enough, N_C/N is a good estimation for $P(X-ray = true \wedge Cancer = false)$, and N_S/N is a good estimation for $P(Pollution = true \wedge X-ray = true \wedge Cancer = false)$. As combined, N_S/N_C is a good estimation for $P(Pollution|X-ray \wedge \neg Cancer)$. A modification of logic sampling is the *likelihood weighting algorithm*, which overcomes the problem with unlikely evidence. Instead of adding one to the run count, the CPTs for the evidence node(s) are used to determine how likely that evidence combination is. That fractional likelihood is the number added to the run count [GH02].

4.3. Learning Bayesian Networks

The problem of learning graphical models like Bayesian networks using structure learning algorithms plays a decisive role for the aggregation and fusion of expert knowledge [HHG02]. In the following a new parameterized structure learning approach is presented. A competing fusion mechanism to aggregate expert knowledge stored in distributed knowledge bases or probability distributions is also described. Experimental results of a medical case study show that the approach can improve the quality of the learned graphical model.

The Bayesian learning problem comes in general in four varieties. The structure of the Bayesian network can be known or unknown, and the variables in the network can be observable or hidden. Figure 4.8 illustrates the Bayesian learning varieties. In the simplest case, the structure is known and all variables are observable. The learning part consists of setting the conditional probability tables based on the given structure. These can be estimated directly applying the statistics of the set of examples. In the case of a known network structure with incomplete training sets ($T_i \subset T$) there is an algorithm well-known as

Expected-Maximization algorithm (EM) [Bea03] which assumes missing data or hidden variables. EM is used for finding *maximum likelihood* estimates. For instance, $P(D|H_i)$ represents the probability that a particular data set D would have been observed given H_i as underlying model. With a uniform prior, we can choose a H_i that maximizes $P(D|H_i)$. Inverted, we are interested in using a most probable hypothesis, for clarification, an H_i that maximizes $P(H_i|D)$. This is called *maximum a posterior* or MAP hypothesis. In the case where the network structure is unknown, we have to reconstruct the topology of the network.

Known structure, complete data set -> fill CPT's ⏲ with parameters	**Known structure, incomplete data set** -> only $T_i \subset T$ well known -> Expectation Maximization Method
Unknown structure, complete data set -> determine network structure -> quality measures for specific network structures, heuristic approach	**Unknown structure, incomplete data set** -> determine network structure -> Structured Expectation Maximization Method

Figure 4.8.: Bayesian network learning situations with known or unknown structure and observable or hidden variables

Assume first the case of observable variables. This can be interpreted as a search through the space of structures, guided by the ability of each structure to model the data correctly. Fitting the data to a given structure reduces the problem to the previous fixed structure problem, and the MAP or maximum likelihood value can be used as heuristic for gradient-based hill climbing or simulated annealing search (see also Chapter 4.3.1 and Chapter 4.3.2). When some variables are unobservable, use an approximate solution as extension of the EM algorithm. The *Structural EM algorithm* [Fri98] only does an expensive computation each time the model is changed. The algorithm starts with some initial DAG G by computing the MAP value of $f(G)$ relative to the data.
Condensed, structure learning algorithms for Bayesian networks play a key role when conditional probability information is missing or imperfect. A parameterized approach describe for instance a complete class of algorithms including standard hill climbing [RN03] as special case. Knowledge fusion is an important field of artificial intelligence [JM99], knowledge science and engineering ([Rq03] [HM04]), which can transform and integrate distributed knowledge resources to generate new knowledge representations or visualizations [PH00]. Experimental results of a medical case study concerning a logical alarm reduction mechanism for intensive care patients show that the approach is very applicable for knowledge fusion to improve the efficiency of working in (distributed) groups.

On constructing BNs from data which arise out of an application area use nodes to represent database attributes. Different BN structure learning algorithms have been developed. A good overview demonstrating general approaches to graphical probabilistic model learning from data is introduced by [Hec95], [Kra96] and [BK02]. It is common to distinguish between search and score methods and the dependency analysis approach [Tor03].
In the first case the algorithm views the learning problem as searching for a structure that best fits the data. The methods start as graphical representation without any edges, using some search method to add an edge to the representation. In the next step they can use score methods to compare the new one with the older structure. The main problem to learn BNs using search and scoring methods is the NP-hard complexity. Representative algorithms belonging to the search and scoring method are *polytree* construction algorithms, the *K2* algorithm applying a Bayesian scoring method or the Lam-Bacchus algorithm applying the minimal description length principle. Using the second dependency analysis method is a different approach. These algorithms try to discover the dependencies from the data and next use these dependencies to infer the structure. The approach introduced in the following section belongs to the first family of algorithms. Based on the complexity examination count the number of independent network parameters as

$$|\theta_m| = \sum_i (|X_i| - 1) \cdot |pa(X_i)|$$

with $|pa(X_i)|$ as number of (joint) states of all parent nodes of X_i to obtain a complexity boundary as model complexity.
To determine and measure the complete quality of the underlying network structure an additional measure is necessary to evaluate the *fitness* of the network which calculates how the structure m goes with data set D. The scoring function $f(m, D)$ illustrates the counteract behaviour of model fitness and model complexity:

$$f(m, D) = model\_fitness(m, D) - \beta \cdot model\_complexity(m, D).$$

The only measurement of the fitness as target value for the determination of the network quality would lead to a complete adjunctive structure with the highest complexity which also increases the memory capacity. Another problem is well-known in machine learning as *overfitting*. Overfitting is generally recognized to be a violation of Occam's razor. When the degrees of freedom in parameter selection exceed the information content of the data, this leads to arbitrariness in the final (fitted) model parameters which reduces or destroys the ability of the model to generalize beyond the fitting data.
The revelation of conditional independence relationships plays a fundamental role within the medical diagnostic and information processing. When for instance learning network structures from application data, apply information theoretic measures to detect conditional independence relations and afterwards use the well known *d-separation* concept [Pea88] to infer the structures of net-

works [Jen01]. Measure the volume of the information flow between two nodes to conclude whether a group of valves corresponding to a condition set can reduce and eventually block the information flow. In BNs information can be determined [Arn01] about the value of a node knowing the value of the other node when both nodes are dependent. The mutual information between two nodes can therefore provide information in the case of two nodes dependency. The degree of their relationship is also important. The mutual information of two nodes X_i and X_j is defined as

$$Inf(X_i, X_j) = \sum_{x_i, x_j} P(x_i, x_j) log_2 \frac{P(x_i, x_j)}{P(x_i)P(x_j)}$$

and the conditional mutual information is defined as

$$Inf(X_i, X_j|C) = \sum_{x_i, x_j, c} P(x_i, x_j, c) log_2 \frac{P(x_i, x_j|c)}{P(x_i|c)P(x_j|c)}.$$

Based on model complexity and evaluation measures built up parameterized scoring functions as follows:

$$\begin{aligned} f_1(m, D) = \\ (\sum_i \sum_{x_i \in dom(X_i)} \sum_{Y_j \in pa(X_i)} \sum_{y_j \in dom(Y_j)} P(X_i = x_i, Y_j = y_j) \\ log_2 \frac{P(X_i = x_i, Y_j = y_j)}{P(X_i = x_i) \cdot P(Y_j = y_j)}) - \beta \cdot \sum_i (|X_i| - 1) \cdot \prod_{X \in pa(x_i)} |X|. \end{aligned}$$

This function takes the following form applying maximum likelihood:

$$f_2(m, D) = (N \cdot \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \frac{N_{ijk}}{N} log_2 \frac{N_{ijk}}{N_{ij}}) - \beta \cdot |\theta_m|$$

where m represents the network structure, D the training data, N the total number of cases, $r_i (1 \leq i \leq n)$ the cardinality of the random variable X_i, $q_i (1 \leq i \leq n)$ the number of joint states of all parent nodes of X_i and $|\theta_m|$ stands for the number of independent network parameters, expressed also as $\sum_{i=1}^{n} (r_i - 1) \cdot q_i$. In special cases regarding the scoring function well known quality measures can be present for specific β allocations like $\beta = 1$ (*AIC - Akaike Information Criterion*), or $\beta = \frac{1}{2} log_2 N$ (MDL - minimum description length metric). The MDL principle frames model learning in terms of data compression. The MDL objective is to determine the model that provides the shortest description of the data set. The MDL scoring criterion is the additive inverse of the Bayesian information criterion (BIC). Different additional quality measure instances are discussed in detail in [Bor00] and [GS04].

4.3.1. Searching Strategies

In several well known searching problems the property that the state description itself contains all the information needed to find a solution is given. The path by which the solution is reached is in this case irrelevant. The calculation of the optimal BN structure includes browsing through the complete network search space. The naive approach is to measure for any search order each graph structure and give back the best valued DAG. The main problem is the huge number of different directed acyclic graphs, which depends on the node number n:

$$g(n) = \sum_{i}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)}.$$

The following table 4.3 gives an impression of the growing number of directed acyclic graphs depending on the number of network nodes n. If n exceeds the value seven machine learning applications need further searching strategies like heuristic strategies.

Table 4.3.: Number of DAGs $g(n)$ depending on the number of nodes n

n	5	6	7	8	10
g(n)	29281	$3.78 \cdot 10^6$	$2.46 \cdot 10^8$	$7.84 \cdot 10^{11}$	$4.18 \cdot 10^{18}$

Iterative improvement algorithms often provide the most practical approach. Consider all states laid out on the surface of a landscape. The height of any point on the landscape corresponds to the evaluation function of the state at that point. The idea is to move around the landscape by trying to find the highest peaks which stand for the optimal solutions [RN03].
Hill Climbing algorithms always try to make changes that improve the current state. They continually move into the direction of increasing values. Hill Climbing does not maintain a search tree, so the node data structure need only record the state and its evaluation by value. When there is more than one best successor to choose from, the algorithm can select among them via random. Instead of starting randomly in the case of stuck in a local minimum, it is also allowed for the search to take some downhill steps to escape the local minimum. This is the idea of *simulated annealing*. Instead of picking the best move, it picks a random move. If the move actually improves the situation, it is always executed. Otherwise, the algorithm makes the move with some probability degree less than one. The probability decreases exponentially with the badness value of the move. A second parameter T is also used to determine the probability. At higher values of T, bad moves are more likely to be allowed. As T tends to zero, they become more and more unlikely, until the simulated annealing algorithm behaves more or less like Hill Climbing. Hill Climbing search contains useful optimization potential which leads to a new class of structure learning algorithms which we have invented and developed as described in detail in the following Section.

4.3.2. LAGD Hill Climbing

Based on appropriate scoring functions like [Hec95] or [Kra96] a local search strategy using greedy hill climbing can be executed to compare the directed acyclic graphs m_{old} and m_{new} using $\Delta f(m_{new}) - f(m_{old})$. Look ahead in good directions (LAGD) hill climbing represents a new structure learning algorithm as generalization approach which calculates in advance k steps in regard to the chosen scoring function. Another parameter l stands for the number of best evaluated network structures per look ahead step. LAGD Hill Climbing offers a new class of parameterized algorithms including the parameters:

- number of look ahead steps k
- number of calculated good operations l per each look ahead step

The LAGD hill climbing algorithm using local search is illustrated as sequence diagram and can be expressed in pseudo code as illustrated in Figure 4.9.
As test bed for the implementation we use the WEKA environment (*Waikato Environment for Knowledge Analysis*), a collection of machine learning algorithms for data mining and knowledge discovery tasks [WF05]. We were able to integrate the developed LAGD hill climbing in the Waikato Environment and today LAGD is an integral part of WEKA.

The LAGD Hill Climbing algorithm spans a whole class of structure learning algorithms parameterized by the number of good operations l, the number of look ahead steps k, the maximal number of parents per node, the score type as instance of quality measures like Bayes, entropy, Akaike Information Criterion or Minimum Description Length, the initiation possibility as naive Bayes classifier [CG01] and the final application of a Markov blanket correction (compare Figure 4.10).

The special case $k = 1$ ($nrOfLookAheadSteps = 1$) results in standard greedy hill climbing. To reduce the calculating time adjust the parameter l to regard only the l best valued operations per look ahead step. The lower bound concerning the network structure quality agrees with standard greedy hill climbing results based on greedy k-step operation sequences. The computational complexity per iteration step (k-step sequence) with n Bayesian network nodes, k look ahead steps and l good operations per look ahead step can be determined as

$$O(\sum_{i=0}^{k-1} l^i \cdot n^2).$$

This term may be assessed by $O(l^{k-1} \cdot n^2)$, as can be seen from the following induction.
We show that

$$\sum_{i=0}^{k-1} l^i \cdot n^2 \leq 2 \cdot l^{k-1} \cdot n^2.$$

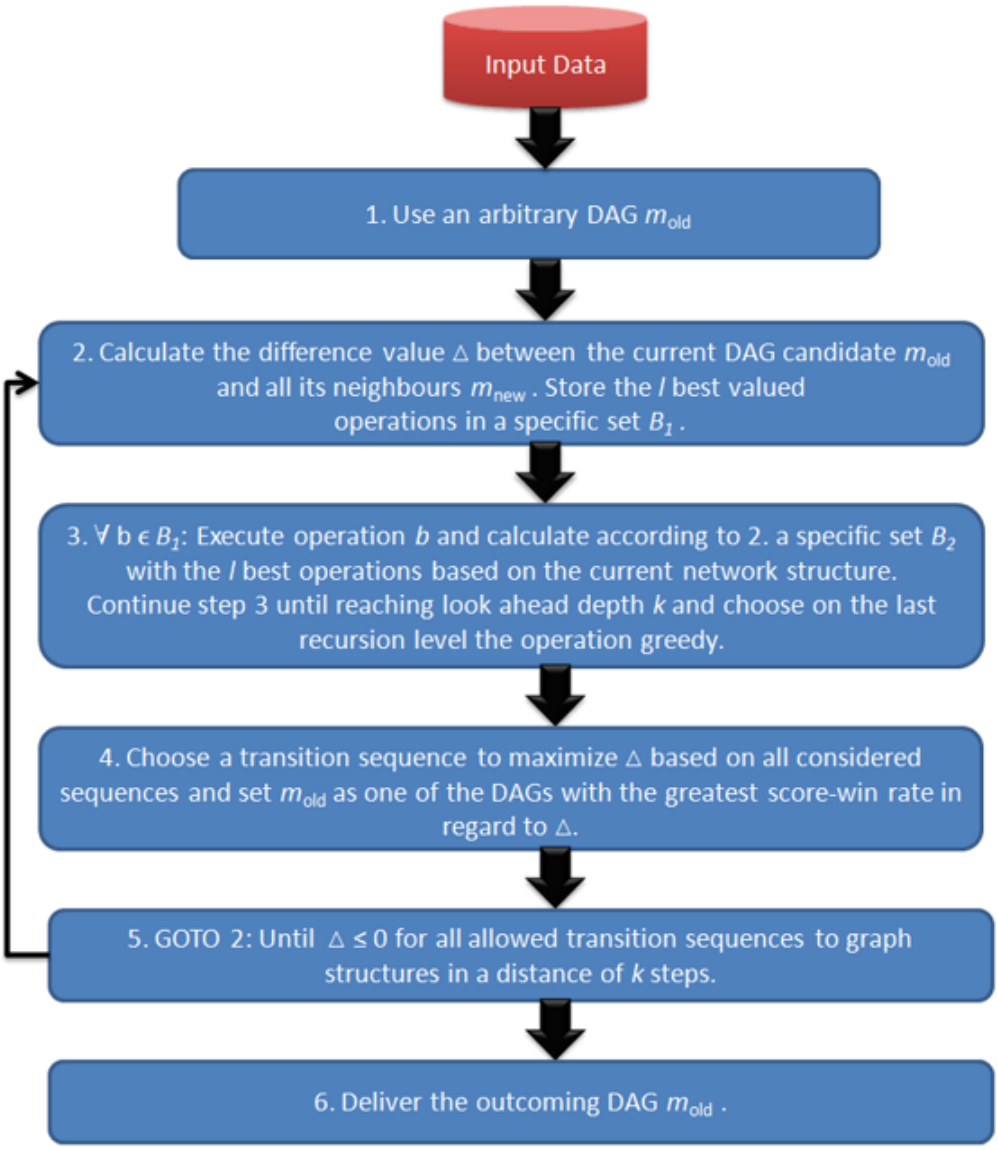


Figure 4.9.: LAGD hill climbing algorithm using local search as pseudo code sequence

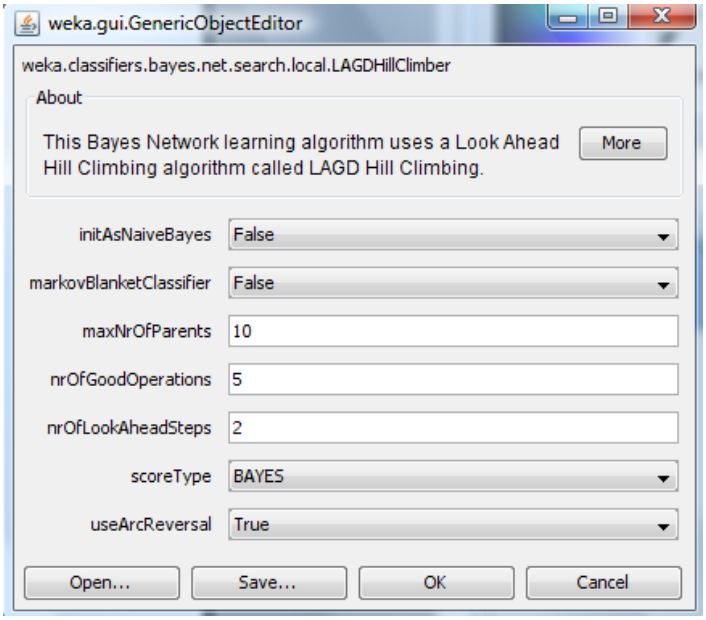


Figure 4.10.: LAGD hill climbing algorithm spans a whole class of structure learning algorithms

Initial induction step $k = 1$:

Proof.

$$\sum_{i=0}^{1-1} l^i \cdot n^2 = n^2 \leq 2 \cdot l^{1-1} \cdot n^2 = 2 \cdot n^2.$$

□

Induction step $k \rightarrow k = 1$:

Proof.

$$\sum_{i=0}^{(k+1)-1} l^i \cdot n^2 = (\sum_{i=0}^{k-1} l^i \cdot n^2) + l^k \cdot n^2 \leq$$
$$2 \cdot l^{k-1} \cdot n^2 + l^k \cdot n^2 \leq l \cdot l^{k-1} \cdot n^2 + l^k$$
$$\cdot n^2 = 2 \cdot l^k \cdot n^2 = 2 \cdot l^{(k+1)-1} \cdot n^2.$$

□

Obviously $\sum_{i=0}^{k-1} l^i \cdot n^2$ can be assessed by $2 \cdot l^{k-1} \cdot n^2$. The argument follows when neglecting the multiplier two according to Bachmann-Landau notation.

4.3.3. Experimental Results for Medical Data Using LAGD Hill Climbing

Here a medical test dataset was chosen to compare the introduced LAGD Hill Climbing algorithm with other classical well known algorithms like simulated annealing or standard greedy hill climbing used for different metrics as instances of quality measures (i.e. AIC, MDL). The ALARM network [BSCC96] is a commonly used network which is a representative of a real life Bayesian network. It was originally described by Beinlich as a network for online-monitoring of patients in intensive care units. The ALARM network structure consists of 37 nodes and 46 edges with 8 diagnosis, 16 medical findings and 13 temporary variables (see Figure 4.11). A Monte Carlo technique named Probabilistic Logic Sampling was used to generate case databases consisting of 10.000 test cases based on the ALARM network. Probabilistic Logic Sampling generates each case or sample by orienting to the weak node order induced by the underlying directed acyclic graph [Hen88]. The criterion to compare the quality or performance of different BNs learned according to different structure learning algorithms is the so called *LogScore*, which is based on the appropriate local score metric. When applying for instance the Akaike Information Criterion (AIC with $\beta = 1$) the LogScore AIC derives as follows:

$$LogScore_{AIC}(m, D) = -h_2(m, D) = (N \cdot \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \frac{N_{ijk}}{N} log_2 \frac{N_{ijk}}{N_{ij}}) - \beta \cdot |\theta_m|.$$

I. e. derive the LogScore AIC by simply multiplying the AIC metric by minus one. It follows from the formula above the equation $LogScore_{AIC}(m, D) = -h_2(m, D) = f_2(m, D)$.

Figure 4.12 illustrates the learning curve for the dataset ALARM using the parameter values *maxNrOfParents = 5*, *nrOfLookAheadSteps = 2* and *nrOfGoodOperations = 5* with LogScore AIC ($\beta = 1$). The quotient $\frac{N_{ijk}}{N_{ij}}$ is always within the interval $(0, 1]$, i. e. the resulting logarithm of this quotient is obviously less than or equal to zero.

The LogScore starts strongly negative and rises successively in finding better valued neighbour graphs. With increasing the number of iteration steps the model-complexity function reduces the total LogScore by the summand $\beta \cdot |\theta_m|$. The experimental results evaluate different structure learning algorithms like simulated annealing, greedy hill climbing and LAGD based on LogScore metrics Bayes, AIC and MDL (see Figure 4.13). Figure 4.14 displays in detail the dependency between the quality of the calculated network structure and the number of look ahead steps with fixed parameter value $l = 2$. Here choose the minimum description length as local score metric for an exemplary comparison of greedy hill climbing and LAGD since results were very similar using the other metrics AIC and Bayes. It is obvious that increasing LAGD look ahead steps causes a better network quality. The startling bending down of the curve when increasing the look ahead parameter from *5* to *6* may be explained as follows. With look ahead parameter $k = 5$ LAGD finds a good local optimum, while with parameter $k = 6$ LAGD falls in a trap, when seeing a better evaluated

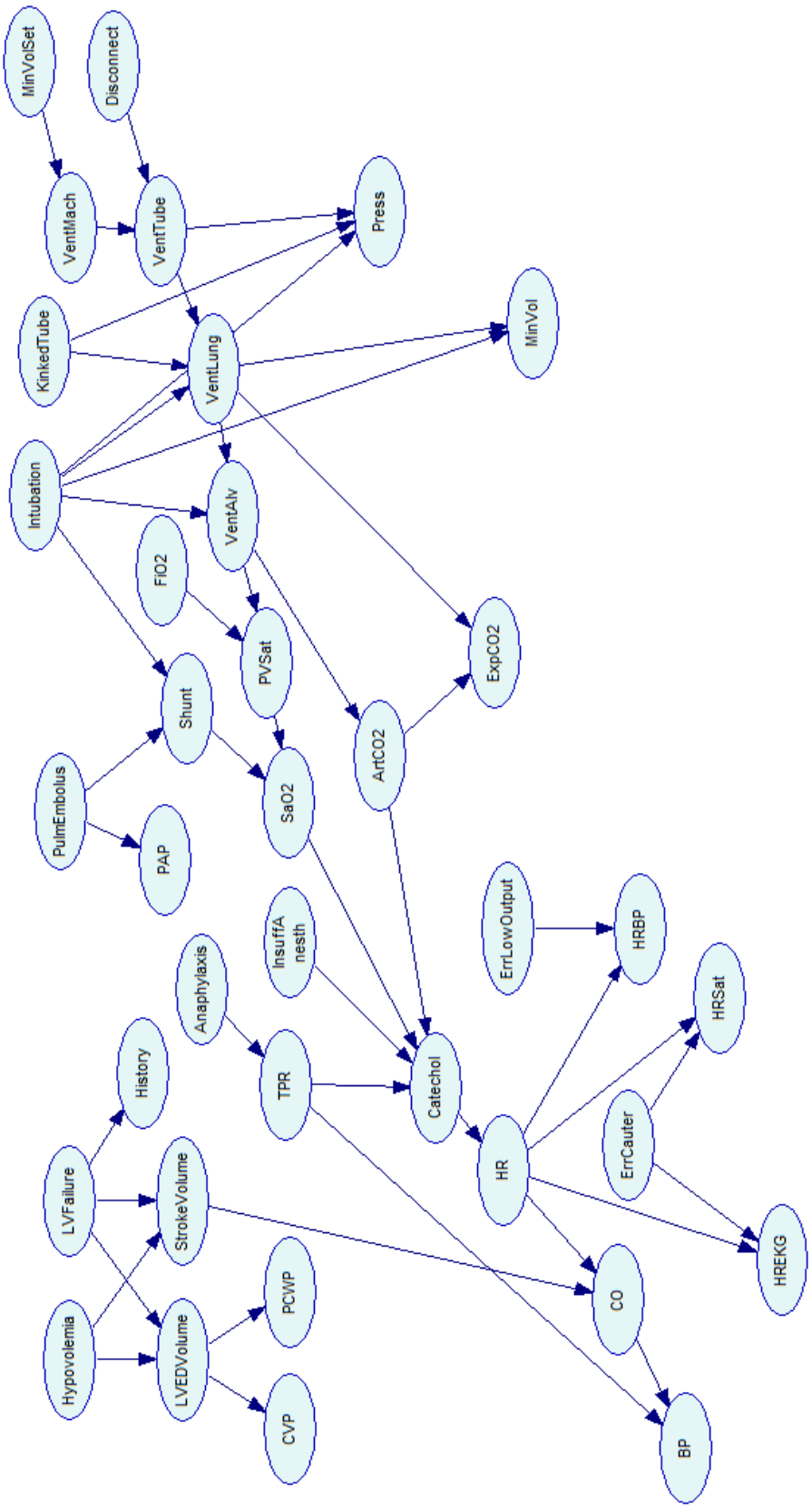


Figure 4.11.: ALARM monitoring network prepared with GeNIe for structure modeling

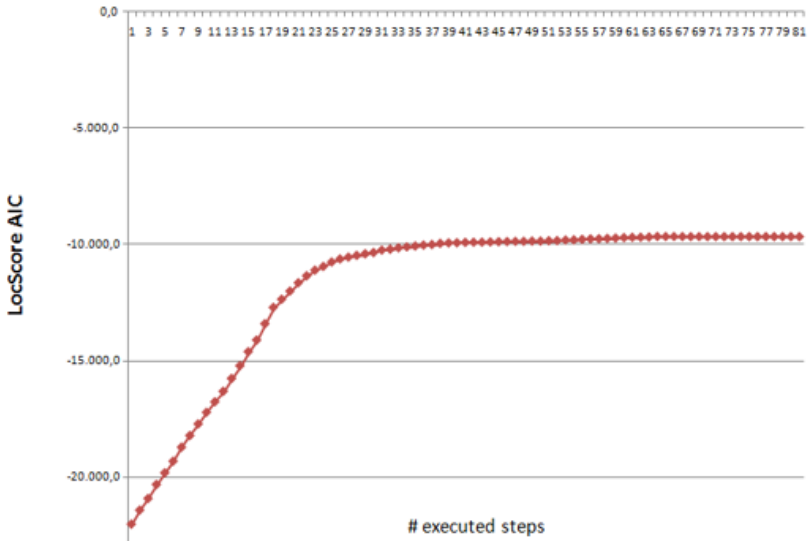


Figure 4.12.: ALARM LAGD hill climbing learning curve with LogScore AIC

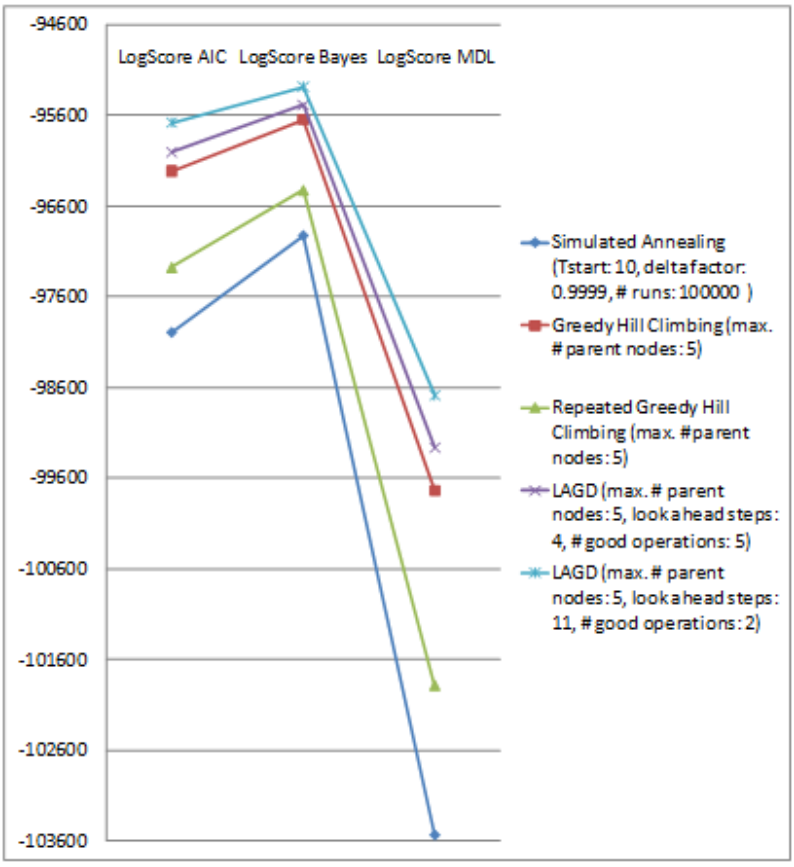


Figure 4.13.: Structure learning results comparing different Bayesian network structure learning algorithms with LogScore metrics AIC, Bayes and MDL

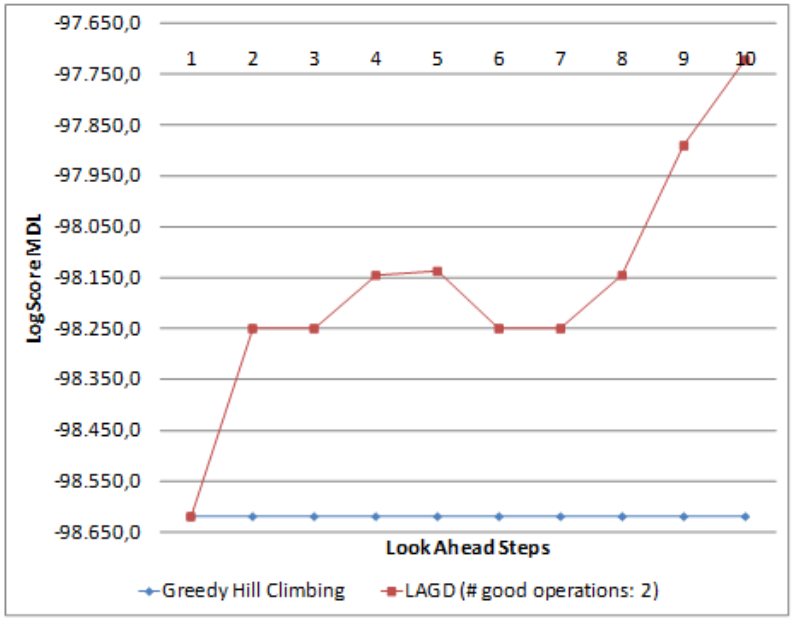


Figure 4.14.: Comparison of greedy Hill Climbing and LAGD with varied look ahead steps using LogScore MDL

graph structure in the beginning. LAGD follows this "false" path, that finally results in a local optimum, which is unfortunately worse than in the case $k = 5$. The first steps are executed independent of the chosen search depth based on greedy behaviour. The algorithm starts with an empty network structure in filling step by step at first those edges, which perform the maximized score value. In a growing connected structure with increasing complexity acyclic effects influence directly the next edges have to integrate in the model.

4.4. Competing Fusion of Distributed Knowledge

The development of knowledge-based systems involves knowledge acquisition from a diversity of sources often geographically distributed. It is obviously difficult to bring together information from different knowledge sources [Mah07] about a subject of common interest. The sources include for instance written documents, interviews, and application data stored in distributed knowledge bases disposed from different experts often specialized in fields like mechanical engineering, computer science or medicine (for instance experts in intensive care, ophthalmology or internal medicine [Luc01]).
Intuitively merging knowledge bases is to find a knowledge base that has at least as much information as each component knowledge base and is the smallest such medical knowledge base. Different experts working together are not in a position to generate the complete Bayesian network structure including conditional probability tables and all necessary random variables. An medical expert E_i with $(i = 1, ..., n)$ working in a specific hospital division has only access to specific medical data or medical knowledge to build a substructure

of a complete Bayesian network structure. The main problem occurs when all included experts compose their individual knowledge to build up the Bayesian network structure representing the domain knowledge [HB06]. Integrate knowledge stored in different Bayesian networks BN_i through techniques of knowledge fusion. Researchers differentiate the aspects *competing*, *complementary* and *cooperative* knowledge fusion. Aggregating expert beliefs is in this sense the task of performing an expert group consensus probability distribution by combining the beliefs of the individual experts of the group in some fashion [PW05].
Competing fusion techniques have been compared and applied by the author of this thesis [HFAN08a], which focus on the unification of expert knowledge often realized in one problem domain with different involved experts. A new sequence diagram (see Figure 4.16) is given for sampling aggregation, which explains how to deal with different Bayesian models within the same problem domain to calculate an aggregated output model. The author has also compared the sampling aggregation technique with linear opinion pool as used method of aggregating the likelihood assessments of different experts in the case of the ALARM network as benchmark. The achieved consolidation process based on diverse network structures and CPT entries founded of multiple expert domain knowledge. The challenge in the complementary case consists of the integration of (partly) disjoint network structures. Obviously, this fusion type solves the problem of individual incompleteness of expert domain knowledge. Cooperation fusion relates to the fact, that a developed network structure with CPT parameters of an expert depends on the model of another expert in the same problem domain.
To merge different Bayesian networks (for example BN_1 and BN_2) it is necessary to determine a measure for the approximation quality. A suitable measure is the Kullback-Leibler divergence between Bayesian network structures like BN_1 and BN_2. The *Kullback-Leibler divergence* [vdGR01] expresses the difference or distance between two probability distributions. Given the probability distributions p and q define $KL(p,q)$ as follows:

$$KL(p,q) = \sum_x p(x) log \frac{p(x)}{q(x)}.$$

The Kullback-Leibler divergence is obviously not symmetric and can also be verbalized using the cross entropy $H(p,q)$ as follows:

$$KL(p,q) = -\sum_x p(x)\ log\, q(x) + \sum_x p(x)\ log\, p(x) = H(p,q) - H(p).$$

The cross entropy between two probability distributions measures in information theory the average number of bits needed to identify an event from a set of possibilities, if a coding scheme is used based on a given probability distribution q, rather than the *true* distribution p. The KL divergence values are not negative with $KL(p,q) = 0$ if and only if $p = q$. Competing fusion includes combined expert knowledge from different fields like medical, financial or engineering problem scenarios. Each expert can generate a case database

with embedded and not obvious inferable conditional probability table settings and network structures. The Bayesian network learning algorithm introduced in Section 4.3 delivers Bayesian networks for each expert to fuse via sampling or via LinOP aggregation.

Competing fusion via sampling contains the following steps:

1. Synthesize for each expert network a case database using a Monte Carlo technique.
2. Aggregate the expert case databases.
3. Learn the aggregated Bayesian network structure based on the case database determined in 2. using an automatic learning algorithm.

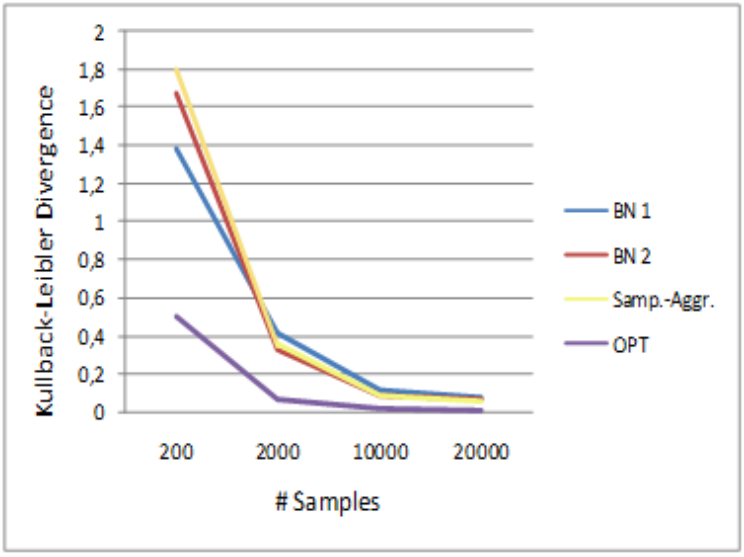


Figure 4.15.: Sampling aggregation results with Kullback-Leibler divergence

The ALARM network [BSCC96] was split in two disjoint case databases based on a Monte Carlo technique named Probabilistic Logic Sampling. The respective Bayesian networks BN_1 and BN_2 were learned with LAGD. According to the described sampling algorithm the Bayesian network Sampl.-Aggr. was generated. The number of samples was varied between 200 and 20000. Results are illustrated in Figure 4.15.

The sampling aggregation sequence diagram in Figure 4.16 explains how to aggregate for instance two models of the same problem domain obtaining an aggregated model as *aggr_BN.net*. There are many systems for academic use free of charge. A *BN Converter* transforms an extensible markup language file in a *Hugin* and *GeNIe* readable $*.net$ file type. *GeNIe* is a graphical network interface to *SMILE* (Structural Modeling, Inference, and Learning Engine) as portable Bayesian inference engine. A graphical editor to create and modify network models is given. It is also possible to allocate cross compatibility with

other software like *Hugin* and the *Hugin* graphical user interface which is an interactive tool enabling to use the facilities of the *Hugin Decision Engine*. The *Hugin Decision Engine* performs reasoning on a knowledge base represented as a Bayesian network or a Decision network. The engine performs all data processing and storage maintenance associated with the reasoning process. Given an original *BN.net* file representing the original network structure, generate examples applying for instance logic sampling and splitting the output in two commensurate parts. The split sample files deal as basis for the WEKA tool to put into action the LAGD hill climbing algorithm (step 3). Simultaneously, the split network files *split_BN1.xml* and *split_BN2.xml* are converted in a *Hugin* and *GeNIe* readable $*.net$ file format. In this step 4, generate additional samples appropriate for the aggregation step 5 with a considerable sample size (factor m with $m \geq 1$). The aggregated sampled file serves as basis to learn the aggregated output model *aggr_BN* based on the previous sampling steps on the split parts.

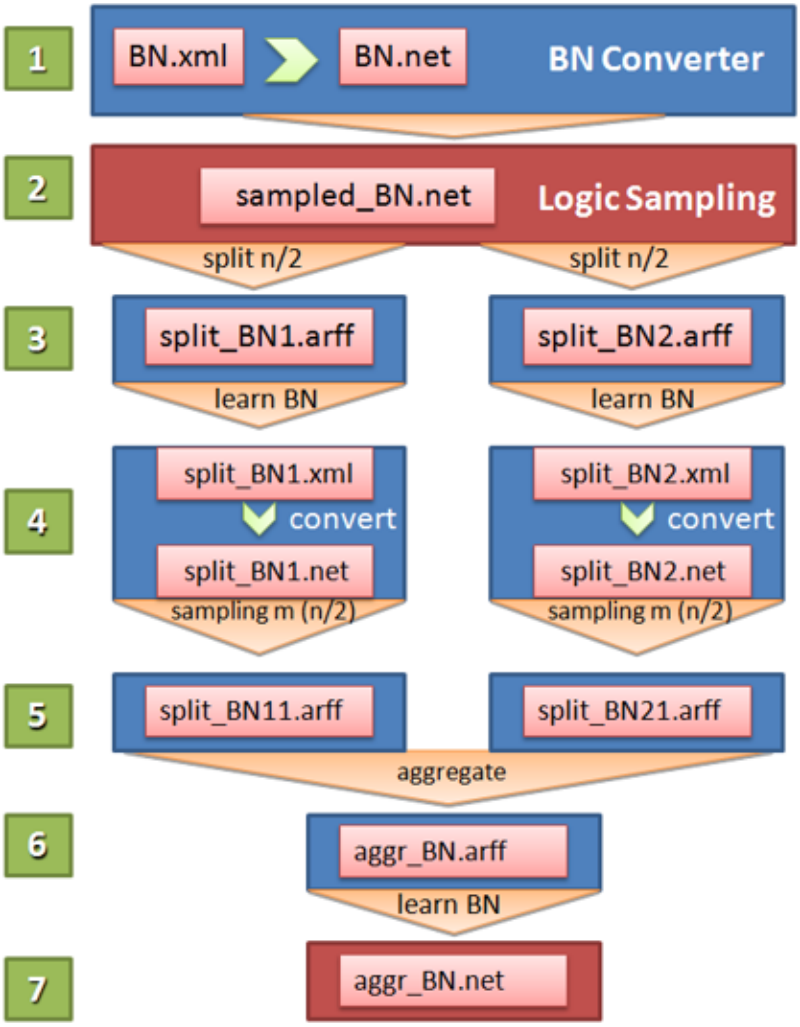


Figure 4.16.: Sampling aggregation sequence diagram

The use of Monte Carlo techniques and the hereby induced noise may be avoided by applying an aggregation operator for a common unified probability distribution. The aggregation of L expert probability distributions $p_1, \ldots, p_L$ using the LinOP operator [Pra97] leads to the following algorithm:
1. Aggregate the probability distributions $p_1, \ldots, p_L$ of L Bayesian networks using the *LinOP* operator for a common probability distribution

$$p^* = \sum_{i=1}^{L} \frac{\alpha_i}{\sum_{j=1}^{L} \alpha_j} p_i$$

with *LinOP*

$$LinOP(\alpha_1, p_1, ..., \alpha_L, p_L) = \sum_{i=1}^{L} \alpha_i p_i.$$

2. Learn the aggregated Bayesian network with a local score metric based learning algorithm using p*.

The ALARM network results using competing fusion via *LinOP* aggregation [CKO01] demonstrates the following Figure 4.17:

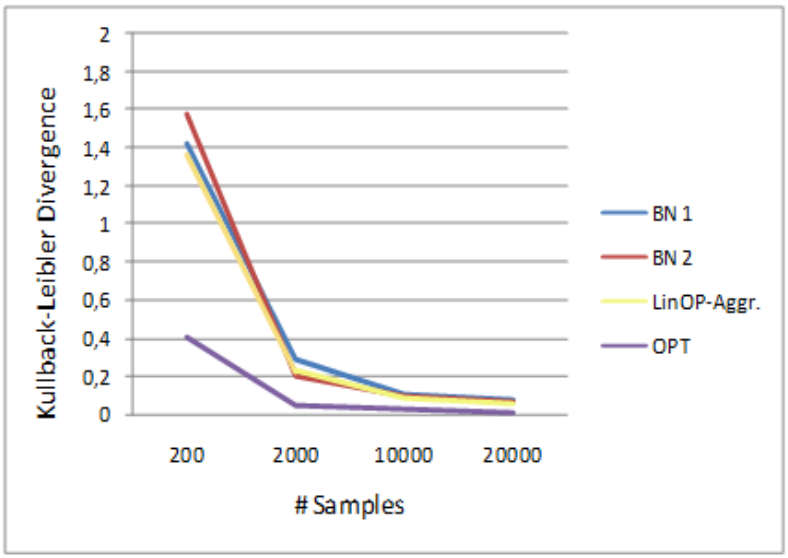


Figure 4.17.: LinOP aggregation results with Kullback-Leibler divergence

When comparing sampling aggregation to *LinOP* aggregation in the case of small sample sizes (less than 2000) the results show that *LinOP* aggregation clearly outperforms sampling aggregation. Due to the introduced noise the aggregated network *Sampling-Aggregation* has an even higher Kullback-Leibler divergence compared to the original Bayesian networks BN_1 and BN_2. Increasing successively sample sizes causes in both cases an asymptotical alignment of Kullback-Leibler divergence of the aggregated network structure to the optimum.
Recapitulating a new local score metric based structure learning approach called

LAGD Hill Climbing was presented here. The number of look ahead steps and the number of operations considered for look ahead are configurable, which spans a whole class of structure learning algorithms. Both the time taken for computing and the quality of the calculated Bayesian network structure may be tuned by adjusting these parameters. The test result for the ALARM network using quality measures like LogScore AIC, Bayes and MDL clarifies the advantages of LAGD in regard to the achievable network quality. LAGD is in the meantime fully integrated in the Waikato Environment for Knowledge Analysis and may be downloaded via the WEKA website. The usage of a generalized structure learning approach including dynamic interdependencies between look ahead steps and number of good operations is intended for the next steps. LAGD hill climbing may be applied in the field of mechanical engineering and especially product lifecycle management to learn from existing condition monitoring data and reveal relationships between sensor data, environmental parameters and failure events (see Chapter 7). In this context competing fusion improves the quality of aggregated knowledge models based on distributed knowledge sources. Aggregation and fusion methods like sampling aggregation and LinOP aggregation make the generated condition monitoring knowledge collected in the product use phase from various customers within a feedback cycle usable for further product development and help to improve the usability and quality of the next generation of a given product, which is topic in [AFHN08a].

4.5. Object-Oriented Bayesian Networks

Large and complex systems are often composed of collections of identical or similar components. Models of such systems will naturally contain repetitive patterns. Typical for complex systems is their composition containing a large number of similar or even identical components, which should be reflected in models of the system to support model construction, maintenance, and reconfiguration. For instance, a diagnosis model for diagnosing machine downtimes could reflect the natural decomposition of a machine into its engine, electrical components, fuel system and so on. Object orientation is a modeling and programming technique that makes use of classes and objects as fundamental building blocks [NFN00]. Objects are used as a generic term for an instance of a class. Classes are arranged in a parent-child hierarchy, where a subclass as child may be allowed to inherit variables and methods of its parent. The basic object-oriented Bayesian network (OOBN) mechanisms support object-oriented specifications of BNs, which makes it simple or easier to reuse models, to encapsulate sub-models, and to perform model construction in a top-down fashion, a bottom-up fashion, or a mixture of both allowing repeated changes of level of abstraction. This comes from the human thinking about systems in terms of hierarchies of abstractions and due to the lack of ability to mentally capture all details of a complex system simultaneously. Hierarchical fashion makes graphical models often less cluttered, which facilitates the communication between knowledge engineers, domain experts, and end users. An OOBN

is a network that, in addition to the usual nodes, contains *instance* nodes. An instance node represents an instance of another network encapsulated in the model. Nodes that are not an instance of a network class represent a basic variable. An instance itself represents an instantiation of a network class within another network class. A network class is a named and self-contained description of a BN and can be characterized by its name, interface, and hidden part. Instances can be nested, so an object-oriented network can be realized as a hierarchical description of a specific problem domain. Figure 4.18 illustrates an instance of a network class within another network class. An instance connects to other variables via some of its basic variables. These variables are known as *interface* variables. The interface nodes usually comprise a strict subset of the nodes of the instance. Interface nodes are subdivided into input and output nodes demonstrated in Figure 4.18. In general, a network class C is a DAG over three pairwise disjoint sets of input nodes $\mathcal{I}(C)$, hidden nodes $\mathcal{H}(C)$, and output nodes $\mathcal{O}(C)$ of C. The scope $\mathcal{S}(C)$ of a network class C is the set of variables and instances which can be referred to by their names inside C. The scope of the network C_N in Figure 4.18 is $\mathcal{S}(C_N) = \{C_1, C_3, C_2, M\}$. The class instance M of the network class C_M is instantiated within another network class C_N. The network class C_N has input variable C_1, hidden variables C_3 and M, and output variable C_2. The network class C_M has input variables C_1 and C_2, output variable C_3, and hidden input variables. The input variable C_1 of instance M is bound to C_1 of C_N, whereas C_2 is unbound. When referring to $C_1 \in \mathcal{I}(C_M)$ inside C_M, we are in fact referring to $C_1 \in \mathcal{I}(C_N)$ as $C_1 \in \mathcal{I}(C_M)$ in instance M as placeholder for $C_1 \in \mathcal{I}(C_N)$.

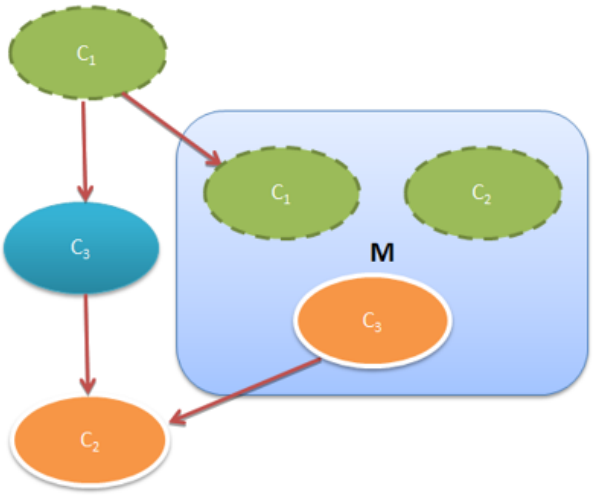


Figure 4.18.: M is an instance of a network class C_M within another network class C_N

OOBNs are also appropriate to model temporal relationships amongst variables. As discussed so far, a BN represents only the probabilistic relationship among a set of variables at some point in time. It includes no information how the value of some variable may be related to its value and the values of other variables at previous points in time. In many real-world situations it is

fundamental to have the ability to model temporal relationships. In medicine it is important to represent and reason about time in tasks such as diagnosis or for processing diagnosis in plant operation and maintenance. Interpret a dynamic BN as extension of BNs to model temporal processes [Gha98]. Assume, that changes occur between discrete time points t with $0 \leq t \leq T$. Let $\{X_1, ..., X_n\}$ be the set of features whose values change over time, and let $X_i[t]$ be a random variable representing the value of X_i at time t. For all t, each $X_i[t]$ has the same space depending on i. A dynamic BN contains the variables that constitute the T random vectors $X[t]$ [Nea04]. An important case comes into consideration between variables at successive time steps by temporal arcs. Relationships between the same variable over time or different variables over time come into consideration. For instance, the temperature of an engine or specific measurable disease values of a patient may change over time. One must consider that a fully temporally connected network structure or short intervals between time slices would lead to complexity problems [SY07].

Example 4.2. Belt Conveyor: The production of high quality products in manufacturing systems engineering sometimes shows an undesired behaviour. Since it is a significantly concern to further improve the quality of high quality products, a lot of effort is dedicated to find the causes of these faults in order to be able to prevent similar faults from occurring in the future. In a database for belt conveyor production its configuration (product line, engine type, special equipment etc.) and any faults detected during production or maintenance are recorded.

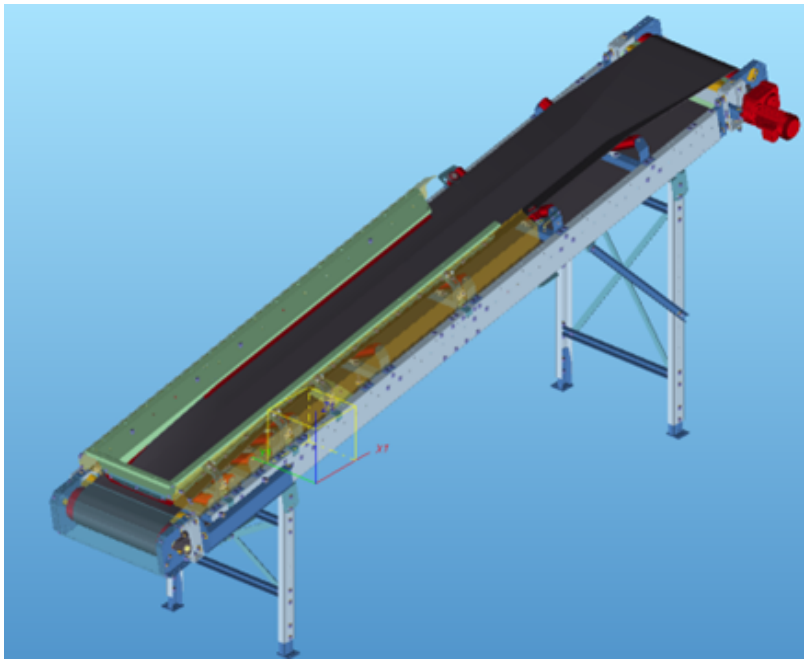


Figure 4.19.: Belt conveyor and its configuration components [AFHN07]

Leading manufacturers in belt conveyor production offer their customers the possibility to configure their final product individually. As a consequence there

are many different configurations, each of which is bought only very few times (the average number of exactly identical conveyors).
Therefore it is not possible to monitor the behaviour of individual configurations, there are too few example cases for each of them. Unfortunately, it is not uncommon that a propulsion fails only when installed in combination with specific other components. Therefore a simple check whether individual propulsions show an unusually high frequency of failure cannot uncover all weaknesses. Their normal behaviour in other configurations may hide the problem. A condition monitoring approach (see also Chapter 7 based on Bayesian networks from sensor data during the production phase) was installed in order to detect dependencies between faults and belt conveyor properties. The underlying idea is to exploit the search methods and evaluation measures developed for realizing such graphical networks from sensor data to search automatically for sets of attributes that are strongly dependent on each other. The belt conveyor configuration and attributes during its usage over time t produces the following dynamic Bayesian network structure specified as object-oriented representation.

Figure 4.20 illustrates a DBN for the introduced belt conveyor scenario. It shows the progress from torque, block part, and non-return device over time. For instance, modifications of torque sensor values depend at the next point of time on the torque value at the current point of time.

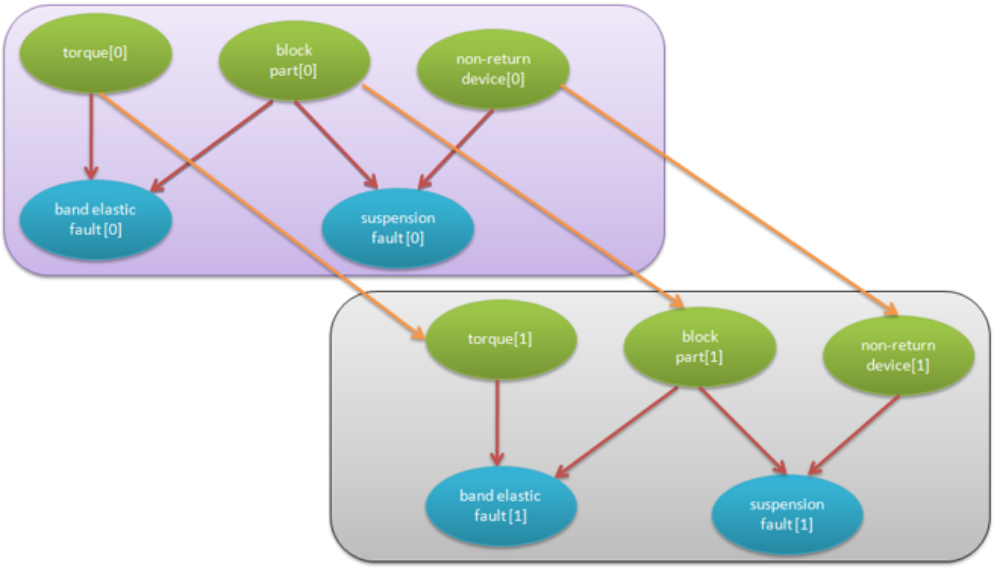


Figure 4.20.: Belt conveyor scenario as dynamic Bayesian network representation with points of time t_0 and t_1

Assume that the structures and the tables of the points of time are identical, and that the transition probabilities are identical for all points of time, the construction of temporal models using instance nodes is very efficient and makes the structure less cluttered.
Figure 4.21 takes as input the component nodes of the previous point of time, where the belt conveyor component nodes act as output nodes, since they should

be bound to the input nodes of the next point of time. To construct two points of time, create two instances of the belt conveyor network and bind the outputs of point of time 1 to the inputs of point of time 2.

To summarize, object-oriented constructs incorporate encapsulation, inheritance, and hierarchy. Its common purpose is efficient modeling and simulation, and providing a convenient language for reuse and exchange of models. OOBNs support a type of object-oriented specification of networks, which takes over characteristics and makes it simple to reuse models, to encapsulate sub-models, and to perform model construction at different levels of abstraction. The use of OOBN models facilitates the construction of rather large and complex domains allowing modifications of BN fragments.

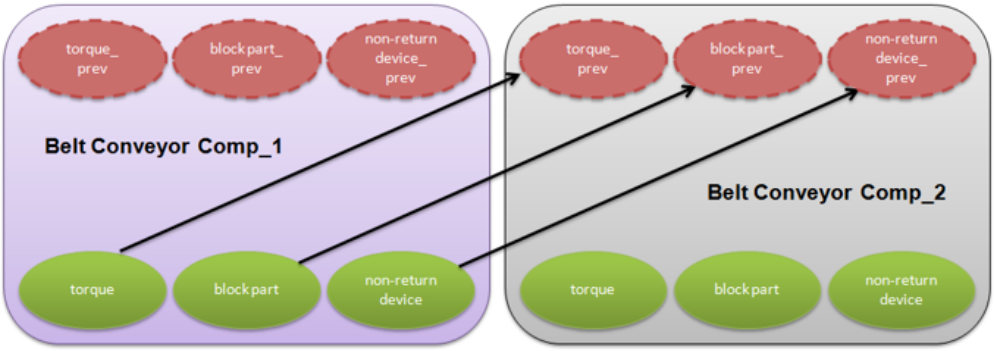


Figure 4.21.: Belt conveyor scenario as dynamic Bayesian network specified as object-oriented network

In practice, OOBN is deployed to model systems and processes, which often are composed of collections of almost identical components. Models of such systems often contain repetitive pattern structures like models of sensors, environment variables, or process assets. Use OOBNs to model signal uncertainties and signal level-trend classifications as small standardized model classes within the problem domain. Weidl [Wei02] use OOBNs for top-down and bottom-up root cause analysis of industrial systems in order to facilitate the construction and usage of models. Repeated changes in the plant and process hierarchy are needed due to the fact that process engineers, operators, service providers, and maintenance staff discuss and argue about systems in terms of process hierarchies. They are often overloaded with detailed facts and details of complex systems in simultaneous causal analysis of disturbances. It also proves to be useful for explanation and visualization of analysis conclusions [WMD03]. The usage of dynamic BNs in this context is recommendable for prediction and risk assessment of events in the process operation chain. Instance nodes are often representing network classes for predicted development of process variables.

4.6. Bayesian Networks Review

Bayesian networks are graphical structures for representing the probabilistic relationships among a frequently large number of variables and for executing probabilistic inference with those variables. BNs have become an increasingly popular paradigm for reasoning under uncertainty addressing many tasks such as diagnosis, decision making or data mining. The graphical nature of BNs gives a decision maker a much better intuitive grasp of the relationships among the nodes, which represent propositional variables in a problem domain. The visualization aspect provides also an instrument for formulating, communicating, sharing, transferring and discussing qualitative interaction models in problem domains. The dependency relationships among the variables are expressed by the arcs between the nodes. The DAG of a BN is encapsulated in a compact graphical representation of the dependence and independence properties of the joint probability distribution represented by the model. With this, DAGs provide a powerful language for expressing and reasoning about causal relationships among variables. In detail, such graphical models provide an inherent mechanism for realizing causal, diagnostic, as well as inter-causal reasoning. The last-mentioned inter-causal reasoning is unique for graphical models and is one of the key differences between automatic reasoning based on probabilistic networks and system solutions based on production rules. We can distinguish between problem domains where causal mechanisms are known and where such mechanisms are unknown but can be revealed through learning of the model structure from data. In the first case, the DAG in a BN is hand-constructed by a domain expert. Then the conditional probabilities are assessed by the expert, learned from data, or obtained using a combination of both techniques. In the second case, a Bayesian structure learning algorithm is used assuming that we have a set of random variables with an unknown relative frequency distribution. BN structure learning algorithms have been developed in most instances for the case of discrete variables. When a single structure is not found to be most probable, averaging over structures is sometimes more appropriate. Another point deals with learning structure when there are missing data items or hidden variables. We have developed and explained in more detail in this chapter a new parameterized structure learning algorithm named LAGD, which is validated with a benchmark case. Finally, consider the situation where application data is stored in distributed knowledge bases representing different experts working in the same problem domain. Knowledge fusion techniques are necessary to aggregate this individual accessible expert knowledge to obtain the complete BN structure including conditional probability tables and random variables. Fusion techniques are used to reduce some type of noise, increase accuracy, summarize information or extract information. Knowledge fusion is a process of combining different expert data into one single datum, which improves the understanding of the application domain. In this context, the author of this thesis has introduced a new sampling aggregation workflow for competing fusion of BNs. With this, a knowledge engineer has the possibility to apply an aggregation mechanism on the basis of stochastic sampling. A graphical editor is available to handle and follow up the obtained aggregated model.

5. Decision Networks

Decision networks or influence diagrams can be interpreted as extension of Bayesian networks augmenting them with decision variables and utility functions. They provide a language for sequential decision problems for a decision maker, where there is a clear order among the decisions. A decision network provides a natural representation for capturing the semantics of decision making with a minimum of interferences for the decision maker [Paw04]. This chapter introduces the semantics of decision networks in detail and describes an extension to support decision making processes [YK03]. Adding an explicit representation of the actions under consideration and the value of utility of the outcomes gives decision networks a basis. Based on utilities a description of how they are represented together with probabilities in decision networks is given. With planning, it is possible to determine the best sequences of decisions or actions. A generalization of decision networks allows the construction of dynamic decision networks [Yos01], which model temporal aspects of decision making or planning under uncertainty [Bly98]. In many real-life decision situations the available preference information of the decision maker is often vague, imprecise or uncertain. Among the appropriate tools to overcome these difficulties is fuzzy logic. This chapter describes preference models that extend the reliability and flexibility of classical decision models. Concluding different examples illustrate the use of decision networks for decision making under uncertainty.

5.1. Introduction

Decision-theoretic models are *declarative* in nature. The basic ingredients of a model are a description of the decision maker's environment and the possible actions, including information about the effect of actions to the environment, the decision maker's preferences, and the decision maker's beliefs. Apart from that, a decision model includes a principle of rationality [EW03]. Probably the best known rationality principle is the expected utility theory [Aug02]. The environment is characterized by a set of possible world states, the effect of an action depends on the true but unknown world state, and the decision maker's preferences and beliefs are modeled, respectively, by means of utility function and a probability measure.
In general, decision theory is a means of analyzing which of a series of options should be taken when it is uncertain exactly what the result of taking the option will be [Rai68]. Decision theory is focused on identifying the best decision option, where the notion of best is allowed to have a number of different meanings, of which the most common one is that which maximizes the expected utility of the decision maker. This theory provides a powerful tool to analyze scenarios in which a decision maker must make decisions in an unpredictable environment

[HR92]. Solving a decision problem [Rao07] amounts to first determining an optimal strategy that maximizes the expected utility for the decision maker or agent and second computing the maximal expected utility of adhering to this strategy.
The classical decision theory is a set of mathematical techniques for making decisions about what action to take when the outcomes of the various actions are not known [Lau05]. Consider an agent who operates in a complex environment typical in real-life situations. The agent is inherently uncertain about that environment, it simply does not have enough information about the environment to know either the precise current state of the environment, or how that environment will evolve. For every variable S_i which captures some aspect of the current state of the environment, all the agent typically knows is that each possible value s_i of each S_i has some probability $P(S_i)$ of being the current value of S_i. Consider the following settings of Bayesian decision theory. The decision model consists of several ingredients which is described next in more detail. Let $\mathcal{A}$ denote the set of possible actions the decision maker can perform in the environment and which are available to him. $\mathcal{S}$ is the set of possible world states or states of nature, with ξ being a probability measure defined on some σ-field $\mathcal{T} \subseteq 2^{\mathcal{S}} : \xi(\vartheta) = \xi(\{\vartheta\})$ is the (prior) probability (density) of the world state $\vartheta \in \mathcal{S}$. We employ the same notation for a probability measure and its associated probability function. A consequence function c maps any pair $(a, s) \in \mathcal{A} \times \mathcal{S}$ consisting of an action $a \in \mathcal{A}$ and a state $s \in \mathcal{S}$ to the corresponding outcome $c(a, s)$. The utility of the decision maker's action depends on the world state and is determined by means of a real-valued utility function $\mathcal{U} : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$. Any outcome $c(a, s)$ with $(a, s) \in \mathcal{A} \times \mathcal{S}$ is mapped to a real number $U(c(a, s)) \in \mathbb{R}$ as utility of the outcome $c(a, s)$. $U(a, \vartheta)$ is the utility experienced by the decision maker if he performs action a in world state ϑ. The decision maker cannot observe the world state directly. However, he has access to some information represented by a random variable

$$X : (\Omega, \mathcal{O}, \nu[\vartheta]) \to (\mathbb{R}^n, \mathfrak{B}_n),$$

where $\mathfrak{B}_n$ denotes the class of Borel subsets of $\mathbb{R}^n$. The distribution of X depends on the world state. For each $\vartheta \in \mathcal{S}, X = X_\vartheta$ is a random variable with distribution $\mu = \mu[\vartheta] = X(\nu[\vartheta])$. Let $\mathfrak{D}_X \doteq X(\Omega)$.
Utilities provide a convenient means of encoding the preferences of an agent. It is possible to define utility functions that faithfully encode preferences such that a state S_i is preferred to S_j, if and only if it has a higher utility for the agent. Consider the decision making problem in different distinguishable situations. Under certainty the rational agent knows the state of the world $s^* \in \mathcal{S}$ and should choose an action $a \in \mathcal{A}$ that maximizes the utility $U(c(a, s^*))$ of the certain outcome. The problem becomes difficult when the agent does not know the state s^*. We distinguish between

- **Decision making under risk:** The agent can represent his knowledge by a maybe subjective probability distribution $\pi : \mathcal{S} \to [0, 1]$. It has been proven, that under risk any rational agent should choose that action $a \in \mathcal{A}$, which maximizes the expected utility [Vau97]. Prerequisite for

the determination of the result is that the agent agrees with some axioms of rational choice [Lui07] and that the utility function U is defined accordingly.

- **Decision making under ignorance:** The agent only knows the set of possible states $\mathcal{S}$ but has no further information. In this context, it is generally impossible to determine an action a that any rational agent should choose. However, it is sometimes possible to eliminate certain actions from the state of actions to consider actions that no rational agent should ever choose. Besides, various heuristics (e.g. maxmin rule) have been proposed which can be regarded as methods for making good decisions in many situations. Again, the agent knows only the set of states $\mathcal{S}$. This corresponds to the completely vacuous belief function Bel on the universe $\mathcal{S}$ which is defined by $Bel(\mathcal{S}) = 1$ and $Bel(\mathcal{X}) = 0$ for each $X \subset \mathcal{S}$.

By making decisions, the probabilities of the configurations of the network are influenced. To identify the decision option with the highest expected utility, calculate the expected utility of each decision alternative. If $\mathcal{A}$ is a decision variable with options a_i, the decision behaviour prescribed by the principle of *maximum expected utility* is determined by some optimal decision function $\Delta^* : \mathfrak{D}_X \rightarrow \mathcal{A}$. For each observation $x \in \mathfrak{D}_X$, this function prescribes an action that maximizes the expected utility whenever such an action exists:

$$\Delta^*(x) \in arg\ max_{a \in \mathcal{A}}\ EU(a|x),$$

where $EU(a|x)$ is the expected utility of action a given the information $X = x$. It is also possible to illustrate the expected utility $EU(a_i)$ of performing action a_i as

$$EU(a_i) = \sum_j U(a_j, h_j) P(h_j|\varepsilon),$$

where $P(\cdot)$ represents the belief of the decision maker in H as hypothesis with states h_i given ε as set of observations in the form of evidence. The utility function $U(\cdot)$ encodes the preferences of the decision maker on a numerical scale. Choose the alternative with the highest expected utility.
It is here important to distinguish between observations and actions. An observation of an event is passive in the understanding that we can assume an observation that does not effect the state of the world whereas the decision on an action is active in the sense that an action enforces a certain event. The event affected by a decision may or may not be included in the model depending on whether or not the event is relevant for the reasoning process. If the event enforced by an action $\mathcal{A}$ is represented in the decision model, then $\mathcal{A}$ is referred to as an intervening action, in the other case it is referred to as a non intervening action.
Based on the decision theoretic foundation follows a description of Decision networks that can be considered as extension of Bayesian networks [Paw04]. A decision network consists of *chance nodes*, *action nodes* and *utility nodes*. Chance nodes represent random variables as introduced for BNs. Each chance node has an associated CPT, giving the probability of the variable. Parent

nodes can be action nodes as well as other chance nodes. Action nodes represent the decision being made at a particular point in time. An action node is associated with the set $\mathcal{A}$ of possible actions, so that the values are the actions that the decision maker must choose between. The parents of the node are evidence variables or information variables. These variables are known and can be observed by the decision maker before acting. In a decision network representing a single decision, only one action node for an isolated decision is required. In a sequential decision process, the action nodes can have other action nodes as parents indicating the order of decision. Utility nodes also known as value nodes are associated with a utility function $\mathcal{U} : pa(U) \rightarrow \mathbb{R}$, where $pa(U)$ denotes the parents of U describing the outcome state that directly affect the utility and may include action nodes. A utility node has an associated utility table concerning a single entry for each possible instantiation of its parents without having children nodes.

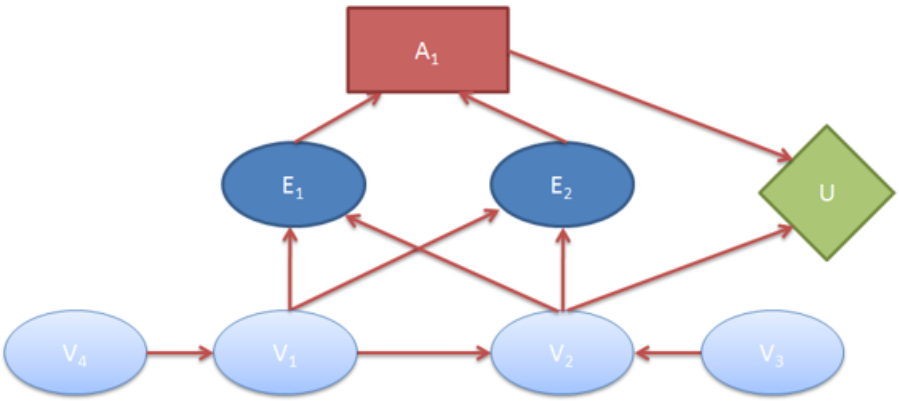


Figure 5.1.: Decision network as extension of Bayesian networks

Figure 5.1 shows a decision network with six involved random variables, two of which are evidence variables as E_1 and E_2. The utility depends on the decision and the value of the variable V_2 which cannot be observed directly. As can be seen, the evidence variables play the role of the information in the context of expected utility theory, and the other random variables correspond to the world state $s \in \mathcal{S}$. In order to compute the expected utility of an action $a \in \mathcal{A}$, the action node is instantiated with that action. Moreover, the evidence variables are instantiated with the corresponding observations. Then, algorithms for BNs are used in order to compute a probability distribution over the random variables which are parents of the utility node. The expected utility $EU(a)$ can be derived on the basis of this distribution. After having performed this procedure for all $a \in \mathcal{A}$, an optimal action is chosen according to the maximum expected utility criterion.

Example 5.1. Bet Agriculture Certificate: Agriculture warrants are intended to capture the performance of certain commodities in the agriculture sector via a notional investment in future contracts. Future markets in this context are amongst others corn, wheat or soybean. An investor *Axan* offers an investor *Bknow* a friendly bet. Both will invest a fix sum in an agriculture

index commodity constructed by taking exposure in its market to a future market in a given tenor. The investor with less profit must invite the other out to dinner limited to 85,- EUR. When deciding whether to accept $Axan's$ bet, $Bknow$ will have to assess his chances of winning which will vary according to the weather for agriculture investments. Under utility aspects, $Bknow$ will be happy in the case of winning and be profitable regardless of the bet.

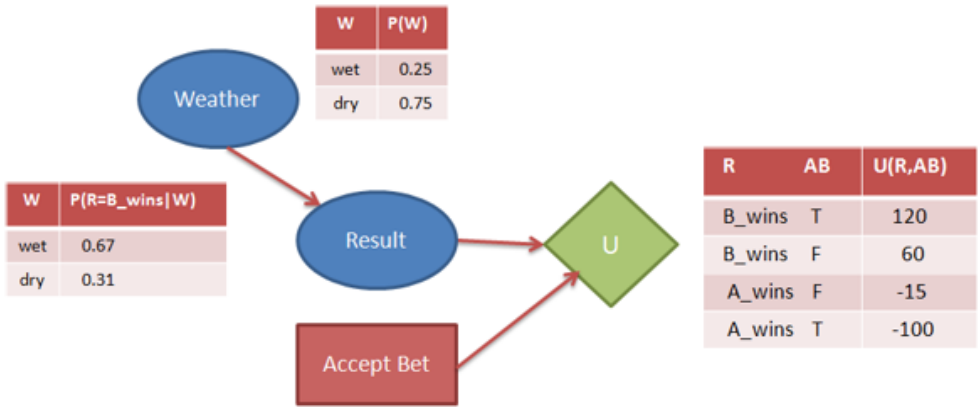


Figure 5.2.: Decision network for the agriculture bet example

The decision network in Figure 5.2 shows the agriculture bet situation. The $Result$ node represents whether investor $Bknow$ wins or loses, the node $Weather$ represents whether it rains or not when the bet is running. The decision node $AcceptBet$ is binary, the utility node U measures the investor's level of satisfaction. Note, that the decision node does not affect any of the variables being modeled. Separate the general process to evaluate a decision network for single decisions as follows:

1. Add any available evidence.

2. For each possible value of the action node set the action node to that value, calculate the posterior probabilities for the parent nodes of the utility node as for BNs using probabilistic inference and calculate the resulting expected utility for the action.

3. Return the maximum action associated with the highest expected utility value.

The expected utility whether or not the investor $Bknow$ bets in the agriculture scenario can be calculated as

$$EU(AB = T) = P(R = B\_wins) \times U(R = B\_wins|AB = T) +$$
$$P(R = A\_wins) \times U(R = A\_wins|AB = T) =$$
$$(0.25 \times 0.67 + 0.75 \times 0.31) \times 120 + (0.25 \times 0.33 + 0.75 \times 0.69) \times -100 = -12$$

and

$$EU(AB = F) = P(R = B\_wins) \times U(R = B\_wins|AB = F) + \\ P(R = A\_wins) \times U(R = A\_wins|AB = F) = \\ (0.25 \times 0.67 + 0.75 \times 0.31) \times 60 + (0.25 \times 0.33 + 0.75 \times 0.69) \times -15 = 15.$$

The probability of the outcome of investor *Bknow* to make profit with the agriculture investment is independent of the betting decision. Investor *Bknow* will not accept the bet without additional available information. Expand the previous example in dealing with observations. Arcs from chance nodes to action nodes are *information links* that indicate, that a chance node needs to be observed before a decision D is made after any decision before this decision D. Calculate explicitly what decision could be made given different values for that chance node. The previous example network can be extended by a *Forecast* node representing the current weather forecast and act as parent node of the action node *AcceptBet*. The expected utility can be calculated by evaluating the network for each evidence case. The calculation for one action and a state s, a decision D, utility functions U_i over domains X_i with $i = 1, ..., n$ and evidence e can generally be declared as

$$EU(D|e) = \sum_{X_1} U_1(X_1)P(X_1|D, e) + \cdots + \sum_{X_n} U_1(X_n)P(X_n|D, e).$$

Maximizing $EU(D|e)$ is chosen as an optimal solution for an agent dealing with a single decision problem.

5.2. Sequential Decision Making

In planning situations it is essential to construct a plan to achieve a goal. A problem solver should appoint the representation of actions, states, goals and plans. Actions generate successor state descriptions, which are used for successor generation, heuristic function evaluation, and goal testing [RN03]. As extension of single decision problems sequences of decisions must be handled by a decision maker in real-life situations. Making an observation or running a test will provide useful information before deciding what further action to take. In a planning situation, the set of discrete random variables and decision variables are subjected to a partial ordering. The random variables are partitioned into disjoint information sets $\mathcal{I}_0, ..., \mathcal{I}_n$ with $\mathcal{I}_i \cap \mathcal{I}_j = \emptyset$ for $i \neq j$ relative to the decision variables specifying the precedence order. The partition includes a partial ordering $\succ$ on the random variables X. The set of variables observed between decisions D_i and D_{i+1} precedes D_{i+1} and succeeds D_i in the ordering $\mathcal{I}_0 \succ D_1 \succ \mathcal{I}_1 \succ \cdots D_n \succ \mathcal{I}_n$, where $\mathcal{I}_0$ is the set of random variables observed before the first decision, $\mathcal{I}_i$ is the set of random variables observed after making decision D_i before making decision D_{i+1} for all $i = 1, ..., n - 1$, and $\mathcal{I}_n$ is the set of random variables never observed or observed after the last decision has been made.

Example 5.2. Purchase a used car : A family in Frankfurt is thinking about buying a second middle class car. While the envisaged car looks fine at first glance, they know that there may be problems with the engine, actuator or hidden not immediately obvious rust. They estimate that there is a 80% chance purchasing a car in good condition with a 20% chance that it could be a dud. They plan to resell the car after doing minor repairs. In the case that the car is in a good condition, they should make 1000,- EUR profit, elsewise, they will lose 500,- EUR. They know a surveyor for which they should pay 250,- EUR. The family cannot be sure, that the report from the surveyor is accurate and have to decide whether it is worth to have the inspection done and then decide to buy the middle class car.

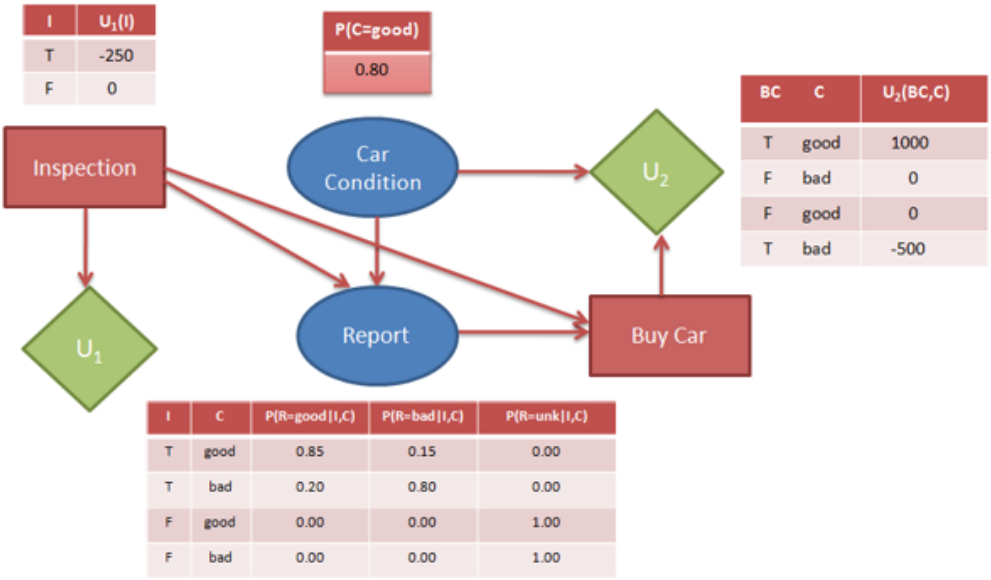


Figure 5.3.: Decision network for the used middle class car example

The used car scenario illustrated in Figure 5.3 induces the following partial order: $\{\} \succ Inspection \succ Report \succ BuyCar \succ CarCondition$. No observations are made prior to the decision on whether or not to fulfill the *Inspection*. After the *Inspection* and before deciding whether or not to purchase the car, the family will make an observation of the *Report*, i.e. the *Report* is available before the decision to purchase the car. After buying the car the real condition is observed.
We could see more than one decision variable in the previous example. In general, with decision variables X_D we have to identify an optimal strategy Δ_{opt} over X_D to maximize the expected utility MEU(Δ_{opt}) of Δ_{opt}. A *strategy* Δ is an ordered set of decision policies including one decision policy for each decision $D \in X_D$. An optimal strategy Δ_{opt} maximizes the expected utility over all possible strategies, if it satisfies $EU(\Delta_{opt}) \geq EU(\Delta)$ for all strategies Δ. A policy for a specific decision D specifies the optimal action for the decision maker for all possible observations made prior to making decision D.

To sum up, a discrete decision network was introduced in this Chapter as model for reasoning and decision making under uncertainty. As component summary a decision network is a BN augmented with decision variables, action variables, and preference relations. The question is how to determine and learn preferences.

5.3. Modeling and Learning of Preferences

Preference modeling is a fundamental step of multi-criteria decision making, operations research, social choice and voting procedures. The preferences of an individual, for instance a user of an operating system [HBH98], a customer of an electronic store [Rie00], or a patient requiring medical care [Cou98], can be expressed in various ways, either explicitly, e.g. in the form of preference statements or implicitly via revealed preferences, e.g. through choices made in different situations. The problem of finding out about an individual's preferences, or about those of a group of individuals, is referred to as preference elicitation. This requires both models for the formal representation of preferences and methods for the automatic data-driven acquisition of such models. Subsequently, discuss two main fields for preference structures, namely preference modeling and preference learning.
These are based on the possibilities for expressing preferences by evaluating individual alternatives or by comparing pairs of competing alternatives. The most basic concept of preference modeling is that of preference structures. Consider a set of alternatives $\mathcal{A}$ and suppose that a decision maker wants to judge them by pairwise comparison. Given two alternatives, the decision maker can act in one of the following three ways:

- the decision maker prefers one to the other
- the two alternatives are indifferent for the decision maker
- the decision maker is unable to compare the two alternatives

According to these cases, three binary relations can be defined in $\mathcal{A}$: the strict preference relation $\mathcal{P}$, the indifference relation $\mathcal{I}$ and the incomparability relation $\mathcal{J}$. The basic relation, often denoted $a \succeq b$ or $\mathcal{R}(a, b)$, is usually interpreted as alternative a is at least as good as alternative b. The reflexive relation $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$, where $\mathcal{A}$ is the set of alternatives, induces a strict preference relation $\mathcal{P}$, an indifference relation $\mathcal{I}$ and an incomparability relation $\mathcal{J}$ in a straightforward way. A triple $(\mathcal{P}, \mathcal{I}, \mathcal{J})$ with an asymmetric relation $\mathcal{P}$, a reflexive and symmetric relation $\mathcal{I}$ and a symmetric relation $\mathcal{J}$ is referred to as a *preference structure*. Formally, the triple $(\mathcal{P}, \mathcal{I}, \mathcal{J})$ must also satisfy the following:

- $\mathcal{P} \cap \mathcal{I} = \emptyset$
- $\mathcal{P} \cap \mathcal{J} = \emptyset$
- $\mathcal{I} \cap \mathcal{J} = \emptyset$

- $\mathcal{P} \cup \mathcal{I} \cup \mathcal{J} = \mathcal{A} \times \mathcal{A}$

In this context, binary relations $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ are considered and may be represented as functions $\mathcal{R} : \mathcal{A} \times \mathcal{A} \to \{0, 1\}$. A triplet $(\mathcal{P}, \mathcal{I}, \mathcal{J})$ of binary relations in $\mathcal{A}$ is a preference structure on $\mathcal{A}$ if and only if [dBvdWK97]:

- (a) $\mathcal{I}$ is reflexive or $\mathcal{J}$ is irreflexive
- (b) $\mathcal{I}$ is symmetrical or $\mathcal{J}$ is symmetrical
- (c) $(\forall (a, b) \in \mathcal{A}^2)(\mathcal{P}(a, b) + \mathcal{P}(b, a) + \mathcal{I}(a, b) + \mathcal{J}(a, b) = 1)$.

The second approach is more inclined to numerical representations of preferences via *utility* or *value functions*. A utility function u assigns an abstract utility degree to each alternative under consideration and thus induces a preference relation $\succeq$ by virtue of $\mathcal{R}(a, b) \Leftrightarrow u(a) \geq u(b)$. The relationship between the two approaches is a focus of research in the field of multi-attribute utility theory. A question of interest concerns, e.g., the characterization of preference structures that can be represented by utility functions have a certain mathematical structure. Of special interest are utility functions that can be decomposed into additive sub-utility functions.
Obviously, the numerical approach provides stronger information than the relational one but is also more restrictive and more demanding from a modeling point of view. In fact, it is usually much easier for people to provide relative or comparative preference information of the form "I prefer A to B" than absolute information of the type "I like A to the degree 0.86", requiring the specification of a utility degree (score) for each product. Moreover, *revealed* preferences are usually of the relational type. Consider, for instance, a person in a situation where a choice between two products A and B must be made. If that person chooses A and not B, it is at least likely that the person prefers A to B. Formally, this gives rise to the information $\mathcal{R}(a, b)$, whereas nothing is known about the absolute utility degree from products A and B.
Advances in both approaches to preference modeling are possible through concepts and tools from fuzzy set theory [Zad65]. The main contribution of fuzzy sets is a unifying framework for handling different types of *incomplete* and *imprecise* knowledge, a point of particular importance in preference modeling. Subsequently, a presentation of two types of fuzzy preference models will be presented which appears to be very interesting in the context of personalization and recommendation, namely fuzzy preference structures and prioritized fuzzy constraints.
People are often inconsistent in their judgments: a person may at one time say $(a, b) \in \mathcal{R}$, and at another time $(b, a) \in \mathcal{R}$. Although a person might be inconsistent in this sense, it is rather acceptable that the person might be probabilistically consistent in the following sense. Let Q be an $\mathcal{A}^2 \to [0, 1]$ mapping such that $Q(a, b)$ is the proportion of times when a person prefers a to b. Probabilistic consistency means that $Q(a, b) + Q(b, a) = 1$ for any $a \neq b$. In that case, (A, Q) is called a pair comparison system. With the advent of fuzzy set theory, and the immense impact of it, the whole theory of (binary) fuzzy relations has become available [dBK94]. Because of the above mentioned inconsistency and

the fact that fuzzy relations allow to express degrees of preference, indifference or incomparability, it is very natural that fuzzy relations have been strongly involved in preference models.

Fuzzy preference structures extend the classical approach by replacing the binary relation $\mathcal{R}$ with a fuzzy relation. In that case, $\mathcal{R}(a, b)$ is no longer forced to be either 0 (which means that $a \not\succeq b$) or 1 (which means that $a \succeq b$) but can take any value in the unit interval $[0, 1]$ or, more generally, some linearly ordered scale. The value $\mathcal{R}(a, b)$ can be interpreted in different ways, e.g. as the strength of the *at least as good* relation between a and b or as a degree of uncertainty. An associated fuzzy preference structure is a triple $(\mathcal{P}, \mathcal{I}, \mathcal{J})$ of fuzzy relations having certain mathematical properties which are generalizations of the properties required in the non-fuzzy case. Fuzzy preference structures and their axiomatic construction is studied for the interested reader in Perny & Roy [PR92] or Fodor & Roubens [FR94b]. Fuzzy preference structures help to overcome several limitations of the classical approach, especially observability and informational problems. In fact, more often than not knowledge about the preferences of an individual will be incomplete, imprecise or uncertain. Especially, this will be true if such preferences are induced indirectly, through observing an individual's choice behaviour. But even direct statements about preferences, expressed in terms of natural language, will often be ambiguous. In fact, people usually tend to qualify their preference statements, rather than simply saying "yes" or "no". In all these cases, the fuzzy approach will help to make formal models more adequate and realistic. We might even go one step further by valuating a preference through intervals [Bil98] or probability distributions rather than using a single number.

Fuzzy constraints share important similarities with the numerical approach to preference modeling. In much the same way as utility functions, such constraints can be used for expressing degrees of preference or satisfaction. The fuzzy approach allows for a purely qualitative setting, where the underlying preference scale consists of an ordered set of labels such as "good" or "very good". In the multi-criteria case, individual degrees of preference are then combined through aggregation functions more generally than arithmetic operations [BM03]. The qualitative setting appears advantageous from an application point of view, since expressing preferences on a finite natural language scale will usually be much simpler for people than providing precise numbers. *Prioritized* fuzzy constraints extend fuzzy constraints by the possibility of assigning a degree of priority or importance to individual constraints. In the common constraint satisfaction framework, the individual constraints which can pertain to different aspects of an alternative are combined by means of a conjunction operator. The overall degree of satisfaction can in principle be increased by increasing the satisfaction of any individual constraint. In the prioritized setting, the increase of a constraint with a relatively low priority is helpful only if the constraints with a high priority are already sufficiently satisfied [LLLJ03]. The topic of preferences has also attracted considerable attention in artificial intelligence, especially in fields such as non-monotonic reasoning, constraint satisfaction, planning and qualitative decision making [BDS04]. Artificial Intelligence offers qualitative and symbolic methods for treating preferences that can

complement or even improve already existing approaches reasonably. Thus, every expressive preference representation might be developed based on concepts from logic or constraints, as well as tools for reasoning and decision making with preferences.
Consider for instance an ignorance situation in which the decision maker must apply a strategy to handle missing values. Assume indifference values in the missing values, because an expert does not provide information on an alternative relating it to the rest of alternatives. Model this situation as a total indifference one and therefore each missing value for the ignored alternative can be replaced by 0.5. We have to solve a decision making problem to find the best of four different alternatives $X = \{x_1, x_2, x_3, x_4\}$. An expert gives the incomplete fuzzy preference relation being indifferent with respect to x_3 (see Figure 5.4). If an incomplete fuzzy preference relation has an ignored alterna-

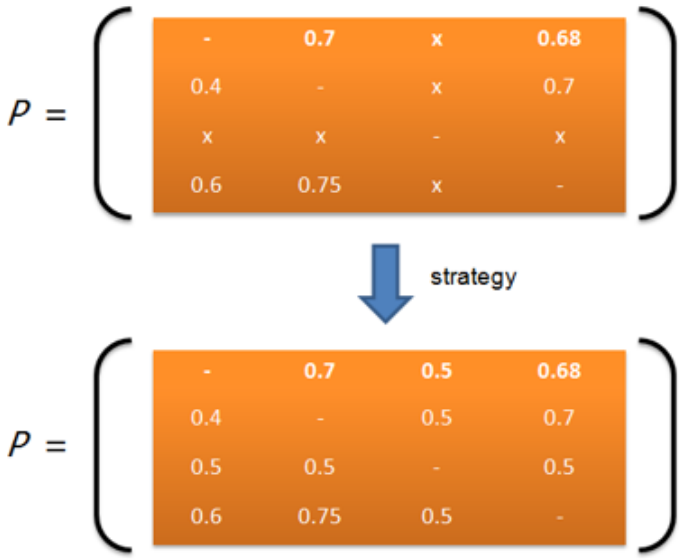


Figure 5.4.: Strategy to manage an incomplete fuzzy preference relation

tive x_i, this strategy computes all its associated missing values with $p = 0.5$. Another strategy estimates the missing values for an ignored alternative as random values within the range of preference values provided by the expert. An unknown preference value will be calculated randomly between the minimum and maximum preference degrees of its corresponding column and row (e.g. $p_{13} \in [0.68, 0.7]$).

The term preference elicitation usually refers to the problem of finding out about the preferences of a single individual. As a simple example consider approximating the utility function of a person by asking for several utility degrees and then interpolating between them. Such problems are considered e.g. in economic utility theory. Another goal is to predict the preferences of an individual on the basis of certain properties of that individual and known preferences of other individuals. Consider a salesman who knows from experience that "middle-aged, working, childless men usually prefer product A to product B to

product C". The salesman has learned from experience to predict preferences of clients on the basis of these clients' features. The example can be seen as an extension of supervised machine learning, with the known preferences playing the role of examples. This type of problem involves the prediction of preference structures in the relational approach or value functions in the utility-based approach. There exist two approaches that closely fit the aforementioned idea of inducing a preference function mapping individuals to preference structures. Distinguish a model-based approach and a case-based strategy. A framework of constraint classification as extension of standard classification problems is illustrated in [HPRZ02]. More specifically, constraint classification means learning a function which maps an instance space $\mathcal{X}$ into the class of partial orders on a set of labels $\mathcal{L}$. With c-constraint classification, the output is restricted to partial orders $\mathcal{R} \subseteq \mathcal{L} \times \mathcal{L}$ of size $|\mathcal{R}| \leq c$. Ha and Haddawy [HH03] developed a similarity measure for (partial) preference structures. This probabilistic distance allows to deal with partially missing preference information (see also Figure 5.4). Moreover, it can be extended to the case of uncertainty, where individuals must choose among probability distributions over alternatives rather than over alternatives directly. It is possible to use a case-based approach [VR93], where the partially known preferences of a user are extended by ascribing to him the preferences of the most similar one among all other users.
It is also interesting to rank-order objects on the basis of preference information of different type. Consider an approach for ranking a set of objects given feedback in the form of preference judgments from different sources (e.g. ranking of documents from different search engines). An approximation of a binary preference relation over the objects can be induced, whereas a heuristic optimization procedure can be used in order to find a ranking of the objects that is maximally compatible with the preference relation. Given training data available in the form of pairwise comparisons of objects, it is possible to train an artificial neural network that takes as input two objects and returns either 0 or 1, depending on whether or not the first object is preferred to the second one. It is also interesting to analyze click-through data in the context of ranking documents retrieved by a search engine according to their relevance. Using this kind of indirect preference information, learning of a retrieval function is accomplished by training a support vector machine.

As concerns the utility-based approach, learning a value or utility function comes down to inducing a mapping $\mathcal{A}_1 \times \cdots \times \mathcal{A}_n \rightarrow \mathbb{R}$ if alternatives are specified in terms of attribute values $a_i \in \mathcal{A}_i, 1 \leq i \leq n$, and if the utility of such alternatives is measured on the real number line. This problem can be approached by means of techniques from statistical regression analysis and function approximation. As an aside, note that in decision theory a utility function is usually not defined on the set $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ of alternatives directly but rather on the set of probability distributions as *lotteries* over $\mathcal{A}$. The problem of learning eliciting real-valued utility functions has been investigated in fields such as decision theory and economics for a long time and has become a topic of research in AI and machine learning as well [CKO01]. If utility is measured on an ordinal scale, the problem of inducing a value function comes down to one of

ordinal regression. Methods for learning ordinal regression functions have been proposed in [PB00]. Here, the value functions are implemented as classification trees whose leaf nodes are labeled with utility degrees.
The development of learning methods for fuzzy preference models and prioritized fuzzy constraints can close a gap in this research field. Reasons for focussing on fuzzy preference models are motivated by the fact that preference information is usually vague, imprecise or uncertain. Moreover, fuzzy models subsume classical models as a special case. The restriction on prioritized fuzzy constraints is justified by the fact that these models appear particularly suitable for applications in personalization. A preference learning framework should distinguish between the representation of individuals, the representation of alternatives, the underlying type of preference structure, the preference information and the preference model to be learned. The framework should comprise a kind of taxonomy for preference learning problems, which is necessary for developing learning methods in a systematic way. It is also important to have a criteria catalogue for measuring the performance of preference learning methods. Methods for the learning of preference models should extend the pairwise preference learning more general than rankings. Another research topic is to develop an instance-based approach to preference learning to derive an overall ranking from the rankings suggested by the given examples. Finally, since preference information is derived from statements or actions of humans, the aspect of imperfect data becomes particularly relevant in the context of preference learning. Addressed questions include how to model different types of imperfect information and how to extend the learning methods to make them approachable to imperfect data.

To classify preferences and preference structures consider in general a decision problem which consists of the parts of values representing symptoms or observations, actions and possible consequences. It is possible to order the consequences by preference by using a utility function $U(\cdot)$. Hence we can choose the action that will lead to the preferred result based on some decision criteria such as least risk or optimistic estimation. As demonstration for the practical relevance of preferences the decision problem represented in Figure 5.5 is employed. The possible actions are performing experiments, the values are the results of these, and the consequences are the relative utilities of the experiment. The problem is how to learn a causal BN as DN from a mixture of observational and experimental data [MC04]. The directed edges are representing autonomous causal relations among the corresponding variables, while in a BN the directed edges represent a probabilistic dependency and not necessarily a causal one.
In order to find the causal relation between two variables X and Y we have to check whether randomizing X by holding all other variables fixed at a certain value induces a variation in Y or vice versa. Performing an experiment at X will give us information on the directionality of all undirected edges connected to X [MLM06]. On the basis of a decision theoretic approach the DN is learned. Assume that the values (e.g. symptoms), actions and possible consequences are given in advance. By using a utility function it is possible to order the consequences by preference. The decision problem is illustrated in Figure 5.5, in

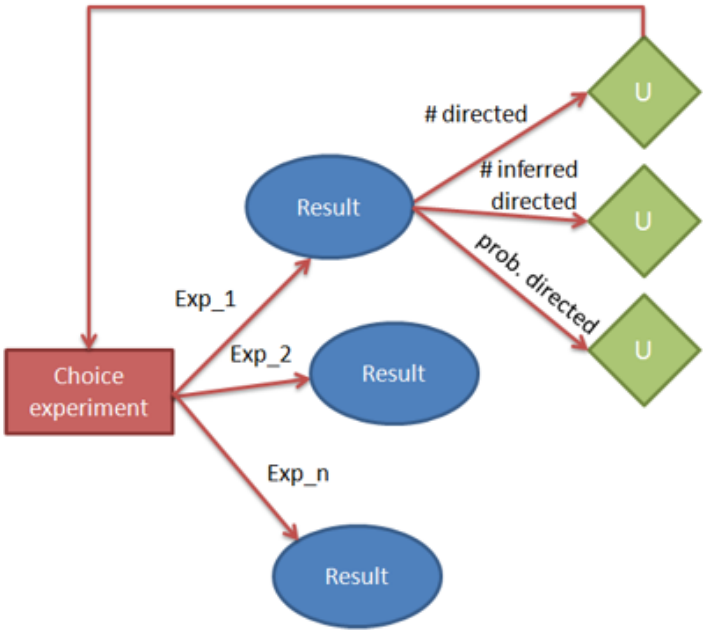


Figure 5.5.: Decision problem of learning a causal Bayesian network

which the possible actions are performing experiments, the values stand for the results of these, and the consequences are the relative utilities of the experiment. Since the problem is iterative, a decision can be dependent on the choice of a previous one.
A utility function $U(\cdot)$ will be a function with three variables $gain(exp)$, $cost(exp)$, $cost(measure)$, respectively the gained information, the cost of performing an experiment and the cost of measuring other variables. Denote performing an action as experiment at X_i by A_{X_i}, and measuring the neighboring variables by M_{X_i}, then the utility function can be declared as $U(A_{X_i}) = f(gain(A_{X_i}), cost(A_{X_i}), cost(M_{X_i}))$. Assume that we perform an experiment on X_i and measure all neighboring variables $Ne(X_i)$. In this case connect all links X_i and $Ne(X_i)$ directly as a result of the experiment. In this case the gain (A_{X_i}) is based entirely on the number of variables that are connected to X_i by an undirected arc. It is possible that directing one arc infers direction of other arcs and to take into account the possibility of inferred edges in gain(A_{X_i}). The amount of edges of which the direction can be inferred after performing an experiment is entirely based on the instantiation of the undirected edges connected to the one being experimented on. It is crucial to choose the appropriate decision criteria for the introduced learning problem. Depending on the type of situation in which to perform the experiments it might be advantageous to choose a specific criterion. The *maximax* decision criterion is an optimistic one, which means to choose the action that could give the best result, i.e. the one that might direct the most arrows. Using the *Laplace* criterion means that the decision maker assumes that all directions of edges are equally probable. The expected utility is based on a distribution of the directions of the links. Based on this distribution it is possible to calculate the probability of any instantia-

tion of directions that might occur. As summary, allow the possibility to add newly discovered knowledge due to the experiments during the learning phase. Since the experiments are performed, we gain information on the direction of certain links. This information may remove the need to perform certain other experiments. Learning DNs is possible with adaptive approaches, in which it is allowed to actively add results of experiments by using any decision criteria or utility function representative to order consequences by preference. This section provides a decision maker preference modeling fundamental notations as well as their applicability in a variety of fields like economy or medicine. The description includes the determination and evaluation of preferences reflecting the behaviour of a decision maker by establishing decision variables and utility functions in a DN. The result is the ability to compare alternatives in different decision situations with incomplete, uncertain or ambiguous information.

5.4. Information Gathering to Facilitate Reasonable Decision Making

The problem of *information overload* has been a topic of concern in science in recent years. The huge amount of information makes it necessary to expose users to the most appropriate and interesting items by applying personalization techniques. To solve the described problem so-called recommendation or recommender systems aim at predicting topics, items, or products a specific person as active user of the system might like [Uch08]. An increasing amount of applications can be found in electronic commerce and information access, with systems recommending web pages, movies, music and so on. As concerns methodological aspects, different information filtering techniques of varying degree of sophistication have been developed, ranging from simple keyword matches over similarity-based filtering to machine learning on heterogeneous information sources. The techniques used mainly depend on the type of information available: features as demographic information of users, features of items, information about user preferences such as ratings of selected items or complete utility functions. This type of surface data can further be completed by deep domain knowledge, consider for example how items can meet a user's needs or interests. The three main types of techniques mostly used for recommender systems are demographic, content-based and collaborative methods. In the first case, personal data of people are used in order to establish a relationship between the type of people and the items they like or dislike. In this connection, clustering methods are often employed in order to create homogeneous groups of people. Moreover, rule induction methods can be used in order to learn rules whose condition part is a logical characterization of people in terms of demographic attributes and whose conclusion part is a recommended item. Content-based methods make use of textual descriptions of items to be recommended and employ methods from both information retrieval and machine learning [ADF02]. Usually, content-based systems analyze the available ratings of an individual user in order to infer a profile of that user. New items are then recommended on the basis of that profile, using some profile-item matching

technique. Roughly, content-based filtering methods recommend items that are similar to other items known to be liked by the user.
Collaborative systems recommend items based on aggregated user ratings. They exploit experience with other users, being available in the form of a database about user actions, votes, or preference patterns. The basic idea of collaborative filtering (CF) is to recommend items that have been chosen by those users whose preferences are likely to coincide at least to some extent with the active user's preferences [GNOT92]. Thus, the principle problem is to predict the active user's preferences on the basis of known preferences of other users. This can be considered as a learning problem that can be approached by methods from machine learning [BP98]. Most commonly applied in CF are memory-based (case-based, instance-based) approaches, a special class of machine learning methods based upon the nearest neighbour estimation principle. The basic idea is to ascribe to the active user the preferences of those users that appear to be similar, where similarity can refer, e.g., to known properties of users or to already revealed preferences. For instance, relying on the assumption that people with similar preferences in the past will have similar interests in the future, a popular measure of similarity between two users is the Pearson correlation coefficient of these users' prior ratings of objects. Alternative measures include the Spearman rank correlation, vector similarity, entropy-based uncertainty measures and the mean squared difference [HKBR99]. Apart from finding similar users, a central problem in case-based recommendation concerns the combination of those users' preferences into a prediction for the active user. As opposed to memory-based approaches, model-based methods aim at inducing a model from the data which is then used for making predictions and recommendations. Once a model has been established, such predictions can usually be derived more efficiently than with the memory-based approach. However, learning a model might of course be computationally complex, even though it can be done offline. Apart from that, passing from the complete data to a model usually brings about a loss of information. Examples for model-based CF methods include the use of BNs [BDK98] or methods based on cluster analysis. Klahold [Kla06] has introduced in his PhD thesis a new procedure for a recommender system different from similar procedures by, first, employing heuristics that link a TF-IDF (term frequency - inverse document frequency) derivate with the properties of the text structure and, second, generating an asymmetrical pre-calculated distance matrix that is independent of any particular language. Any unstructured text may be used as the underlying basis, and no meta-data, thesauri or corpora are required. Further, different premises like user acceptance, fast response time, or omission of manual intervention by the author or user are applied.
All the aforementioned recommendation techniques have particular merits and problems, and their performance will strongly depend on the application at hand. For example, collaborative filtering works best for problems where the user interest is focused on a small and static set of items. This technique usually fails, when the space of ratings is sparse. It is often useful to combine demographic, content-based, and collaborative filtering techniques into hybrid recommenders [Bur02]. This way, the limitations of each individual approach

can be compensated, at least to some extent.

A knowledge engineer dealing with Decision networks needs a framework to gather step by step information that facilitates the decision making process in a specific problem domain by disregarding non-efficient and redundant information [Pre02]. This will lead to a condensed DN, which includes only decision relevant nodes. The basic idea is that the decision making process starts with complete ignorance and moves towards a probability measure until the specified information enables the framework to make a reasonable decision. The decision maker starts with a decision making problem under ignorance and by adding more and more information eventually comes to a decision making problem under risk. A framework supports the decision maker on this way by providing some hints on which information should be supplied and to what precision this information is needed. It is a both crucial and obvious observation that the path does matter. If the decision maker always supplies the wrong piece of information, the path could become extremely long. The approach maintains a belief function Bel which stands for the knowledge of the decision maker or agent. Refer to this belief function by $Bel(i)$ where i denotes the i-th iteration of the main loop of the approach. Starting with an empty set representing no knowledge at all which is represented by the completely empty belief function

$$Bel_{(0)}(X) = \begin{cases} 1 & if X = \Theta \\ 0 & otherwise. \end{cases}$$

Since $\mathbb{P}(Bel_{(0)}) = \{Pr|\ Pr \text{ is a probability measure on the set of states } \mathcal{S}\}$, the belief function $Bel_{(0)}$ really represents complete ignorance. By adding more and more information, the size of the set $\mathbb{P}(Bel_{(i)})$ should be reduced over time. In the i-th iteration of the main loop make the following. First determine the set $\mathcal{A}_{(i)} \subseteq \mathcal{A}$ of actions that might be optimal given the knowledge $Bel_{(i)}$. For any action $a \in \mathcal{A}$ compute the upper expected utility

$$E^*(a) = max_{Pr \in \mathbb{P}(Bel_{(i)})} \sum_{s \in \mathcal{S}} Pr(\{s\}) U(c(a, s))$$

as well as the lower expected utility $E_*(a)$ which is defined analogously. Remember that $U(c(a, s)) \in \mathbb{R}$ is the utility of the outcome $c(a, s)$, where $(a, s) \in \mathcal{A} \times \mathcal{S}$. Then, $\mathcal{A}_i$ is defined to be the set of actions $a \in \mathcal{A}$ such that there is no action $a' \in \mathcal{A}$ satisfying $E_*(a') > E^*(a)$. Since there is such an action a' it would be irrational to chose action a.

Next determine a *best* action $a_i^{best} \in \mathcal{A}_i$ using some decision rule for evidence-based decision making based on the knowledge $Bel_{(i)}$. For example, choose a_i^{best} as proposal for the agent to be the action $a \in \mathcal{A}_i$ that maximizes

$$\rho E^*(a) + (1 - \rho) E_*(a)$$

where the parameter $\rho \in [0, 1]$ has to be chosen by the agent. In addition, the approach computes some data that is used to assist the agent. For example, consider the computation of an *upper bounds loss* for the lost utility, which originates when the agent chooses the action a_i^{best} instead of the optimal but

unknown action a^*. Calculate the loss value based on the agent's knowledge $Bel_{(i)}$ as

$$loss = max_{Pr \in \mathbb{P}(Bel_{(i)}), a \in \mathcal{A}_{(i)}} \sum_{s \in \mathcal{S}} Pr(\{s\}) \Delta(a, s)$$

where

$$\Delta(a, s) = (U(c(a, s)) - U(c(a_i^{best}, s))).$$

If the loss value is small enough, the agent can choose the proposed action a_i^{best}. Remember, that only the agent might have an idea of what this value means. The concrete meaning depends on the definition of the utility function.
Another question is how to find the right pieces of information to supply next. The agent can either stop when the loss value is small enough and then choose the proposed action a_i^{best} or he can add new information. The newly entered information should reduce the amount of ignorance, that is, $\mathbb{P}(Bel_{(i+1)}) \subseteq \mathbb{P}(Bel_{(i)})$ should hold, where $Bel_{(i+1)}$ is the belief function which represents the updated knowledge. If the agent has not stopped the algorithm, the next iteration step $(i + 1)$ of the main loop starts.
An idea to guide the agent in selecting the right pieces of information is based on the basic probability assignment m_i which corresponds to the belief function $Bel_{(i)}$ (see also Section 3.3). Let $X \subseteq \mathcal{S}$ be an arbitrary set satisfying $|X| \geq 2$ and $m_{(i)}(X) > 0$. If there is no such set X then $Bel_{(i)} = Pl_{(i)}$ holds and thus we are facing a decision problem under risk. This means that the weight $m_{(i)}(X)$ assigned to the set X has not yet been distributed to the elements of X. Adding more information leads to distributing the weight $m_{(i)}(X)$ among $X's$ elements. In general, there are many sets X satisfying $|X| \geq 2$ and $m_{(i)}(X) > 0$. Thus the question arises which of these sets X is of particular interest for differentiating between the actions. An approach to cope with this question is to calculate the difference for any set X:

$$diff(X) = m(X)\ max_{a \in \mathcal{A}_{(i)}, s \in X}\ (U(c(a, s)) - U(c(a_i^{best}, s))).$$

The *diff* value is an upper bound for the effect that distributing $m(X)$ among $X's$ elements could have on differentiating between the proposed action a_i^{best} and the remaining actions $a \in \mathcal{A}_{(i)}$. It is reasonable to choose a set X with a high *diff* value and then try to distribute $m(X)$ among the elements or among the subsets of X.
The algorithm introduced by Presser [Pre02] solves the question in how to find the right path. Given a decision problem $(\mathcal{A}, \mathcal{W}) \in \mathbb{E}$ a decision maker chooses an alternative $a \in \mathcal{A}$. It exists a *true* probability distribution $\rho \in \mathcal{W}$ inaccessible for the decision maker with an optimal alternative a^*, which maximizes the expected utility $a^* \in \arg max_{a \in \mathcal{A}} EU(a)$. The decision maker's loss value *loss* represents a quality mass to estimate the made decision a related to the decision problem $(\mathcal{A}, \mathcal{W}) \in \mathbb{E}$. The linear probability information $W \subseteq \mathbb{S}^{(n)}$ with

$$\mathbb{S}^{(n)} = \{p = (p_1, ..., p_n) \in [0, 1]^n | \sum_{i=1}^{n} p_i = 1\}$$

can be expressed as

$$W = \{p \in \mathbb{R}^n | Cp \leq b\}.$$

For instance, $W_1 = \{(1,0)\}$ and $W_2 = \{(0,1)\}$ are linear probability information. A decision maker should state more precisely such constraint with highest influence to the *loss* value. Starting point of the algorithm is a decision prob-

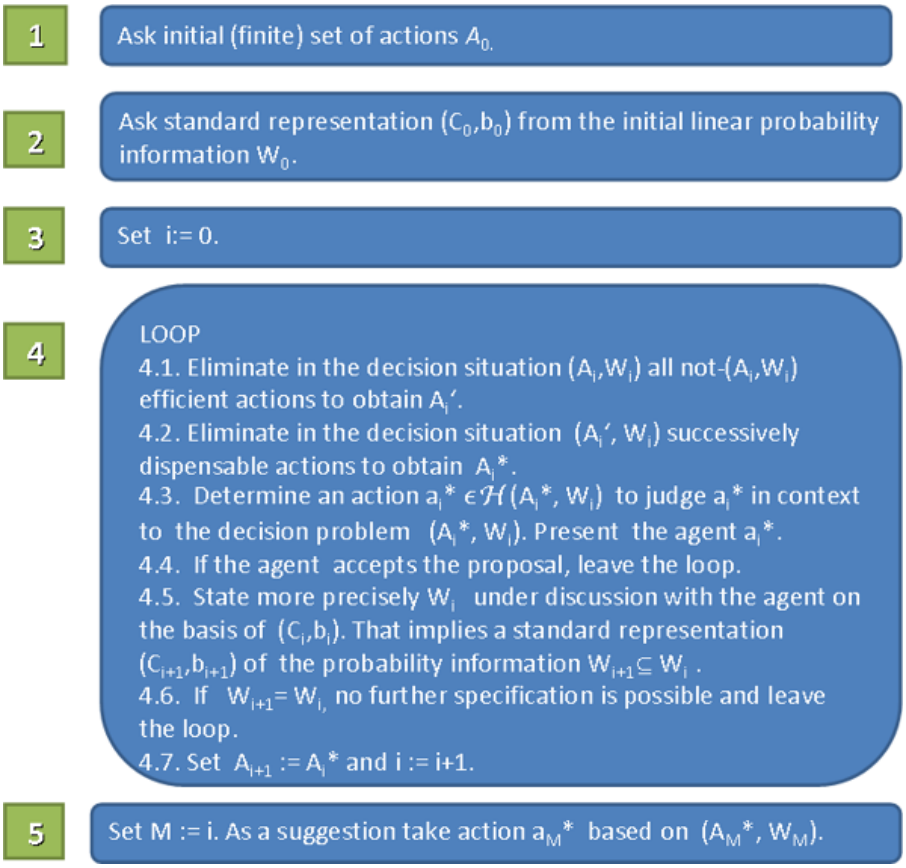


Figure 5.6.: Decision making algorithm to specify the standard information $\mathcal{W}$

lem $(\mathcal{A}_0, \mathcal{W}_0) \in \mathbb{E}$ with partial probability information. The set of actions $\mathcal{A}$ and probability information represented as standard form are inquired by the agent. The decision problem specification occurs in the 4th step. We eliminate all not-$(\mathcal{A}_i, \mathcal{W}_i)$ efficient actions. Action a' is called $(\mathcal{A}_i, \mathcal{W}_i)$-efficient, if $\forall a \in \mathcal{A} \quad \neg(a \succ_{\mathcal{W}} a')$. There exists by definition an action a which dominates action a'. The agent prefers action a' in this situation. The resulting set $\mathcal{A}_i'$ represents all $(\mathcal{A}_i, \mathcal{W}_i)$ efficient actions. In the following step, eliminate all dispensable actions, which are also $(\mathcal{A}_i, \mathcal{W}_i)$ efficient actions, but there exists just as well another preferable action. Action a' is called $(\mathcal{A}_i, \mathcal{W}_i)$-dispensable, if $\exists a \in \mathcal{A} \backslash \{a'\} \quad (a \succeq_{\mathcal{W}} a')$. The agent determines another action a in such a way that action a' is not preferable. The elimination of action a' does not limit the alternatives of the decision problem $(\mathcal{A}_i, \mathcal{W}_i)$. A decision maker has to be able to reduce the number of actions which are conceivable as solution. Next, a proposal is submitted on the basis of a heuristic like $MaxE_{min}$. In this step,

a proposal $a_i^* \in \mathcal{H}(\mathcal{A}_i^*, \mathcal{W}_i)$ is calculated concerning the current probability information $\mathcal{W}_i$ and the *loss*-value. The agent can accept action a_i^* and leave the loop or follows up. In the latter case further specification regarding the probability information is necessary. The agent adds further constraints and specifies the representation (C_i, b_i). If a specification is fulfilled, we obtain (C_{i+1}, b_{i+1}) with $\mathcal{W}_{i+1} \subseteq \mathcal{W}_i$ and the next iteration step follows. To sum up, an agent who applies the algorithm which is illustrated in Figure 5.6, generates a sequence of decision problems $(\mathcal{A}_0, \mathcal{W}_0), ..., (\mathcal{A}_M, \mathcal{W}_M) \in \mathbb{E}$ with $\mathcal{W}_0 \supseteq \mathcal{W}_1 \supseteq ... \supseteq \mathcal{W}_M$. Finally, as a suggestion take action a_M^* including the most precise problem specification.

Example 5.3. Lazy Decision Making: It is often unnecessary to declare the exact probability distribution in a specific decision situation. Abundant information is added in decision situations under risk. Consider the decision scenario $(\mathcal{A}, \mathcal{W}) \in \mathbb{E}$ with $\mathcal{A} = \{a_1, a_2, a_3\}$, whereby $a_1 = (5, 5, 5), a_2 = (10, 5, 0), a_3 = (3, 3, 10)$. First of all, a decision scenario under risk is discussed with $\mathcal{W} = \{p\}$ and $p = (p_1, p_2, p_3) = (0.6, 0.2, 0.2)$. Taking action a_2 delivers the highest expected utility $EU(a_2) = 7.0$. Giving exact probability values is unnecessary from the agent's point of view. The expected utility concerning a_1 is completely independent of $\mathcal{W}$. Calculate $EU(a_2)$ with $10p_1 + 5p_2$. Awareness of $p_1 \geq 0.5$ is sufficient to exclude a_1 as optimal action an agent has to choose. The information $p_3 \leq 0.25$ results immediately in a_2 as optimal choice. The imprecise information of p_1 and p_3 covers everything in this decision situation $(\mathcal{A}, \mathcal{W}) \in \mathbb{E}$.

An agent starts in a decision situation with completely empty belief and therefore cannot add non-trivial information as specification. Successive addition of broad constraints leads to purposeful specification of probability information an agent declares as conditional probability. For instance, an agent describes the initial probability information as $Pr(\{1, 2\}|\{1, 2, 3, 4, 5\}) \leq 0.3$. We obtain $Pr(\{1, 2\}|\{1, 2, 3, 4, 5\}) = (p_1 + p_2)/(p_1 + p_2 + p_3 + p_4 + p_5)$ and due to this $p_1 + p_2 \leq 0.3(p_1 + p_2 + p_3 + p_4 + p_5)$ as linear inequation. The agent chooses an appropriate representation for a clearer understanding and accentuation of the constraints.
Consolidated, there are many application fields in which agents cooperate and the precision of many pieces of information can be improved, but the price for this improvement can be rather high.
Consider a knowledge engineer who has to combine diverse techniques to generate Bayesian networks and their extension augmented with decision variables and utility functions. A Knowledge engineering process allows first the construction of Bayesian models under a variety of circumstances named *Knowledge Engineering with Bayesian Networks (KEBN)*. Korb and Nicholson [KN04] explain the KEBN process as lifecycle model including the network building process, validation, testing, industrial usage and refinement (see Figure 5.7). The author of this thesis has separately marked all new research approaches in Figure 5.7 as violet boxes. In the first network construction phase, the major components and parameters must be determined through elicitation from experts or by learning the structure given fully or partially observable data. In

the validation phase a sensitivity analysis looks at how sensitive the network is to changes in dedicated input and parameter values, which can be useful for validating that the network is correct and second for understanding how best to use the network in practice in the field. Testing puts the network into actual use allowing its usability and performance to be gauged. Model experts, structure experts and system end-users are appropriate and deployed in early release phases to detect hidden bugs.

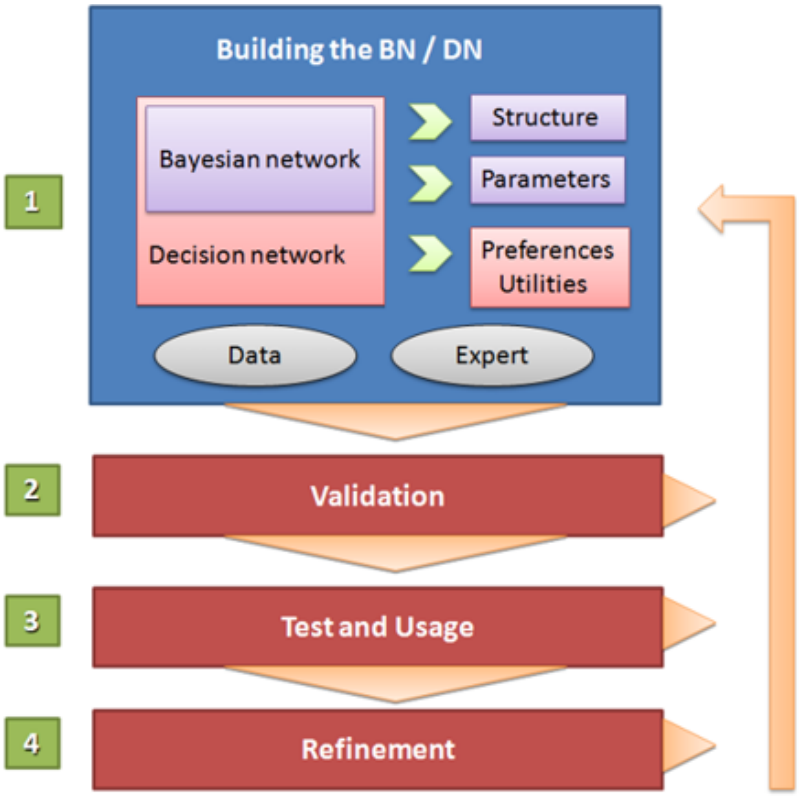


Figure 5.7.: Bayesian and Decision network knowledge engineering process

From the *knowledge transfer* point of view it is required to achieve the end users' acceptance that the graphical model solution meets their criteria in the defined problem domain for usage and to come into operation contemporarily. The industrial use sees the graphical system in regular use in the field and requires that procedures be put in place for the continued use. Statistics monitoring in the application domain can help to further validate and refine the network. The refinement process step requires some kind of change management to deal with requests for enhancements or fixing bugs. Regression tests confirm that any changes do not cause a deterioration of prior performance. Condensed, the initial process step in building the network is a crucial point. When attempting to model large and complex domains it is important to limit the number of variables representing evidence, hidden or query nodes. For the incorporation of actions and utilities it is important to combine beliefs and desires under uncertainty. The preferences captured by utility functions can be

learned from data based on constraints on rational preferences. By evidence-based decision making it is possible to move from an ignorance decision situation towards a probability measure until the specified information enables to make a reasonable decision.

5.5. Decision Networks Review

We have introduced probabilistic networks for reasoning and decision making under uncertainty. A probabilistic network represents and processes probabilistic knowledge which is distinguishable by a qualitative and quantitative component. The qualitative component encodes a set of (conditional) dependence and independence statements among a set of random variables, informational precedence and preference relations. The quantitative component specifies the strengths of dependence relations using probability theory and preference relations using utility theory. A DN is a BN augmented with decision variables, informational precedence relations and preference relations. We focussed our attention on discrete decision networks, which support the use of discrete random and decision variables with an additively decomposing utility function. By adding a temporal dimension to BNs, we get dynamic DNs which allow us explicitly to model and reason about changes over time. DNs provide a language for sequential decision problems for a decision maker, where there is a fixed order among the decisions. Solving a decision problem amounts to determine a strategy that maximizes the expected utility for the decision maker and compute the maximal expected utility. Decision making under uncertainty is the task of identifying the optimal decision strategy for the decision maker given observations. We must be able to take into account preferences between different outcomes. Utility theory provides a way to represent and reason with preferences. A rational decision maker should make choices that maximize the expected utility. Preference elicitation refers to the problem of finding out about the preferences of individuals. To predict the preferences of an individual it is indispensable to gain certain properties of that individual and known preferences of other individuals. Preference learning requires methods for the automatic, data-driven acquisition of preference models. For machine learning, this type of problem is particularly challenging as it goes beyond the prediction of single values. Instead, it involves the prediction of preference structures in the relational approach or value functions in the utility-based approach. With modeling and learning of preferences, a knowledge engineer is able to describe, calculate and estimate the behaviour of a decision maker by influencing action and utility nodes in a DN. Under the cost aspect, the provision of information is expensive. It seems reasonable to use a piece of information in a decision making process only when it is really needed. From a decision maker's point of view, information should be used only when it is needed and then only to the precision that is necessary. We have discussed an approach which considers this message and thus makes decisions based on some interaction with the agent or user. This framework disregards stepwise all non-efficient and dispensable actions to obtain the most precise problem specification.

Part III.

Transformation Framework

6. Transformation of Graphical Models

We outline a transformation framework for approximating decision networks by fuzzy rule-based systems. The usage of fuzzy *IF-THEN* rules for representing decision functions has multiple advantages like efficient implementation from a computational point of view, approximation of non-linear functional dependencies, information granulation facility and comprehensibility. Rule-based models are intelligible at least if the involved membership functions are restricted to semantically meaningful ones. This chapter introduces the key concepts of fuzzy rule bases, the transformation process of a decision network in a rule base representation including a transformation framework, algorithm *AGoRuB* for automatic generating a rule base structure on the base of the transformation framework, a generalization on fuzzy rule bases and evaluation measures for calculating the granulation criteria based on partition fundamentals.

6.1. Fuzzy Rule Bases

Consider a set of variables X_i with domains $\mathfrak{D}_{X_i}$ $(1 \leq i \leq n)$ and a variable Y with domain $\mathfrak{D}_Y$. Moreover, let $\mathcal{F}_i$ be a fuzzy partition of $\mathfrak{D}_{X_i}$, that is a finite set of fuzzy subsets $F \in \mathfrak{F}(\mathfrak{D}_{X_i})$ such that

$$\sum_{F \in \mathcal{F}_i} F(x) > 0$$

for all $x \in \mathfrak{D}_{X_i}$. Likewise, let $\mathcal{F}$ be a fuzzy partition of $\mathfrak{D}_Y$.

A fuzzy rule base $\mathcal{R}$ is a finite set of fuzzy rules of the form "IF X_1 is in F_1 and X_2 is in F_2 and ... and X_n is in F_n then Y is in F", formally written as

$$< F_1, F_2, ..., F_n | F > .$$

Consider that a variable X_i can formally be omitted in the antecedent of the rule by means of a proper definition of the fuzzy set F_i. There are different types of fuzzy inference schemes. Formally, an inference scheme identifies a function $\phi_{\mathcal{R}}$ to a fuzzy rule base $\mathcal{R}$, such that:

$$\phi_{\mathcal{R}} : \mathfrak{D}_{X_1} \times ... \times \mathfrak{D}_{X_n} \rightarrow \mathfrak{F}(\mathfrak{D}_Y),$$

where $\mathfrak{F}(\mathfrak{D}_Y)$ is the class of fuzzy subsets of $\mathfrak{D}_Y$. If a defuzzification operator $\mathfrak{F}(\mathfrak{D}_Y) \rightarrow \mathfrak{D}_Y$ is applied to the output of this function, the fuzzy rule base $\mathcal{R}$ induces a function

$$\varphi_{\mathcal{R}} : \mathfrak{D}_{X_1} \times ... \times \mathfrak{D}_{X_n} \rightarrow \mathfrak{D}_Y. \tag{6.1}$$

Here, we do neither stick to a particular inference scheme nor to a special defuzzification operator. The important point to realize is simply the following: Once an inference scheme and a defuzzification operator have been determined, each fuzzy rule base $\mathcal{R}$ can be associated with a function (see 6.1).

6.2. Transforming Decision Networks

Our point of departure is a decision network with a set $\mathcal{V}$ of variables. Let $\mathcal{E} = \{E_1, ..., E_n\} \subseteq \mathcal{V}$ denote the set of evidence (information) variables. Moreover, let $\mathcal{A}$ be the set of actions available to the decision maker. The decision maker's preferences are modeled by means of the utility function

$$\mathcal{U} : \mathcal{V} \times \mathcal{A} \to \mathbb{R}.$$

Of course, utility will generally not depend on all variables $V \in \mathcal{V}$. Thus, the domain of $\mathcal{U}$ might be reduced correspondingly.
Note that $\mathcal{S} = \mathfrak{D}_{E_1} \times \mathfrak{D}_{E_2} \times ... \times \mathfrak{D}_{E_n}$ can be considered as the set of potential decision problems or *situations*: Each situation is specified by a vector

$$(e_1, e_2, ..., e_n) \in \mathcal{S}.$$

Given this information, the decision maker has to choose an appropriate action. Let

$$\Delta^* : \mathcal{S} \to \mathcal{A} \tag{6.2}$$

denote some optimal decision function induced by the decision network.
We are now interested in approximating the decision network, that is the optimal decision function (6.2), by means of a fuzzy rule base $\mathcal{R}$. In this connection, the evidence variables $E_1, ..., E_n$ play the role of the variables $X_1, ..., X_n$ in (6.1), and the action variable A corresponds to the variable Y. Moreover, $\mathcal{F}_i$ is a fuzzy partition of $\mathfrak{D}_{E_i}$, and $\mathcal{F}$ is a fuzzy partition of $\mathcal{A}$. Thus, the function (6.1) induced by a fuzzy rule base $\mathcal{R}$ can be considered as an approximation

$$\Delta_{\mathcal{R}} : \mathcal{S} \to \mathcal{A} \tag{6.3}$$

to the optimal decision function (6.2).
Note that the transformed decision model (6.3) will generally involve much less variables than the original model (6.2). In fact, the function (6.3) has $n = card(\mathcal{E})$ arguments, whereas the decision network involves $card(\mathcal{V})$ variables. Apart from that, the domain of the variables E_i is reduced by passing from $\mathfrak{D}_{E_i}$ to the fuzzy partition $\mathcal{F}_i$. In fact, the maximal number of rules in a rule base $\mathcal{R}$ is given by

$$\prod_{i=1}^{n} card(\mathcal{F}_i). \tag{6.4}$$

That is, a fuzzy rule base can be considered as a table with at most (6.4) entries. A *granulation* G specifies a set of fuzzy partitions

$$G = \{\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_n, \mathcal{F}\},$$

i.e. a fuzzy partition for each evidence variable E_i as well as for the action variable A. Let $\mathcal{G}$ denote the class of *allowed* granulations. The class $\mathcal{G}$ might be restricted, for instance, to fuzzy partitions composed of triangular membership functions, or to (Ruspini) partitions $\mathcal{F}_i$ satisfying

$$\sum_{F \in \mathcal{F}_i} F(e) = 1 \tag{6.5}$$

for all $e \in \mathfrak{D}_{E_i}$.
Moreover, let $\mathcal{M} = \mathcal{M}_{\mathcal{G}}$ denote the class of fuzzy rule bases that can be defined using granulations $G \in \mathcal{G}$. Note, that several rule bases can be defined for one granulation G.

6.3. Transformation as Optimization

The quality of a fuzzy rule base $\mathcal{R}$ as a decision function depends on several factors, notably its approximation quality and its complexity. A natural measure of the approximation quality of a rule base $\mathcal{R}$ is the *expected utility loss*: The expected utility of a decision maker using the decision network for choosing actions is given by

$$EU(\Delta^*) = \sum_{e \in \mathcal{S}} EU(\Delta^*(e)|e) \cdot Pr(E = e),$$

where $Pr(E = e)$ is the probability of observing the situation e. Now, the expected utility loss experienced by a decision maker choosing actions according to the mapping (6.3) is given by

$$\alpha(\mathcal{R}) = EU(\Delta^*) - EU(\Delta_{\mathcal{R}}).$$

The complexity of a rule base $\mathcal{A}$ can be defined in different ways. For example, a reasonable measure of complexity is the number of rules, or the sum of the lengths of the rules in $\mathcal{A}$. Subsequently, assume a complexity measure

$$\kappa : \mathcal{M} \to \mathfrak{R}$$

to be given that assigns a real number to each rule base $\mathcal{R} \in \mathcal{M}$. The overall quality of a rule base is a function

$$\mathcal{Q}(\kappa(\mathcal{R}), \alpha(\mathcal{R}))$$

of the approximation quality (expected utility loss) and the complexity. Typically, the decision maker faces an upper complexity bound κ_{max}, i.e.

$$\mathcal{Q}(\kappa(\mathcal{R}), \alpha(\mathcal{R})) = \begin{cases} -\alpha(\mathcal{R}) & \text{if } \kappa(\mathcal{R}) \leq \kappa_{max} \\ -\infty & \text{if } \kappa(\mathcal{R}) > \kappa_{max} \end{cases}. \tag{6.6}$$

The compilation problem can now be formalized as an optimization problem: Find a rule base $\mathcal{R}^* \in \mathcal{M}$ such that

$$\mathcal{R}^* \in arg \ \ max_{\mathcal{R} \in \mathcal{M}} \ \ \mathcal{Q}(\kappa(\mathcal{R}), \alpha(\mathcal{R})). \tag{6.7}$$

6.4. Transformation Framework

It is needless to say that the above optimization problem will generally be difficult to solve. Apart from that, the suitability of a particular optimization method will strongly depend on the structure of the search space $\mathcal{M}$. In most cases, it will be reasonable to apply heuristic methods, such as evolutionary algorithms.
A strategy of turning a decision network into a fuzzy rule base is to derive the latter directly from the formal specification of the network. Here we pursue a *data-driven* approach. That is, the decision network is represented in an *extensional* form as the set

$$\{(e, \Delta^*(e)) | e \in \mathcal{S}\}$$

of situations with optimal decisions, i.e. as the pointwise specification of an optimal decision function (6.2). The objective, then, is to approximate this function by means of a function $\Delta_{\mathcal{R}}$ such that $\mathcal{R}$ does not exceed an upper complexity bound.
Of course, the class $\mathcal{S}$ of situations will generally be large. Therefore, it might not be possible to compute $\Delta^*(e)$ for all $e \in \mathcal{S}$. Rather, optimal solutions will only be derived for a sample $\mathcal{S}_0 \subseteq \mathcal{S}$. Therefore, the problem of compilation is also closely related to (machine) learning. Recall that classical machine learning frameworks usually assume the sample to be generated at random [Bis08]. As opposed to this, the sample $\mathcal{S}_0$ can be chosen freely in this context. That is to say, the decision maker can select a set of *exemplary decision problems* he wants to solve in order to gain in experience. In machine learning, this type of setting is discussed under the name *active learning* [PZ06]. Within this setting, the success of learning can be increased considerably by means of a proper choice of training examples.
The following algorithm abbreviated *TF* is a frame for the transformation of decision models:

1. Specify a class $\mathcal{G}$ of granulations (e.g. range of values, probability information, condense attributes) and a class $\mathcal{M}$ of rule bases. Let $\mathcal{S}_0 = \emptyset$.

2. **Sampling:** Extend the current set $\mathcal{S}_0$ of examples by selecting further decision problems $e \in \mathcal{S}$. Employ the decision network in order to derive optimal solutions $\Delta^*(e)$ for these problems.

3. **Search:** Derive a fuzzy rule base $\mathcal{R}^* \in \mathcal{M}$ which is (approximately) optimal in the sense of (6.7), with the expected utility loss replaced by the empirical utility loss

$$\alpha(\mathcal{R}) = \frac{1}{card(\mathcal{S}_0)} \sum_{e \in \mathcal{S}_0} EU(\Delta^*(e)|e) - EU(\Delta_{\mathcal{R}}(e)|e).$$

   The expected utility $EU(\Delta_{\mathcal{R}}(e)|e)$ is derived on the basis of the decision network by incrementally adding rules, which improve the overall quality of the rule base by calculating each situation.

4. **Evaluation:** Evaluate the current rule base, e.g. by means of some kind of cross validation (which requires a further sample). Decide whether the current rule base $\mathcal{R}^*$ is good enough. If not, proceed with step 5, otherwise exit.

5. **Model Adaption:** Adapt the current class $\mathcal{G}$ of granulations and the class $\mathcal{M}$ of rule bases.

The objective of the above transformation scheme is to find a rule base which guarantees a certain approximation quality with a certain probability.

Example 6.1. Transformation Scenario: Here, we present a short example to illustrate the basic idea of knowledge compilation. This example cannot reveal the advantages of knowledge compilation, since the underlying decision network is not complex enough.
Suppose that a decision maker has to estimate the number of black balls m in an urn containing $n = 20$ balls altogether. That is, an action $\mathcal{A}$ of the decision maker corresponds to an estimation of the number m. Action a yields a loss of $|a - m|$ if $a \leq m$ and of $(a - m)^2$ if $a > m$, i.e. overestimation is more punished than underestimation. The information available to the decision maker is the number of black balls in a sample with replacement of size 6, i.e. the set of situations is given by $\mathcal{S} = \{0, 1, ..., 6\}$.
Fuzzy partitions have to be defined for the evidence variable E, i.e. the number of black balls in the sample, and the action variable A. Suppose that only granulations $\mathcal{F}_1$ and $\mathcal{F}$ with $card(\mathcal{F}_1) = card(mathcalF) = 3$ are allowed. Moreover, only triangular membership functions are allowed, and property (6.5) must be satisfied. Consequently, a fuzzy partition is actually determined by only one number. In fact, a fuzzy rule base $\mathcal{R}$ is obtained by assigning to each of the antecedents $(0, 0, \alpha)$, $(0, \alpha, 1)$, $(\alpha, 1, 1)$ one of the consequences $(0, 0, \beta)$, $(0, \beta, 1)$, $(\beta, 1, 1)$, where (a, b, c) denotes the triangular membership function with support (a, c) and core $\{\beta\}$. As can be seen, the class $\mathcal{M}$ of rule bases is small enough and can be searched completely. Since the class $\mathcal{S}$ of situations is very small as well, some difficult aspects of the above transformation framework simply disappear in this example, namely sampling, evaluation and model adaption.
We interpret a rule base $\{< F_{11}|F_1 >, < F_{12}|F_2 >, < F_{13}|F_3 >\}$ by the additive

fuzzy inference

$$\phi : x \mapsto \Big(\sum_{i=1}^{3} F_{1i}(x) \cdot F_i\Big)\Big(\sum_{i=1}^{3} F_{1i}(x)\Big)^{-1}$$

with subsequent center of gravity defuzzification (plus rounding of the result to the closest integer). It turns out that the optimal rule base then consists of the following rules:

$$\langle (0,0,3)|(0,0,3)\rangle$$
$$\langle (0,3,6)|(0,3,20)\rangle$$
$$\langle (3,6,6)|(3,20,20)\rangle.$$

The expected loss induced by this rule base is 8.633, which is quite close to the expected loss 8.244 associated with the optimal decision function Δ^*.

6.5. Automatic Generation of a Rule Base by Means of a Decision Network (AGoRuB)

On the base of the framework for the transformation of decision networks in fuzzy rule bases, presented in the preceding sections, an algorithm has been implemented, which is taking up the essential ideas from the rough algorithm *AGoRuB*. For this, an especially fitting environment for the development has been searched. Since actually only few software support is granted, which furthermore supports the main operations for decision networks, practically only *HUGIN Expert* as specific program came into consideration. *HUGIN Expert* is a commercial software system with its roots in Denmark, which provides all necessary functions for the creation of -, and the inference inside Bayesian networks and decision networks as well. Because of the fact that it is a commercial product, there was no possibility to attain the source code. Nevertheless, HUGIN Expert's homepage [HUG08] provides a special version for research- and demonstrational purposes, whereby, after a cost - free registration, the download is free as well. This version also contains an API (Application Programming Interface), which enables the user for certain important methods like load, save, set and propagation of evidence in decision networks. Hence, a disadvantage is that merely small decision networks (to a maximum of 25 nodes) can be handled, thus nevertheless being enough for the demonstration of the algorithm in the case of a small example.
On this base a JAVA-program named *DNCompiler.java* has been written by the author of this thesis, which the HUGIN Expert API integrates in the form of the two data files *hapi67.jar* and *hapi67.dll* (also compare HUGIN API reference manual with version 6.7 [HA08]). The *AGoRuB* algorithm is exemplified as sequence diagram and can be expressed in pseudo code as illustrated in Figure 6.1. The algorithm is roughly separated in five parts [HBF06]. As input, a decision network which is compatible with HUGIN Expert is given as $*.hkb$ file. We are able to use the COM.hugin.HAPI package which is integrated in the HUGIN Java API and provide us access to all HUGIN classes and methods.

After compiling the DNCompiler file via "$c:\j2sdk\bin\javac\ DNCompiler$" the DNCompiler class-file is generated and availabe. Next start the Java-program with *"java DNCompile"* and load the decision network which is given as $*.hkb$ file, for instance appointed as test.hkb file.

Step 1: Load the decision network and initialize the new domain as "decisionNetwork = new Domain(test.hkb)". The compiled decision network with domain *myNet* contains the chance nodes associated with CPTs and decision nodes representing the decisions being made at a particular point of time.
Step 2: Generate a node list and get all nodes from the new domain *myNet*. The program distinguish between chance nodes and decision nodes. Chance nodes are successively added to the *chanceNodeList* after the domain affiliation is proven. We support in this algorithm discrete decision nodes which are initially set as output zero expressed as *decision=null*. A system printout displays the number of nodes and their corresponding node names at that time.
Step 3: Allocate the chance nodes with valid data and initialize a situation based on the chance node list. It follows the initialization of the rules and the rule base RB. The algorithm starts without observations in the domain myNet and set the evidence to zero. By now, the rule base is given and all chance notes are allocated.
Step 4: The calculation of the optimal rule base plays a key part in this algorithm. The rule base initialized in the previous step calls the method *getOptimalRuleBase* to calculate an optimal rule base compared to the original decision network (see Table 6.1). The method *getOptimalRuleBase* backfills the rule-base "ruleBase" step by step with respective best-fitting rules. The backfilling of the rule-base terminates when the addition of a further rule does not increase the quality of the rule-base any further:

Table 6.1.: Methods getOptimalRuleBase and getOptimalRuletoAdd of the decision network myNet in step 4

Method and Description
protected static RuleBase getOptimalRuleBase (RuleBase ruleBase, int [] rule, int[] situation, Domain myNet, NodeList chanceNodeList, DiscreteDecisionNode decision) Serves the compilation of the decision network myNet and provides an (approximately) optimal rule-base of the RuleBase-type.
protected static int [] getOptimalRuleToAdd(RuleBase ruleBase, int [] rule, int[] situation, Domain myNet, NodeList chanceNodeList, DiscreteDecisionNode decision) Auxiliary method for the determination of an (approximately) optimal rule that, in the further consideration, is adopted into the rule-base.

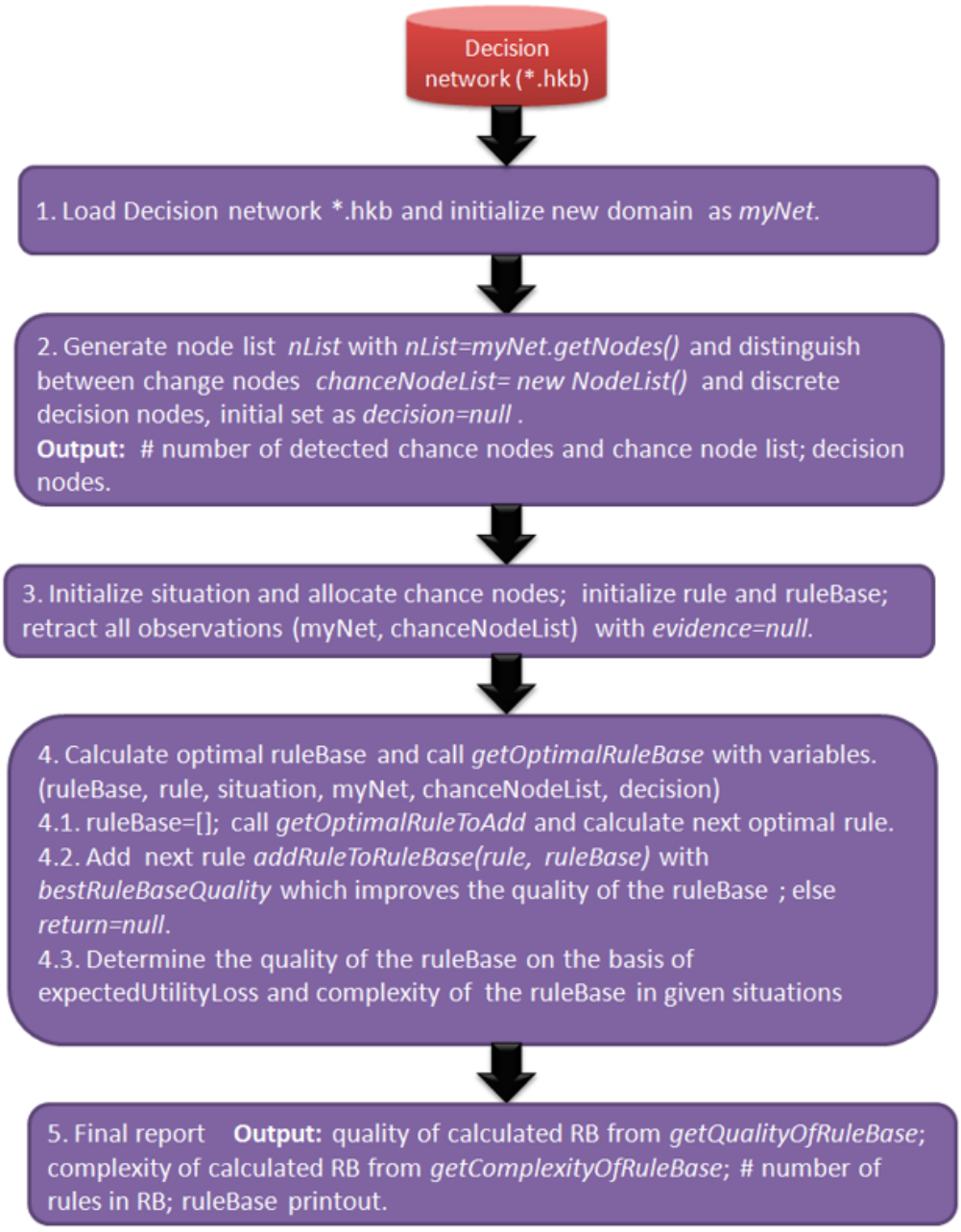


Figure 6.1.: AGoRuB algorithm which determines the quality and complexity of ruleBase

```
if (bestRuleBaseQuality >= oldBestRuleBaseQuality) {
 return(null);}
 else {
 return(bestRuleToAdd);}
```

In this case the subordinated method *getOptimalRuleToAdd* provides null back and the calculated rule-base, refilled with rules, is given back.
Just like the name already tells, the method *getOptimalRuleToAdd* is determining a rule that improves the quality of hitherto generated rule-bases in the best way. The method *getOptimalRuleToAdd* calls the method *getQualityofRuleBase* (see Table 6.3) and prints out the temporary rule base quality consisting of the expected utility loss on the basis of the decision of the decision network *myNet* and the complexity in RB.
As a measure for the *complexity* of a rule-base the sum of lenght's off all rule-bases has been chosen. Accordingly clear the code is configured in two interlaced loops, whereby the exterior loop runs about the number of involved rules and the inner loop about the certain length of a rule.
For the determination of the *quality* of approximation of a rule-base, according to the achievements of the preceding Section 6.3, the loss of utility, which occurs, if instead of the decision-network (*getDecisionOfDecisionNetwork*), the rule-base for the determination of the decision for a certain given situation is used (*getDecisionOfRuleBase*), has to be calculated. The methods *getDecisionOfRuleBase* and *getDecisionOfDecisionNetwork* are hereby used as aided methods to select situations, in which the resulting decision differs and the loss of utilization respectively has to be actualised. Inside the while-loop thereby it

Table 6.2.: Auxiliary methods for decision making

Method and Description
protected static int getDecisionOfDecisionNetwork (int[] situation, Domain myNet, NodeList chanceNodeList, DiscreteDecisionNode d) Auxiliary method for the determination of a decision of the decision-network with a given situation "situation".
protected static int getDecisionOfRuleBase (int[] situation, RuleBase ruleBase) Auxiliary method for the determination of a decision of the actual rule-base with a given situation "situation".

gets iterated about all considered situations (see Table 6.3). Depending on a certain implementation of the auxiliary method *getNextSituation* we can operate in this place, which situations are viewed as exemplarily and therefore are included in the calculation of the *ExpectedUtilityLoss*. After the *ExpectedUtilityLoss* is summed up over all exemplary situations, the return-part grips and provides the quality of the rule-base "ruleBase".

This quality contains the sum of all successively summed up losses of utility

Table 6.3.: Method getQualityOfRuleBase for the determination of the quality of a rule base

Method getQualityOfRuleBase

Step 1:
Initialize new Domain *myNet* and load Decision network *DecNet* as HUGIN file "*DecNet.hkb*".

Step 2:
Get detected ChangeNodes CN and set node list chanceNodeList= $\{CN_1, ...CN_n\}$.
Get detected discrete DecisionNode DN and set with decision=null.

Step 3:
Start with an empty RuleBase $RB = \{\}$.
Initialize variable $\beta \in [0, 1]$ as influencing factor, which represents the complexity of the RuleBase. The quality of the RuleBase is composed of the calculated expectedUtilityLoss and complexity. Set $\beta := 1$.
Set expectedUtilityLoss := 0. Set ComplexityofRuleBase := 0.

Step 4:
4.1.: Determine the expected utility of the Decision Network.
Calculate for each situation $(cn_1, ...cn_n) \in CN$ and action in DN the local utility based on utility values. Get the maximal ExpectedUtility with action $a \in domain(DN)$.
Set QualityOfRuleBase := ExpectedUtility(decisionDecisionNetwork).

4.2.: Determine the expected utility of the RuleBase.
Pass through all combinations of situations and actions of all rules.
Start with situation[i] = -1. Get decision d of RuleBase.
Calculate ExpectedUtility of RuleBase based on d.
Add rule to the RuleBase.

LOOP
4.3.: Calculate QualityOfRuleBase
Compute ComplexityOfRuleBase as sum of number of rules plus rule length.
Calculate ExpectedUtilityLoss := ExpectedUtility(decisionDecisionNetwork)-ExpectedUtility(decisionRuleBase).

4.4.: Find new minimum
Find next rule which increases the expected utility of the RuleBase (mark as new minimum).
Add rule to the RuleBase if RuleBaseQuality > tempRuleBaseQuality.
Calculate QualityOfRuleBase := ExpectedUtilityLoss+ $\beta\cdot$ ComplexityOfRuleBase.
GOTO 4.3. UNTIL $\text{QualityOfRuleBase}_{old} \geq \text{QualityOfRuleBase}_{new}$.

Step 5: Final RuleBase Quality Output
Get QualityOfRuleBase and printout the final report including the quality of the calculated RuleBase, the expected utility loss, the complexity of the calculated RuleBase and all calculated rules.

of all considered situations and further of the complexity of the rule-base, just like it has been discussed on a more general level in the preceding Section 6.3. The beta parameter weights in this context the single values for complexity and complexity of approximation of the respective rule-base.
Step 5: The final report contains the total quality of the calculated rule base on the basis of the return value from *getQualityOfRuleBase*, the complexity of the calculated rule base on the basis of the return value from *getComplexity-OfRuleBase*, the number of rules and the rule base including the chance node list and decision.

The AGoRuB algorithm for the compilation is tensely connected to the frame for the transformation of decision models presented in Section 6.4. However certain simplifications are made on two different places. At first, following algorithm *TF*, it is necessary to accomplish a sampling, which serves for the construction of a set S_0 of exemplary decision situations. In this cohesion it is absolutely uncertain, which decision situation has to be included into the set S_0, and also which not. For this reason all possible situations are considered and weighted with the certain possibility of their appearance. This is resulting in the fact that the sampling of the first step is reduced to the inclusion of all possible situations inside the set S_0. An alternative possibility might be to identify certain standard-cases with the aid of experts knowledge and to build the set S_0 correspondingly, thus being very time- and cost-consuming. Even a random assortment of standard situations would be conceivable in this context, however, by this, important exemplary situations are neglected. A further simplification of the algorithm is in the fact that the output does not contain fuzzy rules, but rather is represented by a rule-base comprising exact *IF THEN*- rules. With this, the problem of model adaption, implied in the preceding section, gets eluded and step 5 of algorithm *TF* therefore can be economized without substitution. Because of the modular structure of the algorithm it is easily possible in future implementations to hurdle both of these problems. Furthermore it has to be noticed that the inclusion of all possible decision situations (weighted with the respective possibility of their appearance) does not solely imply disadvantages at all, but, considering the quality of the resulting rule base, quite positive consequences can occur, since rather seldom appearing decision situations are also considered in the calculation of the rule base. As the heuristic search strategy in the rule-base space a Greedy Hill Climbing algorithm is adopted.
Referring finally to the general remarks in Section 6.4, two factors are conclusive in this coherence. On the one hand, the **complexity** of the rule-base, which is determined by the method *getComplexityOfRuleBase*, and on the other hand the **quality** of approximation of the rule-base, which is determined by the method *getQualityOfRuleBase*.

To sum it up the process of transformation comprises the following steps:

1. Learn a Bayesian network with the aid of a database (for example with *LAGD Hill Climbing*) or construct the Bayesian network together with

conditional probability tables manually (for example with experts' interviews).

2. Amplify the Bayesian network with the aid of *HUGIN Expert* with utilization- and decision-nodes to represent the users preferences.

3. Use the *DecisionNetworkCompiler* to compile the decision network, constructed in 2., into a *IF-THEN* rule-base.

To test the algorithm a "crop problem" is used.

Example 6.2. Crop Problem: Consider the working farm from Mr. Spyers who is located in Oklahoma. Mr. Spyers tries to yield a large crop with his corn field. Someday he observes that different ears of corn bend down in most cases caused by long periods of dryness or specific disease (for instance attacked by mildew). Mr. Spyers considers either to invest in a treatment or to wait for the next pluvial. To model the temporal aspect a dynamic BN is constructed and augmented with the decision variable *treatment* and the utility functions *cost* and *crop* as illustrated in Figure 6.2.

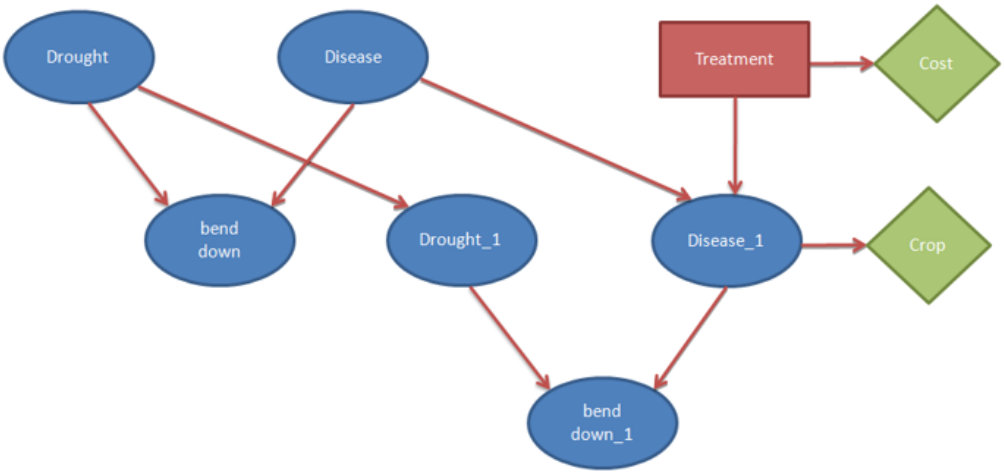


Figure 6.2.: The complete qualitative representation of the crop problem as a decision network

Since the crop network is a complete Decision network, all preferences of a user are already declared inside the network, so that step 3 of the previous process of compilation, the execution of *DecisionNetworkCompilers*, can be used as the starting point.

Here all occurring random and decision nodes are being automatically detected and the initialization of a blank rule-base, which gets filled step by step in the course of the further compilation with exact rules, is done on this base. Right here, the output of the DNCompiler for the Decision network "crop problem" is presented in Table 6.4 and Table 6.5.

The condition "-1" here shows that for the respective node no evidence is existent, while "0" refers to the condition "false" and "1" represents "true".

Table 6.4.: Output of the DecisionNetworkCompiler for the "crop problem"

Final report

Quality of calculated RuleBase: 230.07820364277225
Expected Utility Loss: 209.07820364277225
Complexity of calculated RuleBase: 21
 	
# Rules in calculated RuleBase: 11 	
 	
Rules 1-6:
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=1)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=0) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=0) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=0) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=0) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=0) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)

Table 6.5.: Output of the DecisionNetworkCompiler for the “crop problem“

Final report

Quality of calculated RuleBase: 230.07820364277225
Expected Utility Loss: 209.07820364277225
Complexity of calculated RuleBase: 21
 	
# Rules in calculated RuleBase: 11 	
 	
Rules 7-11:
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=0) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=1) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=1) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)
 	
IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND (drought_1=1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0)

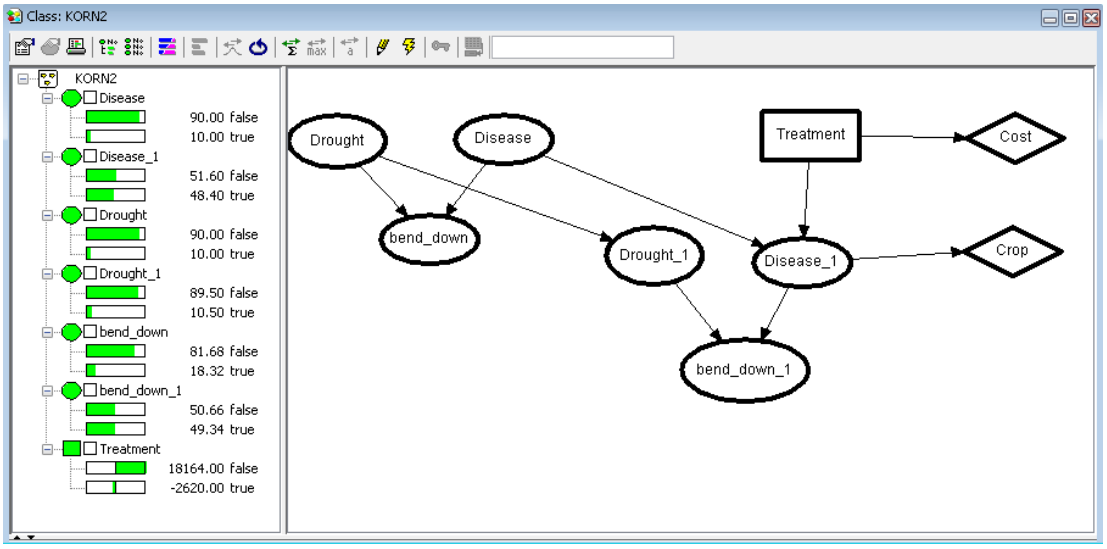


Figure 6.3.: HUGIN Expert crop decision network with node list

Furthermore it has to be noticed that in the case of multiple firing rules the respective rule will be taken, which is last listed in the rule-base for the determination of the decision. Apparently the compilation leads to an assessable rule base consisting of 11 rules. The accuracy of approximation is, despite minor complexity of the rule base, relatively high, because the Expected Utility Loss is roughly about 209 (compared with the underlying decision network). The optimum would of course be an Expected Utility Loss of zero. Consider the fact that an empty rule base has an approximate value of 18563. The rule base is very compact and can therefore be efficiently evaluated. It becomes obvious that the calculated rule base has found a good compromise between the accuracy of approximation and the complexity. The rules themselves are quite feasible: If nothing is known about the conditions of every single random node or if no other rule fires, the rule base provides the conclusion *treatment=1*. Therefore the treatment of the grain is recommended to prevent a disease. If, from the first, the condition of the random node *disease_1*, that is anyway the state of the grain at the temporary point of harvesting, is already known, the rule base provides the answer *treatment=0*. This also makes sense, because the treatment, or the neglect of any action does not affect the condition of the random node *disease_1* any more. Because of the tense connection between *disease* and *disease_1* it is also applicable that: If no disease is existent at the actual point of time, then do not make any treatment (compare with rule 3).
The two last rules in Table 6.5 represent a prime example for the phenomenon *"Explaining away"* [WH93]. Explaining away can be interpreted as common pattern of reasoning in which the confirmation of one cause of an observed or believed event reduces the need to invoke alternative causes. The opposite case occurs, where the confirmation of one cause increases belief in another.
First consider rule 11:

IF (drought=-1) AND (disease=-1) AND (bend down=-1) AND

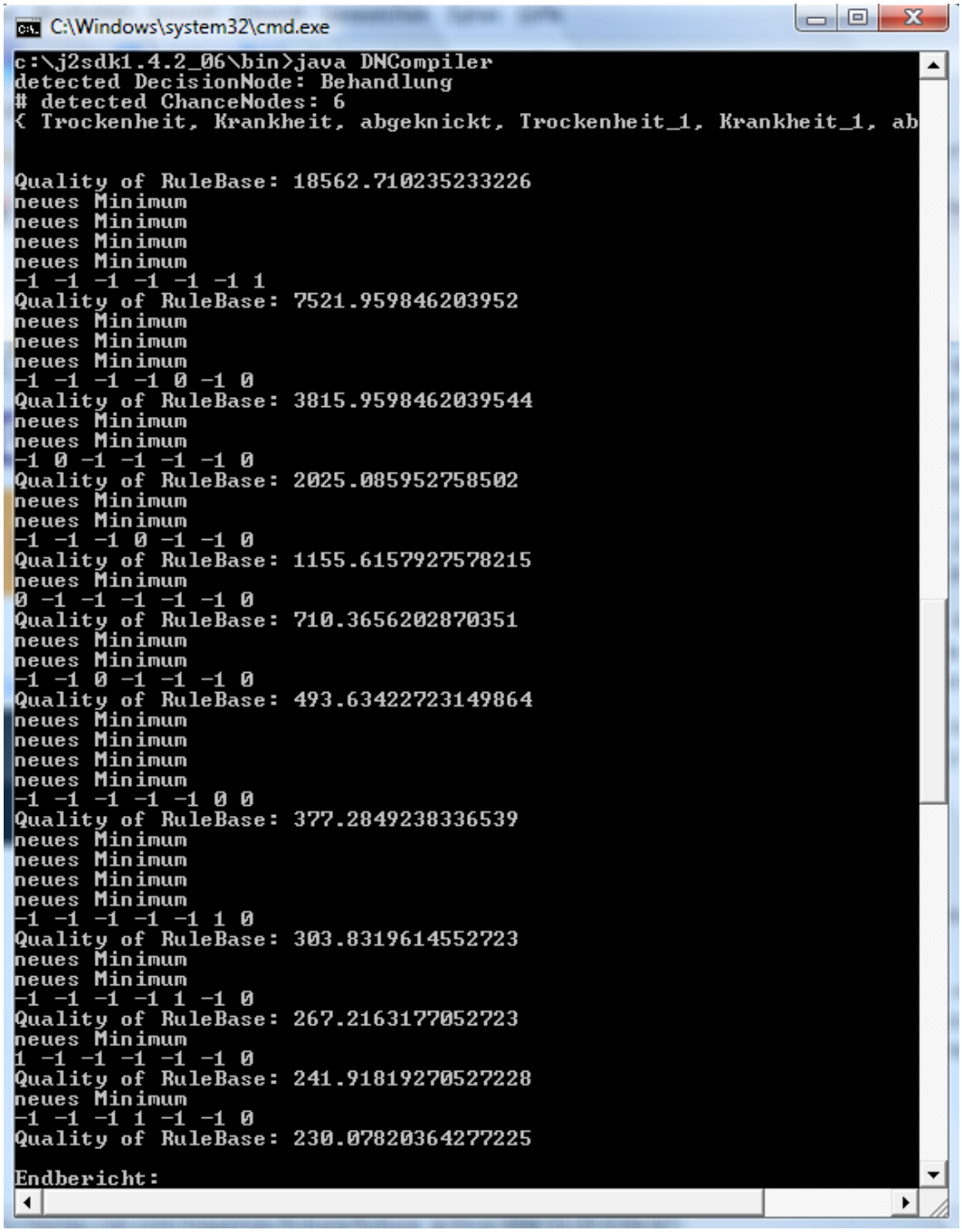


Figure 6.4.: DNCompiler output of the crop decision network

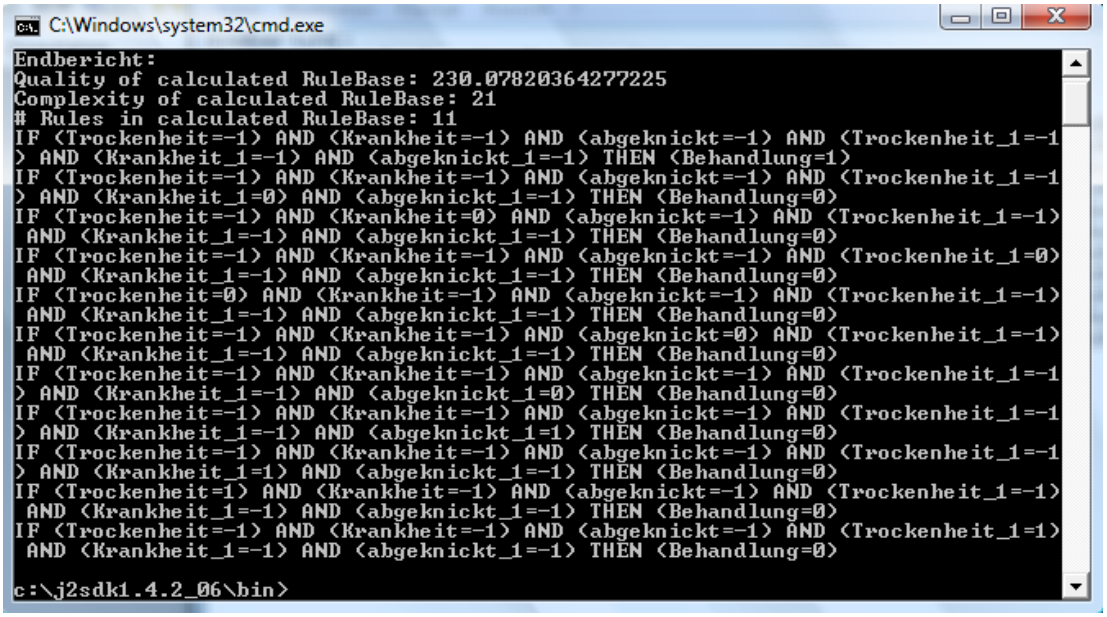


Figure 6.5.: DNCompiler final report of the crop decision network

(drought_1=1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0).

Apparently the situation, in which solely the evidence *drought_1=1* (that declares the aridity at the date of harvesting) is available, the best decision would be not to conduct any treatment for the grain, in order to save costs. A closer look at the structure of the underlying decision-network in Figure 6.2 reveals why this is the case. There are two competing causes for the deviation of the ears at the date of harvesting (bend down_1=1): On the one hand it is drought_1=1 and on the other hand disease_1=1. The observation drought_1=1 (from the requisite part of the rule) creates another possible cause, that is to say disease_1=1, quite less possible. In other words: drought_1 *explains* disease_1 *away* (hence the term explaining away). Due to the lowered probability for disease_1=1 the decision is made in favour of the non-treatment of the grain, since it is most likely healthy. In statistics this phenomenon is well known as *Berkson's Paradoxon* (compare [DR82] and [Edw00]). For instance, two diseases may be found to be correlated amongst patients but be unrelated in the general population. Probably the same phenomenon is also behind the other rules.
Rule 10:

IF (drought=1) AND (disease=-1) AND (bend down=-1) AND (drought_1=-1) AND (disease_1=-1) AND (bend down_1=-1) THEN (treatment=0).

Here with rule 10 evidence for aridity is existent at the actual point of time drought=1. By means of the structure of the Decision network (compare 6.2) and the occupancy of the connected conditional probability tables (see table 6.2.5) we can see that the decision network is almost modeling the implication

drought → *drought_1*, whereby again the situation discussed beforehand is arising, and because of *Berkson's Paradoxon* the decision will be made in favour of the non-treatment of the grain.

In a second test a stock exchange situation is used. First, on the base of an acquired database, a Bayesian network inside the domain *stock exchanges and economic situation* has been learned. For this, 1314 data sets have been gathered in a data base for the period from 14.02.2003 to 14.09.2007 and been processed for the WEKA-internal *ARFF*-format.

Example 6.3. Stock Exchange: Consider the stock exchange situation in which the single variables comprise the price for crude oil, the market price of the Dax, the market price of the EuroStoxx50, the exchange rate for EUR/US-Dollar, the unemployment rate (in Germany), the capacity utilization and the Ifo Business Climate Index as illustrated in Figure 6.6.

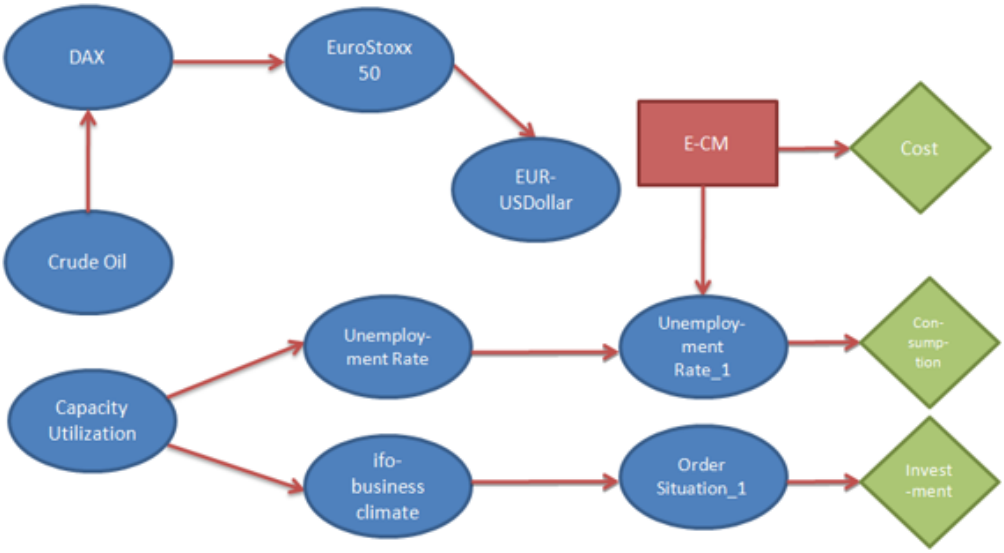


Figure 6.6.: The complete qualitative representation of the stock exchange problem as a decision network

By applying the newly presented LAGD Hill Climbing structure-learning algorithm a Bayesian network has been calculated, which models the interdependencies and interactions between every single variable. Following this, the ready-learned Bayesian network could be improved to the state of a decision network by firstly integrating two additional random variables ("Unemployment Rate_1" and "Order Situation_1"), which are modeling the temporal aspect. Furthermore the decision node "E-CM" as employment-creation measure and three utility nodes ("Cost", "Consumption" and "Investment"), which represent the user's preferences, have been integrated into the actual dynamic bayesian network (see Figure 6.6).
Apparently, many different dependencies exist between the price of crude oil, the market price of the DAX, the market price of the EuroStoxx50 and the exchange rate for EUR/US-Dollar. Nevertheless, these variables are, on the base

of the currently available data, not directly connected with the unemployment rate, the capacity utilization and the Ifo Business Climate Index, so that the result is a partially disconnected network. The decision node *E-CM* represents the decision *"Is a E-CM procedure necessary?"* and can therefore contain the values *false* and *true*. Of course, such a measure causes several costs, therefore a straight connection to the utility node *Cost* has been established. On the other hand, such a decision also effects the unemployment rate in a future point in time (Unemployment Rate_1). The future unemployment rate again effects the consumption, while the Ifo Business Climate Index on the actual state effects the future order situation (Order Situation_1). This future order situation on the other side is tensely connected with corporate investments, thus being the reason for the direct linkage to the benefit node "Investment". A further result of the learning process is the dependency of the unemployment rate as well as the Ifo Business Climate Index on the capacity utilization. In analogue causes to the crop problem the *DecisionNetworkCompiler* has been used to compile the decision network into an efficiently evaluable rule base. Here, the output of *DNCompiler* concerning the decision network *stock exchanges and economic situation* is given as illustrated in Table 6.6 and Table 6.7.
Just like the compilation of the Decision Network "crop problem" has already shown, the situation "-1" here represents the state, that the respective node has no evidence. "0" refers to the situation "low", "1" to the situation "normal" and finally "2" to the situation "high". As can be seen, decision making here only requires the situations of the random nodes *Order situation_1*, *Unemployment Rate_1*, *Unemployment*, *ifo-business climate index* and *Capacity Utilization*. Just 11 rules are sufficient to decrease the actual *Expected Utility Loss* from about 25308 (empty rule-base) to about 177. The rules themselves seem to be plausible, but with one exception. As soon as the situation of the random node *Unemployment Rate_1* is known, it is recommended not to do any E-CM procedure (compare rules 3,4,5,6 and 7 in Table 6.6 and Table 6.7). The simple reason for this is that the accomplishment of an E-CM procedure (*E-CM=1*) has not an influence on the situation of the node *Unemployment Rate_1* any more, and therefore the costs for this measure can be saved. In practise, any access to the situation of the node *Unemployment Rate_1* (unemployment in the future is not very realistic, so that the other rules might be more interesting to handle. So it is advisable, unbeknownst to the situations of the involved variables, to arrange a job creation measure (compare rule 1). This also counts for the case, if no other listed rule might fit later on. If actually the employment rate is low (*Unemployment Rate=0*), it is advisable not to enforce any E-CM measure (compare rule 2). The last rule on the other side seems to be a bit strange: If it is already known that the employment level in future will be low (*Unemployment Rate_1=0*), but the actual level is still high (*Unemployment Rate=2*), the arrangement of an E-CM measure is advisable in any case. The addition of this rule nevertheless decreases the Expected Utility Loss, so that the decision behaviour of the underlying decision network can be approximated in a better way by the resulting rule base, than without this rule.

Table 6.6.: Output of the DecisionNetworkCompiler for the "stock exchange problem"

Final report

Quality of calculated RuleBase: 205.76167039852027
Expected Utility Loss: 176.76167039852027
Complexity of calculated RuleBase: 29
 	
# Rules in calculated RuleBase: 11 	
 	
Rules 1-6:
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=-1)
AND (Unemployment Rate=-1) AND (ifo-business climate=-1)
AND (Capacity Utilization=-1) THEN (E-CM=1)
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=-1)
AND (Unemployment Rate=0) AND (ifo-business climate=-1)
AND (Capacity Utilization=-1) THEN (E-CM=0)
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=1)
AND (Unemployment Rate=-1) AND (ifo-business climate=-1)
AND (Capacity Utilization=-1) THEN (E-CM=0)
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=2)
AND (Unemployment Rate=-1) AND (ifo-business climate=-1)
AND (Capacity Utilization=-1) THEN (E-CM=0)
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=0)
AND (Unemployment Rate=1) AND (ifo-business climate=-1)
AND (Capacity Utilization=-1) THEN (E-CM=0)
 	
IF (Order Situation_1=-1) AND (Unemployment Rate_1=0)
AND (Unemployment Rate=-1) AND (ifo-business climate=1)
AND (Capacity Utilization=-1) THEN (E-CM=0)

Table 6.7.: Output of the DecisionNetworkCompiler for the "stock exchange problem"

Final report
Quality of calculated RuleBase: 205.76167039852027 Expected Utility Loss: 176.76167039852027 Complexity of calculated RuleBase: 29 # Rules in calculated RuleBase: 11 Rules 7-11: IF (Order Situation_1=-1) AND (Unemployment Rate_1=0) AND (Unemployment Rate=2) AND (ifo-business climate=-1) AND (Capacity Utilization=-1) THEN (E-CM=1) IF (Order Situation_1=-1) AND (Unemployment Rate_1=0) AND (Unemployment Rate=-1) AND (ifo-business climate=0) AND (Capacity Utilization=-1) THEN (E-CM=0) IF (Order Situation_1=-1) AND (Unemployment Rate_1=0) AND (Unemployment Rate=2) AND (ifo-business climate=-1) AND (Capacity Utilization=-1) THEN (E-CM=1) IF (Order Situation_1=-1) AND (Unemployment Rate_1=0) AND (Unemployment Rate=-1) AND (ifo-business climate=2) AND (Capacity Utilization=1) THEN (E-CM=0) IF (Order Situation_1=-1) AND (Unemployment Rate_1=0) AND (Unemployment Rate=2) AND (ifo-business climate=-1) AND (Capacity Utilization=-1) THEN (E-CM=1)

Altogether it can be declared that the tradeoff between a high accuracy for approximation on one side and a low complexity of the rule base on the other side in both executed tests is already working well. With another weighting of the complexity of the rule base (by adjusting the parameter "beta" inside method *getQualityOfRuleBase* (compare Table 6.3)) the algorithm can be operated into the direction of higher accuracy of approximation or lower complexity of the entire rule base.

6.6. Generalization on Fuzzy Rule Bases

Although the algorithm presented in the preceding chapter automatically solves the problem of compiling into an evaluable rule base it would still be desirable to attain the result of a fuzzy rule base, instead of a rule base consisting of sharp rules. The advantages are obvious:

- Fuzzy rule bases are efficiently evaluable.
- Fuzzy rule bases can approximate non-linear functional dependencies.
- Fuzzy rule bases are easily interpretable.
- The integration of fuzzy enables the rule base to include vague observations into the decision making process.

Especially the second and last point clarify the dominance compared with rule bases consisting of sharp rules. Furthermore, with the integration of fuzzy, the possibilities of the underlying decision network are improved, which is merely working with clearly specified values for all involved variables.
For the implementation, a structure similar to the structure of *DNCompiler*, presented in the preceding section, is applied. Since *DNCompiler* is working with sharp rules, all methods, accessing the constructive rule base, have been adapted. Fuzzy structures and mechanisms of inference have been embedded in the program *FuzzIT* of department 1 (department for informatics, University of Dortmund). *FuzzIT* stands for the term *Fuzzy Inference Tool* and enables to design a Fuzzy Controller with an editor or textually, to further use it for inference. The problem concerning the initialization and adaption of fuzzy rule bases, discussed in Section 6.2, has been avoided by the distinction of the fuzzy controller by the user himself. This controller aligns, besides the linguistic terms of the linguistic variables, a fuzzy controller and therefore determines methods, which are used for fuzzyfication, inference, aggregation and defuzzification. The result is a strict dependence of the quality of the calculated rule base and the design of the Fuzzy Controller.
Since, at the actual stage of development, it still is not clear, how a concrete model adaptation can be performed (compare Section 6.2) the initial definition of the user seems actually to be the best solution. In summary the process of transformation is composed of the following steps:

1. Learn a Bayesian network with a database (for example with LADG hill climbing) or construct the Bayesian network including conditional probability tables manually (for example with the aid of experts' interviews).

2. Amplify the Bayesian network with the aid of *HUGIN Expert* with utility and decision nodes for the representation of the user's preferences.

3. Determine the fuzzy controller's design with the aid of FuzzIT.

4. Use the *DecisionNetworkCompiler* to compile the decision network constructed in 2 on the base of the Fuzzy Controller determined in 3 into a rule base.

To test the algorithm the "crop problem", presented in detail in Section III, is applied. Since this is a complete decision network, all user's preferences are declared in the network and so step 3 can be started, the determination of the fuzzy controller's design. Figure 6.7 presents the, for the process of compilation,

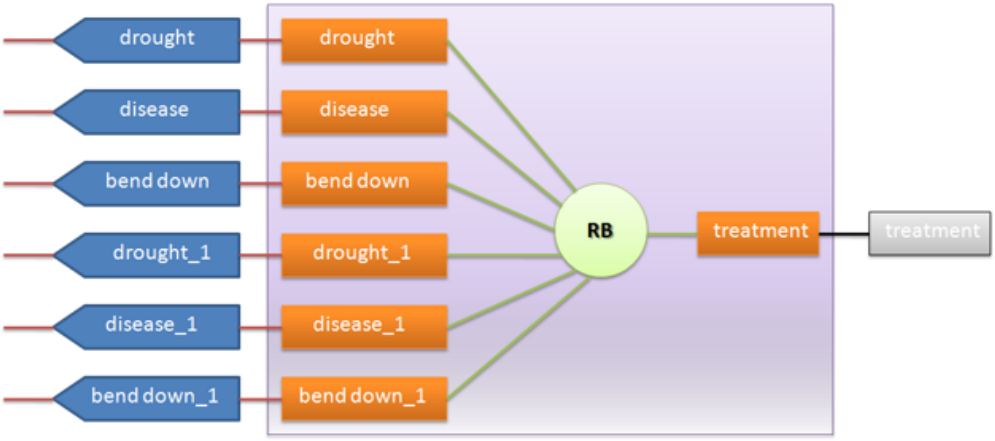


Figure 6.7.: Design and settings of the fuzzy controller for the crop problem

underlying design of the fuzzy controller, whereby its rough structure has already been determined by the utility and decision nodes of the decision network. Figure 6.8 presents the linguistic terms that can be aligned to the variable *bend down*. The blue line belongs to the linguistic term "false", the green line to the term "a bit" and the red line finally to the term "true". Similar to this, the remaining linguistic variables have been designed. A fuzzy logic system which uses fuzzy uncertainties consists of three main blocks: fuzzification, inference mechanism, and defuzzification [BD97]. Furthermore the following design decisions have been made inside the fuzzy controller (compare Figure 6.7):

- **Fuzzyfication:** Fuzzification is a mapping from the observed numerical input space to the fuzzy sets defined in the corresponding universes of discourse. The fuzzifier maps a numerical value into fuzzy sets represented by membership functions in the universe U [EVW05]. We have to determine the degree of membership for each term in an input variable as illustrated top left in Figure 6.9. On the left border of Figure 6.7 six orange-coloured fuzzyficators can be seen applying FuzzIT, which comply with the random nodes of the underlying decision network for the transformation. Though, while the decision network only works with the

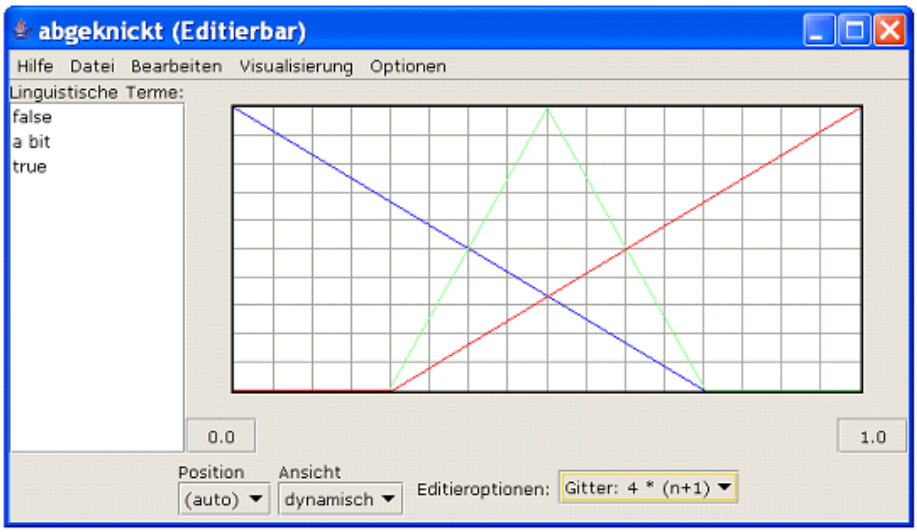


Figure 6.8.: Defined linguistic terms on the sample of the linguistic variable bend down with FuzzIT

values "false" and "true" for the situation of a respective node, the designed fuzzy controller enables the decision network to work with any value inside the interval [0,1]. As the respective method for fuzzification *Fuzzy Sinlgetons* have been chosen. The reason why Fuzzy Singletons have been chosen and not triangular functions for affiliation is that for the evaluation of the rule base the situations declared inside the program are to be pulled up. These situations are nevertheless nothing more than fixed allocations of certain states to respective variables. Therefore no uncertainty is existing and *Fuzzy Singletons* have to be preferred. The problem that in certain situations some variables are without a value has been solved the way that the Fuzzy Output of the respective fuzzificator in this specific case has been set blank.

- **Inference:** Inference mechanism is the fuzzy logic reasoning process that determines the outputs corresponding to fuzzified inputs. Each IT-THEN rule defines a fuzzy implication between condition and conclusion rule parts. The implication operator used here is the minimum operator. In other words, for each rule the firing strength is applied to modify the its consequent fuzzy set resulting in a new fuzzy set as the response of the rule. The fuzzy implication is a mapping from the antecedent linguistic terms to the rule conclusion linguistic term.

- **Aggregation:** Aggregation combines responses of individual rules to yield an overall output fuzzy set using the max-operator for union. For the aggregation of the premises referring to the logical AND the minimum-method has been chosen. This assures that the strength of a rule is determined by the weakest part of the 6 involved input variables while,

as an operator for implication, the maximum was applied. Since the controller fits with the FITA principle, inference follows on the aggregation of rules. As an operator for aggregation the maximum has been chosen, because this emulates the logical "OR". The strength of the result of the aggregation therefore is best dominated in this context by the best fitting rule.

- **Defuzzification:** Defuzzification maps output fuzzy sets defined over an output universe of discourse to crisp outputs as valve in Figure 6.9. It is employed because in many practical applications a crisp output is required. A defuzzification strategy is aimed at producing the non-fuzzy output that best represents the possible distribution of an inferred fuzzy output. On the right side of Figure 6.7 the orange-coloured defuzzificator "treatment" can be noticed. This defuzzificator converts the fuzzy output of the inference engine into a crisp value from the output membership inside the interval [0,1]. As the method for defuzzification the *Center of Gravity* method has been applied. Another well-known methods are the max criterion method or the mean of maximum method [KGK95].

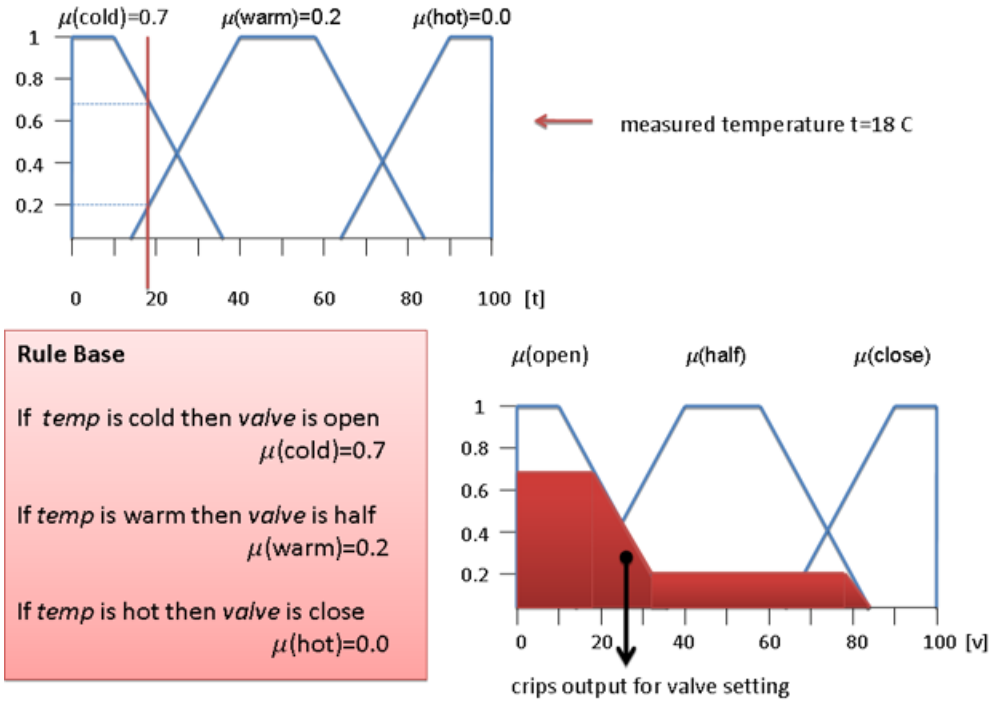


Figure 6.9.: Fuzzification of the linguistic variable temperature

Consider the example from fuzzy control, where we use a fuzzy rule based system to approximate the behaviour of a complex process. To facilitate the understanding of the main blocks which are described before, a simple example is illustrated in Figure 6.9. On the basis of a measured temperature at a

machine in a factory, we have to set a valve which is determined as crisp value from the defuzzification step.

On the basis of this fuzzy controller the process of transformation with an empty rule base has been started. *DecisionNetworkCompiler* provided the following output:

Table 6.8.: Fuzzy rule base output of the DecisionNetworkCompiler for the "crop problem"

Final report

Quality of calculated RuleBase: 230.07820364277225
Expected Utility Loss: 209.07820364277225
Complexity of calculated RuleBase: 21
 	
# Rules in calculated RuleBase: 11 	
 	
calculated Fuzzy RuleBase: RB "RuleBase","RB",359,157,{
premiseLVs={
"drought", "disease", "bend down", "drought_1",
"disease_1", "bend down_1"
}
conclusionLVs={
"treatment"
}
Rule "..."&"..."&"..."&"..."&"..."&"..." → "true"
Rule "..."&"..."&"..."&"..."&"false"&"..." → "false"
Rule "..."&"false"&"..."&"..."&"..."&"..." → "false"
Rule "..."&"..."&"..."&"false"&"..."&"..." → "false"
Rule "false"&"..."&"..."&"..."&"..."&"..." → "false"
Rule "..."&"..."&"false"&"..."&"..."&"..." → "false"
Rule "..."&"..."&"..."&"..."&"..."&"false" → "false"
Rule "..."&"..."&"..."&"..."&"..."&"true" → "false"
Rule "..."&"..."&"..."&"..."&"true"&"..." → "false"
Rule "true"&"..."&"..."&"..."&"..."&"..." → "false"
Rule "..."&"..."&"..."&"true"&"..."&"..." → "false"
};

The rule base itself in Table 6.8 can be interpreted in the following way: Every line starting with "Rule" represents a rule. Following this there are the assumptions connected with the logical "AND" and finally the conclusion. The order of the single linguistic variables is the same as defined under premiseLVs, respectively conclusionLVs. "..." shows in this context here that for the respective random node no evidence is existent, while "true", "false" and "a bit" refer to the respective linguistic term of the certain random node. Surprisingly

the transformation of the decision network "crop problem" led to the same rule base as the sharp version of the algorithm. This version, of course, provides the same quality as the rule base, consisting of sharp rules, that has been determined in the preceding section. Possibly the stagnation among the quality of the calculated rule base is caused by the low complexity of the decision network. Another reason could be found in the chosen design of the fuzzy controller, thus underlining the great relevance of initialization and adaptation steps in algorithm *TF*. However, the advantage that the calculated fuzzy rule base, connected with the user-customized fuzzy controller, can also handle vague inputs and therefore provides more possibilities for the underlying decision network, is still persistent.

6.7. Evaluation Measures

Analysis criteria for fuzzy rule bases were discussed under various aspects in the past. [Kos92] has introduced probabilistic concepts and definitions useful for comparing fuzzy sets. It is possible to make a point to which degree a fuzzy set is a subset of another fuzzy set. Inconsistency aspects like self reference of rules or loops within the rule base are explained in [LS93]. An affinity measure calculates the consistence of terms, whereas a probability measure identifies the degree of term overlapping. [SMMSPY95] compares different similarity degrees for fuzzy sets based on a geometrical distance model, a theoretical fuzzy set approach, and matching functions. The interested reader can engross the thoughts and achieve comprehensive background information of measures for analyzing fuzzy rule bases in Chih-Tien [CT93] or Karacapilidis & Pappis [KP93].
We have developed new evaluation measures for analyzing fuzzy rule bases obtained by the transformation process, which are not yet published. *Granulation* plays a key role in directly influencing the complexity of the transformed model through partition of the input- and output universe.
Granulation Criteria: At the beginning of the criteria description basic terms are introduced. Given a fuzzy set F characterized by the mapping

$$F : U \rightarrow < 0, 1 >$$

from the universe U in the real-valued interval $< 0, 1 >$. All elements of the universal set are members of a given fuzzy set, therefore, two fuzzy sets may have an overlap in the boundary definitions. The domain $DEV(LV)$ of a linguistic variable is given as

$$DEV(LV) \subseteq \{F | F : U \rightarrow < 0, 1 >\}.$$

Elements of $DEV(LV)$ are labeled with names also well-known as linguistic terms (LT). Consider the temperature of a production machine in a factory. Corresponding fuzzy sets to the *LT low_temp, medium_temp, high_temp* allows any temperature in the interval $[0, 150]$ representing the working temperature of the production machine (e.g. milling machine). As a specific instance, the temperature 94 is a member of all the fuzzy sets but the membership of temperature

(=94) to belong to the sets *low_temp, medium_temp, high_temp* respectively are for instance 0.02, 0.63 and 0.87. The relative grading of the memberships can be easily understood from the usual meaning of the three terms introduced above. A fuzzy rule base $\mathcal{R}$, which contains j linguistic variables, assigns $LT(V_j)$ to LV_j. In other words, $LT(V_j)$ represents the set of linguistic terms assigned to LV_j. The granulation concept is often introduced as *resolution mass* to describe and understand input- and output domains [DHR07]. A high resolution is given in the case of minor rule variations, where $LT(V_j)$ with shortened support are chosen in consequence. In the opposite case a large support is given. This classification is not sufficient to differentiate the granulation understanding. It is more advantageous if a general structuring is achieved. Implement a standard partition with the numerical value one as *standard granulation*. Comparable granulations should numerically lie in $< 0, 1 >$. We understand the granulation term as partition of input- and output universe by use of $LT(LV)$ to structure input- and output domains.
At first formal standard granulation assumptions must be fulfilled. The granulation represents the standard form and takes the value one in the standard case. Suppose that margin sets are trapezoids and all other fuzzy sets of a LV are triangles. The parameters $(x_{a_i}, x_{b_i}, x_{c_i}, x_{d_i})$ are defined as illustrated in Figure 6.10 for trapezoid membership function positions. For triangles, it is obvious that $x_{b_i} = x_{c_i}$ holds. Fuzzy sets represented as singletons are ignored and out of the scope of this thesis.

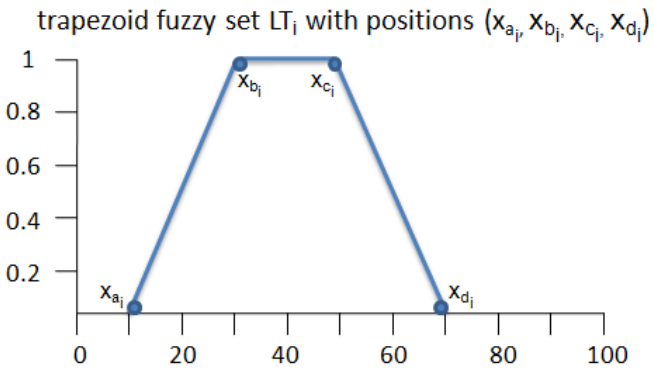


Figure 6.10.: trapezoid fuzzy set of an arbitrary LV

Determine membership functions with *support*, *core* and *co-core* conceptually for LT_i

$$Core(LT_i) = \{x |\ x \in U \wedge LT_i(x) = 1\}$$
$$CCore(LT_i) = \{x \mid x \in U \wedge LT_i(x) = 0\}$$
$$supp(LT_i) = \{x \mid x \in U \wedge LT_i(x) > 0\}\ .$$

The assumptions for the standard granulation are as follows:

(1)

$$\big(Core(1) \cap CCore(2)\big) \bigcap_{i=2}^{n-1} \big(Core(i) \cap CCore(i-1) \cap CCore(i+1)\big) \cap \big(Core(n) \cap CCore(n-1)\big) \neq \emptyset \tag{6.8}$$

$$Core(LT_i) \cup CCore(LT_i) = U \qquad \forall i \in \{1, ..., n\} \tag{6.9}$$

(2)

$$\begin{aligned}
x_{b_i} &= x_{c_i} \quad \forall i \in \{2, ..., n-1\} \\
x_{b_i} &< x_{c_i} \quad \forall i \in \{1, n\} \\
x_{c_1} &= x_{a_{i+1}} \text{ for i=1 and } x_{b_i} = x_{d_{i+1}} \text{ for i=n} \\
x_{b_i} &= x_{c_i} = x_{d_{i-1}} = x_{a_{i+1}} \quad \forall i \in \{2, ..., n-1\} \\
x_{a_1} &= x_{b_1} = U_{min} \text{ with } U = [U_{min}, U_{max}] \\
x_{c_n} &= x_{d_n} = U_{max} \text{ with } U = [U_{min}, U_{max}] \\
x_{a_i} &= x_{b_i} < x_{c_i} < x_{d_i} \quad \text{for i=1} \\
x_{a_i} &< x_{b_i} = x_{c_i} < x_{d_i} \quad \forall i \in \{2, ..., n-1\} \\
x_{a_i} &< x_{b_i} < x_{c_i} = x_{d_i} \quad \text{for i=n}
\end{aligned} \tag{6.10}$$

Next core cardinality and core position definitions are given
(3)

$$\begin{aligned}
Card(Core(LT_i)) &= 1 \quad \forall i \in \{2, ..., n-1\} \\
Card(Core(LT_i)) &> 1 \quad \forall i \in \{1, n\} \\
Core(LT_i) &= x_{d_i} - \Big(\frac{x_{d_i} - x_{a_i}}{2}\Big) = x_{b_i} = x_{c_i} \ \forall i \in \{2, ..., n-1\} \\
Core(LT_1) &= \{x \mid x \in [U_{min}, x_{c_1}] \wedge LT_1(x) = 1\} \\
Core(LT_n) &= \{x \mid x \in [x_{b_n}, U_{max}] \wedge LT_n(x) = 1\} \\
CCore(LT_i) &= \{x \mid x \in [U_{min}, x_{a_i}] \wedge x \in [x_{d_i}, U_{max}] \wedge LT_1(x) = 0\} \\
&\quad \forall i \in \{2, ..., n-1\} \\
CCore(LT_1) &= \{x \mid x \in [x_{d_1}, U_{max}] \wedge LT_1(x) = 0\} \\
CCore(LT_n) &= \{x \mid x \in [U_{min}, x_{a_n}] \wedge LT_n(x) = 0\}
\end{aligned} \tag{6.11}$$

The distance of the core position from the margin cores can be described as
(4)

$$Max(Core(LT_1)) - Min(Core(LT_n)) = (n-1) \cdot \Big(\frac{U_{max} - U_{min}}{n+1}\Big).$$

We determine the x-axis position of linguistic terms with intervals. It follows

$$LT_i : U_i \longrightarrow]0, 1] \quad \forall i \in \{1, ..., n\}$$

with

$$U_i = \Big](i-1)\cdot\frac{U_{max}-U_{min}}{n+1}, (i+1)\cdot\frac{U_{max}-U_{min}}{n+1}\Big[.$$

Next consider the *support* of each linguistic term in the standard case. The following relations hold:
(5)

$$sup\{supp(LT_i)\} = U_{max} - \Big((i-1)\cdot\frac{U_{max}-U_{min}}{n+1}\Big) \forall i \in \{1,...,n\} \text{ with } U = [x_{a_1}, x_{d_n}]$$

$$inf\{supp(LT_i)\} = U_{min} + \Big((i+1)\cdot\frac{U_{max}-U_{min}}{n+1}\Big) \forall i \in \{1,...,n\} \text{ with } U = [x_{a_1}, x_{d_n}]$$

$$sup\{supp(LT_i)\} = inf\{supp(LT_{i+2}) \qquad \forall i \in \{1,...,n-2\}$$

Two adjacent linguistic terms LT_i and LT_{i+1} overlap the x-range of U:

$$sup\{supp(LT_{i+1})\} - inf\{supp(LT_i) = \Big(\frac{3(U_{max}-U_{min})}{n+1}\Big) \qquad \forall i \in \{1,...,n-1\}$$

Consider $\mathcal{G}$ as class of fuzzy partitions composed to Ruspini partitions (see also 6.5) satisfying

$$\sum_{F\in\mathcal{F}} F(X) = 1 \qquad \forall x \in U$$

and

$$\sum_{x\in U} F(X) > 0 \qquad \forall F \in \mathcal{F}.$$

The complete membership degree $\mu_{complete(LV)}$ of a linguistic variable LV containing n linguistic terms for all sharp values over U are calculated as

$$\mu_{complete(LV)} = Min\{Max(F_i(x)) \mid x \in U \ \wedge i \in \{1,...,n\}\} = 0.5.$$

In diverse applications it is necessary to fulfill the following condition

$$\mu_{complete(LV)} = Min\{Max(F_i(x)) \mid x \in U \ \wedge i \in \{1,...,n\}\} > 0.5.$$

Specific sharp x-values in U take for LT_i the same membership degrees $LT_i(x)$. Compare the membership degrees of x for the linguistic terms LT_i with their left and right neighbour terms:

$$LT_i\Big(\frac{x_{d_i}-x_{c_i}}{2}\Big) = LT_{i+1}\Big(\frac{x_{b_{i-1}}-x_{a_{i-1}}}{2}\Big) = 0.5 \qquad \forall i \in \{1,...,n-1\}$$

respectively

$$LT_i\Big(\frac{x_{b_i}-x_{a_i}}{2}\Big) = LT_{i-1}\Big(\frac{x_{d_{i-1}}-x_{c_{i-1}}}{2}\Big) = 0.5 \qquad \forall i \in \{2,...,n\}.$$

The substracted membership degrees at the neighbouring positions are calculated as zero.
Trapezoid membership functions of the margin fuzzy sets LT_1 and LT_n are rep-

resented as flanks of an isosceles triangle. The inner fuzzy sets $LT_2, ..., LT_{n-1}$ are also isosceles triangles. We determine a γ-value to calculate specific membership degrees at concrete x-positions in U. Start at the core position of the inner linguistic terms $Core(LT_i)$ and add or substract γ. We obtain the same membership degrees on the x-axis. Broad this calculation step by step to the other inlying and margin fuzzy sets. The γ-values are calculated as follows:

$$\gamma_i \in \left[0, \frac{x_{d_i} - x_{a_i}}{2}\right] \text{ with } i \in \{1, ..., n\}.$$

Fuzzy margin sets contain designated x-values where the same membership degrees are allocated.

$$LT_1\Big(\Big(x_{d_i} - (\frac{x_{d_i} - x_{a_i}}{2})\Big) + \gamma_i\Big) = LT_n\Big(x_{d_i} - (\frac{x_{d_{i-1}} - x_{a_{i-1}}}{2})\Big) - \gamma_i\Big).$$

The determination of the same membership degrees for $x \in U$ is also discussed in [KGK95]. A horizontal and vertical view on fuzzy sets is given. Specific level sets A_i with $i \in \{0, ..., 1\}$ are declared and collateral to the x-axis for each $(0, i)$ represented. A decision maker is now able to specify γ_i which facilitates the calculation of membership degrees for designated $x \in U$.
Now an instrument is given to apply the granulation determination based on partition of input- and output universe. The granulation criteria is refined in integrating the following analysis topics:

- Support of the linguistic terms in relation to the standard granulation demands
- Evaluation of the LT_i cores
  - Core size for LT_i with $i \in \{1, n\}$
  - Core size for LT_i with $i \in \{2, ..., n-1\}$
  - Core position for LT_i with $i \in \{1, n\}$
  - Core position for LT_i with $i \in \{2, ..., n-1\}$
- Position of the LT_i for linguistic variables LV

The results of support, core and position calculation are presented separately. After that, the user runs a chance to weight this intermediate values for completing the granularity estimation.

Support of the linguistic terms:
The author of this thesis has developed a formula to calculate the covering of the linguistic terms in comparison with the standard partition case. We proceed on the assumption that maximal doubling of the support of each LT_i is valid. That is, with each doubling of the support the receiving value of f_{LT} for this LT_i is equal to zero. All LT_i count as equal with $\frac{1}{n}$ in f_{LT}:

$$f_{LT} = 1 - \sum_{i=1}^{n} \frac{1}{n}\Big[\frac{|(x_{d_i} - x_{a_i}) - \frac{2(U_{max} - U_{min})}{n+1}|}{\frac{2(U_{max} - U_{min})}{n+1}}\Big].$$

Consider the situation, in which each LT can possibly cover the complete universe, we have to calculate $f_{\overline{LT}}$:

$$f_{\overline{LT}} = 1 - \sum_{i=1}^{n} \frac{1}{n} \Big[\frac{|(x_{d_i} - x_{a_i}) - \frac{2(U_{max} - U_{min})}{n+1}|}{(U_{max} - U_{min}) - \frac{2(U_{max} - U_{min})}{n+1}} \Big].$$

Figure 6.11 illustrates top left the standard case with ideal support and maxi-

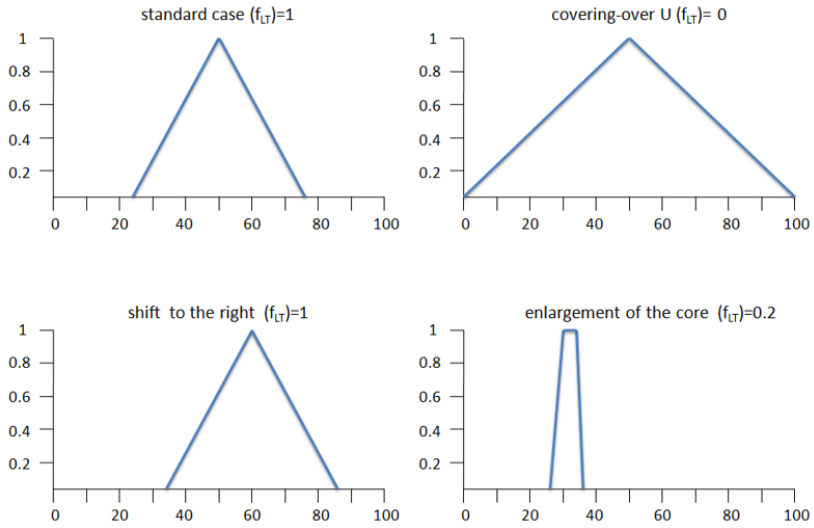


Figure 6.11.: Support of one linguistic term compared to the standard case

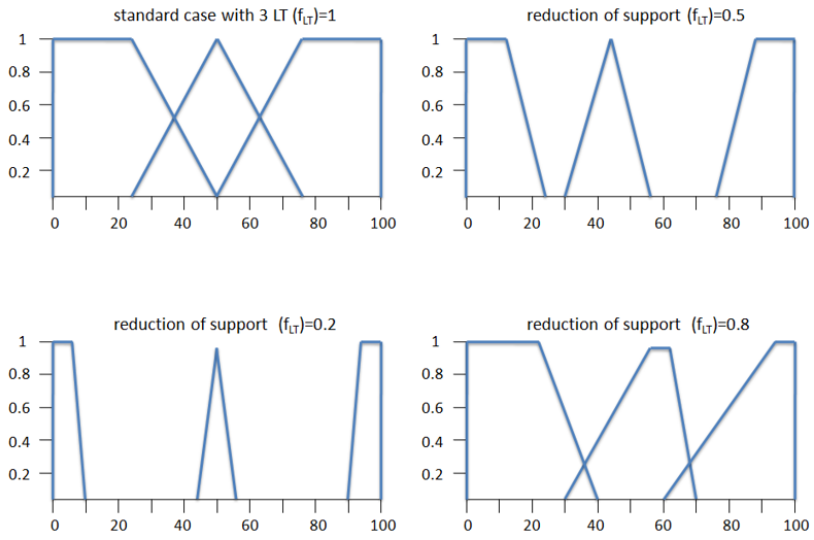


Figure 6.12.: Support of 3 linguistic terms compared to the standard case

mal f_{LT}. Shifting this triangle LT to the right (or to the left) has no influence on the calculated support output. At the top on the right we can see a linguistic

term, which covers the complete universe U and the zero output is received. A reduction of the support and simultaneously enlargement of the core is visualized as fourth case at the bottom on the right. Figure 6.12 illustrates three situations in each case with three linguistic terms we have to consider to determine the support value. Top left the standard case is visualized with expected outcome f_{LT}=1. The other cases clarify the effect when reducing the support of all linguistic terms by 80%, 50% or 20%.

Core size and core position :
We will next study the core position and core size of the linguistic terms. The core size examines the cardinality compared to the standard case. Consider that a margin fuzzy set covers the support range halfway through. All other fuzzy sets have cardinality one as $Card(Ker(LT_i)) = 1 \ \forall i \ \in \{2, ..., n-1\}$. By now, we take no account of the core position.

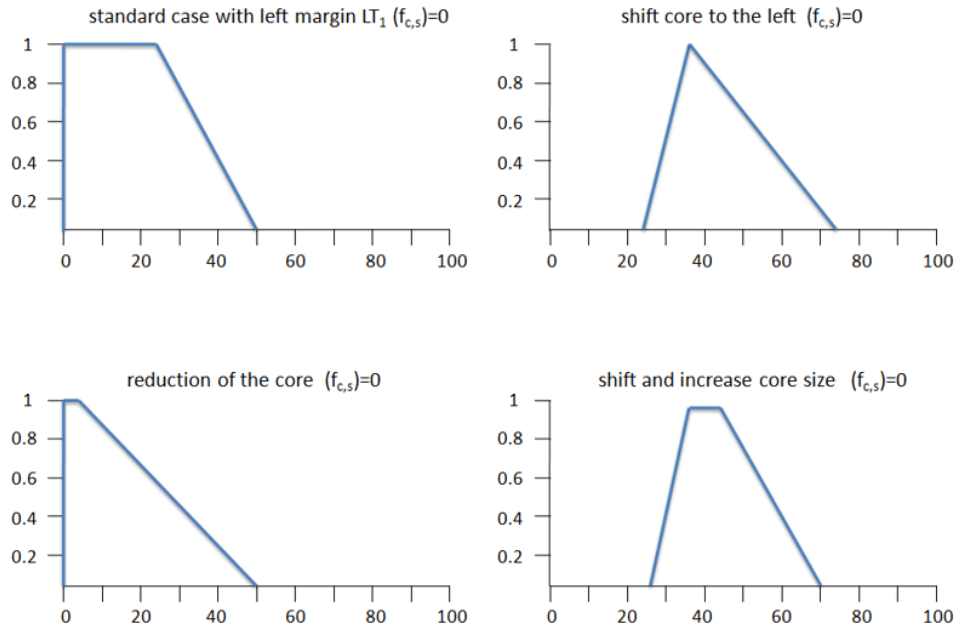


Figure 6.13.: Core size of margin and inner linguistic terms compared to the standard case

It is possible to weight the core size and core position differentially. We define $\frac{1}{\kappa}$ as standard mass for covering-over the support of a LT by the core, e.g. κ=2 for margin sets. The core size function is given as

$$f^i_{c,size} = \left[\frac{|(x_{c_i} - x_{b_i}) - \left((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\right)|}{(x_{d_i} - x_{a_i}) - \left((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\right)}\right] \ \text{for} \ \ i \in \{1, n\}$$

with $\kappa \geq 2$ and respectively

$$f^i_{c,size} = \Big[\frac{|(x_{c_i} - x_{b_i}) - \Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)|}{\Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)}\Big] \text{ for } i \in \{1, n\}$$

with $2 > \kappa \geq 1$ and

$$f^i_{c,size} = \frac{x_{c_i} - x_{b_i}}{x_{d_i} - x_{a_i}} \text{ for } i \in \{2, ..., n-1\}.$$

Each $f^i_{c,size}$ represents the core size in comparison to the standard-LT_i case. From this it follows from a holistic point of view

$$f_{c,size} = 1 - \sum_{i=1}^{n} \frac{1}{n}\Big[f^i_{c,size}\Big].$$

A second important aspect measures the position of the core of LT_i. Margin set core positions correlate to $\frac{1}{\kappa}$-fold of the support respectively. The left margin set corresponds to the $\frac{1}{\kappa}$ of the left support and vice versa:

$$f^i_{c,pos} = \Big[\Big(\frac{x_{b_i} - x_{a_i}}{x_{d_i} - x_{a_i}} + \frac{|x_{c_i} - (x_{a_i} + \Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)\Big)|}{(x_{d_i} - x_{a_i}) - \Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)}\Big) \cdot \frac{1}{2}\Big] \text{ for } \kappa \geq 2$$

and

$$f^i_{c,pos} = \Big[\Big(\frac{x_{b_i} - x_{a_i}}{x_{d_i} - x_{a_i}} + \frac{|x_{c_i} - (x_{a_i} + \Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)\Big)|}{(x_{d_i} - x_{a_i}) - \Big((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}\Big)}\Big) \cdot \frac{1}{2}\Big] \text{ for } 2 > \kappa \geq 1.$$

The first addend compares the left core margin, the second addend the right core margin with the standard case of the left margin fuzzy set. The weight is cut into halves.
It follows for $i = n$

$$f^i_{c,pos} = \Big[\Big(\frac{|(x_{d_i} - (x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}) - x_{b_i}|}{(x_{d_i} - x_{a_i}) - ((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa})} + \frac{x_{d_i} - x_{c_i}}{x_{d_i} - x_{a_i}}\Big) \cdot \frac{1}{2}\Big] \text{ for } \kappa \geq 2$$

and respectively

$$f^i_{c,pos} = \Big[\Big(\frac{|(x_{d_i} - (x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa}) - x_{b_i}|}{((x_{d_i} - x_{a_i}) \cdot \frac{1}{\kappa})} + \frac{x_{d_i} - x_{c_i}}{x_{d_i} - x_{a_i}}\Big) \cdot \frac{1}{2}\Big] \text{ for } 2 > \kappa \geq 1.$$

The core position of the inner fuzzy sets should be in the middle of the support. Shifting this position to the left or right direction directly influence $f^i_{c,pos}$. It

follows for the calculation of the inner fuzzy sets $i \in \{2, ..., n-1\}$

$$f^i_{c,pos} = \Big[\frac{|x_{c_i} - \Big(x_{d_i} - \frac{x_{d_i} - x_{a_i}}{2}\Big)| + |\Big(x_{a_i} + \frac{x_{d_i} - x_{a_i}}{2}\Big) - x_{b_i}|}{x_{d_i} - x_{a_i}}\Big].$$

To determine $f^i_{c,pos}$, we have to measure the difference values x_{b_i} and x_{c_i} compared to the standard position. It can happen, that for instance shifting the core 25 percent to the left with kept cardinality one becomes more accountable concerning the degree of variance with regard to the standard case than shifting the core 25 percent to the left and simultaneously increasing the core cardinality (Card ≥ 1). In the former case, x_{b_i} and x_{c_i} are located 25% away from the standard value. In the latter case, the enhancement of the core cardinality causes a x_{c_i}-value, which is located nearer to the support center. This effect

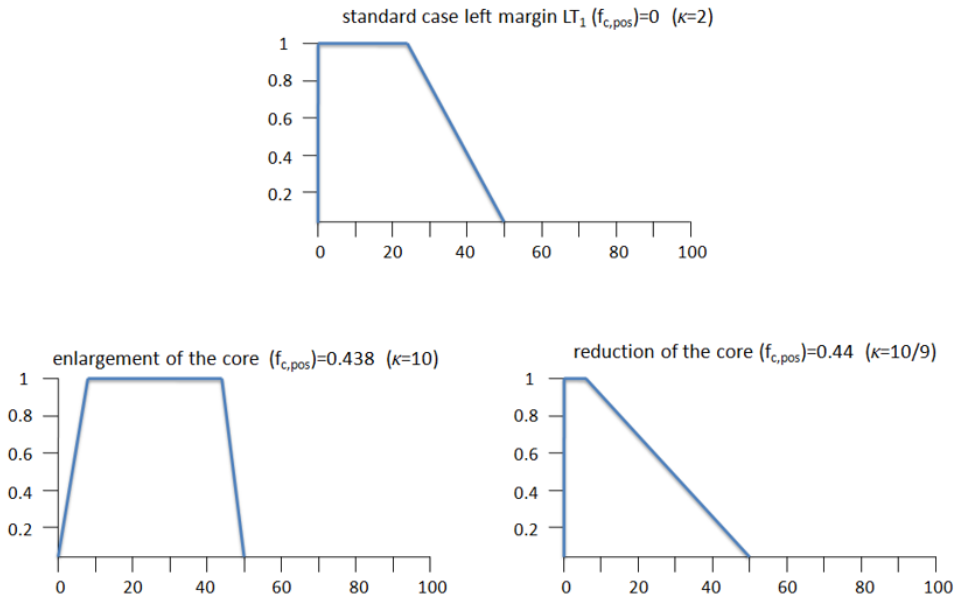


Figure 6.14.: Core position of margin linguistic term LT_1 compared to the standard case

is illustrated in Figure 6.15 at lower left (only shifting the core to the left) and at the top on the right (shifting and simultaneously enlarging the core of the inner linguistic terms). By now, we are able to integrate a weighting factor to measure the core variance on the whole:

$$f_{core} = \nu_{size}\Big(1 - \sum_{i=1}^{n} \frac{1}{n}\Big[f^i_{c,size}\Big]\Big) + \nu_{core}\Big(1 - \sum_{i=1}^{n} \frac{1}{n}\Big[f^i_{c,pos}\Big]\Big) \text{ with } \nu_{core} + \nu_{size} = 1.$$

Position of the linguistic terms:
We will next study the position of the linguistic terms, which depends on the position of the support of each linguistic term LT_i within the universe U. Either we shift linguistic terms on the x-axis or enlarge or reduce the support area.

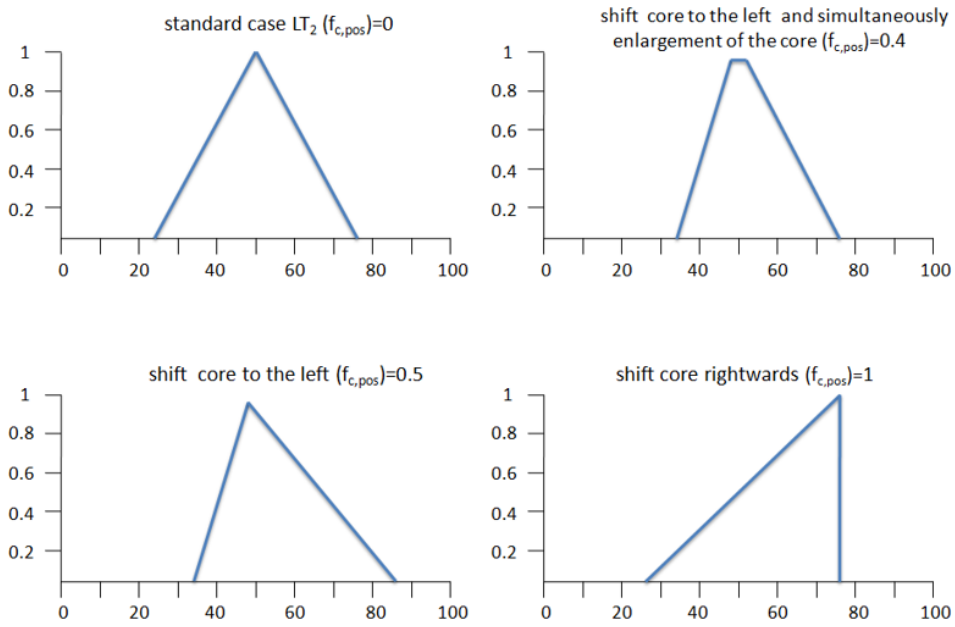


Figure 6.15.: Core position of inner linguistic terms compared to the standard case

In the former case it takes no effect to the standard case, in the latter case the granularity value concerning the position of LT_i decreases. At first, we have to declare the valid covering degree of the linguistic terms. It should be allowed to shift a linguistic term one position to the left or to the right or to differ from the standard support by maximally doubling the support area. The error boundary value is achieved at

$$\frac{2(U_{max} - U_{min})}{n+1}.$$

Note that U_{max} and U_{min} are valid within the following conditions:

$$\begin{array}{lll} U_{max} > 0 & \text{and} & U_{min} \leq 0 \\ U_{max} \leq 0 & \text{and} & U_{min} < 0 \\ U_{max} < 0 & \text{and} & U_{min} < 0 \end{array}$$

A maximum doubling of the support of LT_i depending on the x_{a_i} and x_{d_i} standard position is allowed. In other words, x_{a_i} and x_{d_i} can differ from their standard position maximal equal to their support value. Maximal shifting the inner linguistic terms one position to the right from LT_i to LT_{i+1} and respectively to the left from LT_i to LT_{i-1} for $i \in \{2, ..., n-1\}$ is possible. Margin sets can be shifted from LT_1 to LT_2 and LT_n to LT_{n-1}. The last important point occurs in the case of maximal shifting LT and simultaneously modifying the support area. We have to calculate an enlargement factor *enlfac* and reduction factor *redfac* depending on the modification of the support area.

Case 1: Shift x_{a_i} and x_{d_i} to the right ($\forall i \in \{1, ..., n-1\}$) or to the left ($\forall i \in \{2, ..., n\}$). In this case we are able to maximal duplicating the modified

support without violating the valid range. Else, the granularity value for the LT position exceeds one.
Case 2: Shift x_{a_i} to the right and x_{d_i} to the opposite left direction ($\forall i \in \{1, ..., n\}$). Take the multiplier for $\frac{U_{max}-U_{min}}{n+1}$ as γ value into account. We are able to calculate the enlargement factor including γ as presented in Table 6.9. The enlargement factor for the determination of the modified support

Table 6.9.: Maximal enlargement factor *enlfac* on the basis of modified x_{a_i} and x_{d_i}

Modification of x_{a_i} and x_{d_i}	*enlfac*
$3/4 \cdot \gamma$	8
$1/2 \cdot \gamma$	4
$1/3 \cdot \gamma$	3
$1/4 \cdot \gamma$	8/3
$1/5 \cdot \gamma$	5/2
$1/n \cdot \gamma$	$\frac{2}{1-\frac{1}{n}}$

of linguistic terms can be expressed as follows:

$$enlfac = \frac{2}{1-\gamma} \qquad \forall i \in \{1, ..., n\}.$$

Case 3: Shift x_{a_i} to the left and x_{d_i} to the right ($\forall i \in \{2, ..., n-1\}$). The calculation of *enlfac* for inner linguistic terms is given as

$$enlfac = \frac{2}{1+\gamma} \qquad \forall i \in \{2, ..., n-1\}.$$

The determination of the modified values for x_{a_i} and x_{d_i} arises from the closed interval $\left[0, \frac{U_{max}-U_{min}}{n+1}\right]$. It is conceivable to specify a reduction factor *redfac* which in fact decrease the support area. The consideration of *redfac* is similar to the enlargement case. The interested reader can engross the thoughts and add up again the three cases mentioned before at this point for the reduction factor determination. With the previous cogitations, we are able to declare a function $f^i_{LT,pos}$ which measures the variation of the standard case in shifting the LT_i one position to the left LT_{i-1} or one position to the right LT_{i+1}. At this, a modification of the support is only allowed within the predefined valid boundary of maximal term shifting:

$$f^i_{LT,pos} = \Big(1/\frac{2(U_{max}-U_{min})}{n+1}\Big) \cdot |x_{d_i} - \Big(U_{min} + \Big(\frac{U_{max}-U_{min}}{n+1} \cdot (i+1)\Big)\Big)| + \Big(1/\frac{2(U_{max}-U_{min})}{n+1}\Big) \cdot |x_{a_i} - \Big(U_{min} + \Big(\frac{U_{max}-U_{min}}{n+1} \cdot (i-1)\Big)\Big)|.$$

It follows the total calculation of the position of linguistic terms with $f_{LT,pos}$

as

$$f_{LT,pos} = 1 - \sum_{i=1}^{n} \frac{1}{n} \left[f^{i}_{LT,pos} \right].$$

If required, each linguistic term LT_i can weight slip in the final calculation on the basis of a weighting factor $w_{LT,pos}$. Figure 6.16 illustrates three examples regarding the left margin set, in which the enlargement of the core size with simultaneously maintained position and the shifting process to the right is visualized. The holistic view concerning the granulation aspect includes analyze

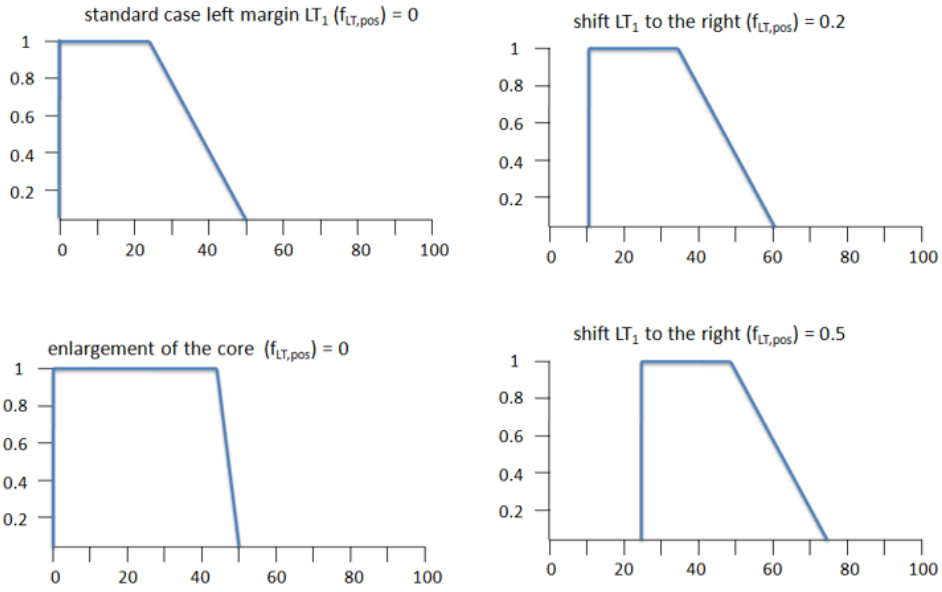


Figure 6.16.: Position of margin linguistic term LT_1 compared to the standard case

the support

$$f_{gran} = 1 - \left[(f_{LT} \cdot w_{LT}) + (f_{core} \cdot w_{core}) + (f_{LT,pos} \cdot w_{LT,pos}) \right]$$

with $0 \leq w_{LT}, w_{core}, w_{LT,pos} \leq 1$ on condition that $w_{total} = w_{LT} + w_{core} + w_{LT,pos} = 1$. Finally, we are able to assess a special case (see Figure 6.17 on

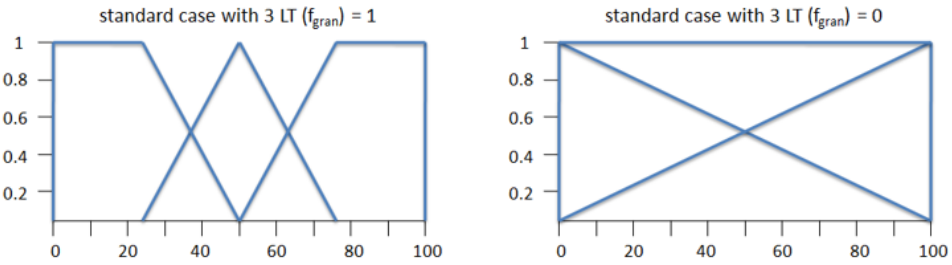


Figure 6.17.: Position of margin linguistic term LT_1 compared to the standard case

the right hand side), where the core of the inner set LT_2 overlaps the complete support area. The support of each linguistic term is equal to the universe U. The core position of the margin sets are located at the opposite universe margins. This case has opposed measuring characteristics compared to the standard case and the granularity value results zero.

6.8. Transformation Process Flow

We will now introduce and run through the transformation process we have used in this thesis to obtain a fuzzy rule base. The various techniques and algorithms we have previously introduced for building Bayesian networks, Decision networks and rule base representations are tied together. From the knowledge engineering point of view, it is very important that the transformation process be properly managed to achieve the desired representation. Participants are mostly the knowledge engineer and domain expert. The former must learn about the problem domain and the last-named must understand what different model representations are and what they can do. Communication plays a critical issue to be productive. The participants can start by constructing simplified models to build mutual understanding. As knowledge engineering proceeds, both the domain expert and the knowledge engineer's understanding of the problem domain deepen. For modeling decisions, it is important to ensure that there is consistency across different parts of the model. The author of this thesis has marked all new research approaches in Figure 6.18 as green-coloured abstract process symbols. Each process step is described in detail next.

- **Learn Bayesian network structure:** Opening with the first transformation process step, we are looking for methods for eliciting Bayesian network structures. Here we presuppose only that we have a set of random variables with an unknown distribution. Learning the network structure from observations is a typical example. For instance, in the medical domain, a great deal of observational data is contained in collected electronic medical records. In the case of missing data, sampling methods to approximate the probability of data containing missing items are available. Apply the new LAGD structure learning algorithm, which is introduced in Section 4.3.2, to obtain a Bayesian network structure which clarifies the conditional (in)dependencies.

- **Create Decision network:** In the next process step, the participants are able to extend Bayesian networks with two kinds of nodes, which are ascertainable by preference elicitation and specification of the user's probability information about alternatives in a decision situation. These two kinds of nodes are decision nodes representing decisions to be made and utility nodes whose possible values are the utilities of the outcomes. Utility nodes are closely linked to preferences of an individual. The term preference elicitation refers to the problem of finding out about the preferences of a single individual. Numerical representations of preferences are given by utility or value functions. Specifying standard information

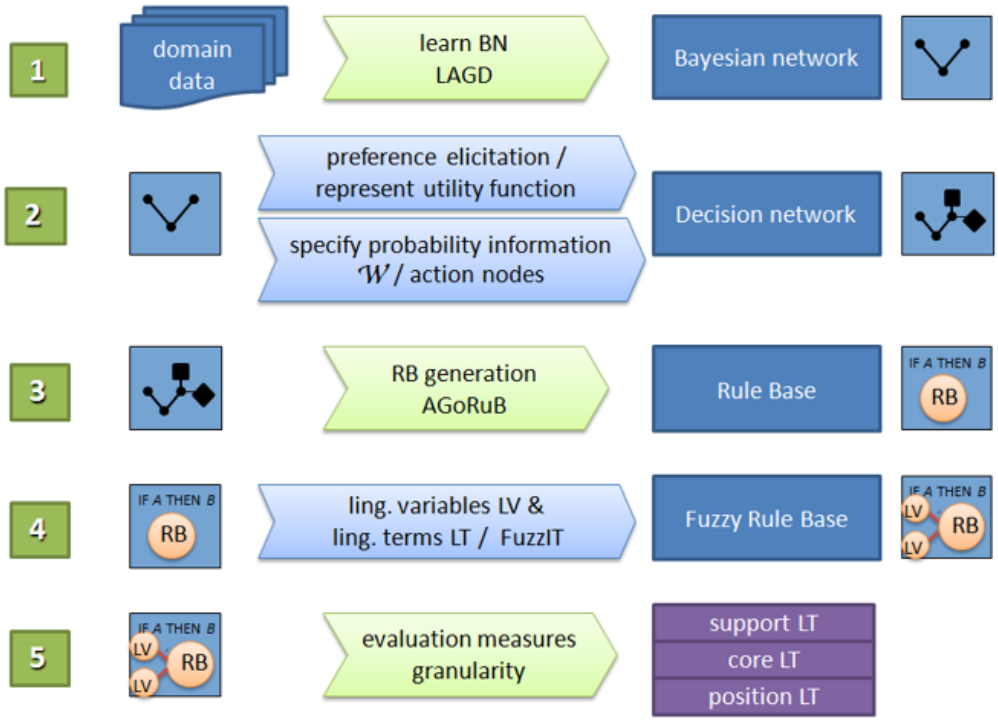


Figure 6.18.: Transformation process flow from data acquisition to rule base generation

$\mathcal{W}$ eliminates all non-efficient and redundant actions (see figure 5.6) in a decision situation to obtain a shortened set of alternatives and finally condensed set of decision nodes. After this process step is fulfilled, a focussed Decision network is arrived.

- **Automatic generation of a rule base:** The described algorithm for automatic generation of a rule base by means of a decision network is applied next. We are able to calculate outcomes on the basis of the underlying decision network and simultaneously on the basis of the generated rule base. Quality and complexity measures are involved and deal as explanation component for the expected utility loss.
- **Determine linguistic variables and linguistic terms:** At this point, the available fuzzy inference tool named FuzzIT allows vague observations to be included into the transformation and finally decision process. A controller component aligns a fuzzy controller and provides methods for fuzzification, inference and defuzzification.
- **Evaluate fuzzy rule-base:** The chosen set of linguistic variables and linguistic terms is studied on the basis of new evaluation measures, which compare selected sets to standard cases. Granularity measures are introduced concerning the core, size and position of linguistic terms.

To sum up, there is interplay between structure representations and parameters. To augment the classical KEBN process, which sketches a framework and

describes in more detail methods to support at least some of knowledge engineering tasks, a transformation process flow to model rule bases and to include degrees of vagueness is given.

6.9. Transformation Review

A methodology for transforming Decision networks in an appropriate representation form is described in this section. The transformation procedure is applied to the problem of automatically generating rule-based models from Decision networks. Models generated by this approach have much flexibility, understandability and usability based on their ability to explain and represent knowledge in an intelligible manner. A quality measure calculates the expected utility loss and further the complexity of the rule base, which offers a decision maker an instrument to comprehend the meaning of the best fitting rule base output. In addition, a fuzzy inference tool is available and enables to design a fuzzy controller to further use it for inference. This controller aligns linguistic terms of the linguistic variables, which can be evaluated based on granularity criteria. Granularity directly influences the complexity of IF-THEN rules of the input- and output universe. An introduced transformation process assists a decision maker from the first stage corresponds to gather data in the problem domain to the desired generation of fuzzy rule-based systems. The obtained fuzzy rule-base provides an efficiently realization architecture and can be evaluated for vague observations, whereas Decision networks requires evidence variables to be precisely specified. The implementation of the transformation algorithm is described and the approach is applied to the modeling of components of stock exchange and crop problem in agriculture.

Part IV.

Studies and Applications

7. Product Lifecycle Management

With increasing competitive pressure, manufacturing systems are being driven more and more aggressively. Manufacturing systems and processes are becoming increasingly complex, making more rational decision making [EW03] in process control a necessity. In current industrial practice, quality is ensured in the product engineering cycle at the product design stage and the process control methods at the inspection stage [ADEK05]. A third level of quality assurance can be implemented during machining in process [GW06]. Product lifecycle management (PLM) has become one of the key technological approaches and enablers for the effective management of product development and product creation processes [Abr06]. To achieve successful new and innovative products PLM helps to improve the efficiency and quality by supporting every stage of the product's life from the company's portfolio management to product development, product manufacturing, ongoing maintenance and finally retirement or recycling. Customers and users believe that PLM systems should especially concentrate on the downstream phases of the product lifecycle. Integrating the product use phase into the PLM concept represents a necessary enhancement of the conventional product type PLM in view of managing product item data. Future systems that are based on the exposed concept are to enable producers, customers and service providers to derive new use models on the product type and item level in a distributed knowledge network and to subsequently generate scalable business models. These business models are to establish beside innovative concepts for maintenance, servicing and availability efficiency feedback mechanisms, incorporating condition monitoring results from the product use phase of the last product generation in a target oriented fashion in the development of the next product generation. Within the scope of a feedback cycle this provides a basis for faster product improvements. A schematic of monitoring and control of machine processes including in-process quality control at the machine stage is therefore a prerequisite to reduce production cost and to automate many operations and functionalities. Enhanced solutions integrate the product information from design and engineering step with sourcing, compliance, suppliers, and complete supply chains to decrease product development time, guarantee quality and therefore contribute sustained company profitability and customer loyalty. Relationship management for identifying and linking will be established on the basis of product information management on the product item level. Users can access and process data on numerous granularity and compression levels using a viewing and access concept.

7.1. Basic Components of the PLM

In the past years product lifecycle management (PLM) has become one of the technological and organizational key approaches that allow to effective manage product development and manufacturing processes in design engineering and in the industry. PLM is an integrated approach that is made up of a consistent set of methods, models and IT tools for managing product information, construction processes and applications along the various phases of the product lifecycle [SBM+05]. It is not only aimed at a single company, but at globally distributed, interdisciplinary collaborations between manufacturers, suppliers, partner companies and customers [ASI07, Mil03]. PLM developed from the 90's product management (PDM) approach [Sch99]. Whilst PDM has a restricted focus on managing product data in the product development phase [ES08], PLM focuses on incorporating all product data, processes and applications in the entire life of a product (see Figure 7.1) [AS06]. PLM is more than administrating and characterizing values and properties of a product through its development and life. From the mechanical and business engineering point of view PLM helps analyzing the product operations and uses in the market with respect to quality and financial measures [HTB02] that facilitate the integration of new advanced product development processes.

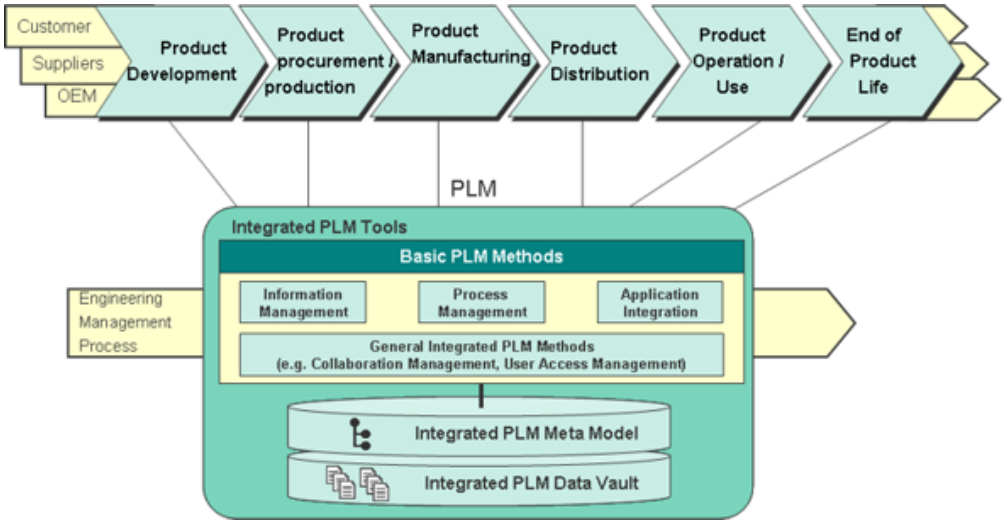


Figure 7.1.: Basic components of the product lifecycle management approach [AS06]

The core of the PLM approach lies in the integrated data and process metamodel, which is managed by a database management system and a central data archive for saving all proprietary models and documents (e.g. CAD models and text documents). Available PLM methods and tools can be split into the following three groups [Abr07]:

- **Information management:** methods for identifying, structuring, classifying, retrieving, splitting, spreading, visualizing and archiving product-, process- and project-based data.

- **Process management:** methods for structuring, planning, handling, and controlling formal and semi-formal processes, such as releasing designs and revision, change and notification processes [Bür01]. The close connections between the various process steps and the resulting product models are covered by so-called configuration management methods and tools.

- **Application integration:** methods for defining and managing interfaces between PLM and various source systems, such as CAD, CAM, CAE and integrated company software, e.g. ERP, SCM, CMS, DMS or CRM systems.

In addition to the basic PLM methods and tools, the PLM approach includes a set of general complementary PLM methods and tools, e.g. engineering collaboration support, user access management and data analysis, reporting and visualizing [MQB07]. The described PLM approach is conceptualized and designed as a framework, which can be used as a reference for creating company-specific PLM concepts and implementations. The focus of all available PLM solutions currently lies on product development activities [AS04]. Generally all current PLM solutions offer generic and preconfigured templates for data models, processes and functions for specific domains and applications. The strengths of available PLM solutions are in managing CAD models and technical documents; supporting construction releases and change processes and the close integration with CAD and ERP systems. The main weaknesses of existing PLM solutions lie in the insufficient support of product lifecycle activities beyond the development phase, as well as in the integration of service components. A further problem is their high complexity and the necessary, very high adaptation efforts. There are no generally acknowledged industrial standards for PLM meta-data models and PLM processes, despite intense standardization activities. Even though the PLM approach is not new and there are already a number of PLM solutions available on the market, only 8 % of companies having a clear-cut vision of PLM and implement wide ranging PDM/PLM systems. Around 50 % of companies implementing PLM are still in the beginnings [AS04]. The existing, described PLM meta-models, methods and tools form a core platform for continuing PLM improvements, enhancements and new developments. It is necessary to classify the multi-dimensional development frame because PLM is a very complex, distributed and interdisciplinary subject. These are the development directions [Abr07]:

- a general instantiation approach for various industrial sectors or application domains,

- considered PLM users or partners,

- covered product lifecycle phases,

- supported types of processes,

- covered types of products.

This work focuses on the direction of development concerning the "covered phases in the product lifecycle". In contrast to existing PLM solutions that are restricted to supporting the product development phase, the PLM models, methods and tools currently being developed by the authors of this paper are to support the management of downstream product lifecycle phases. Especially the feedback of information from the product use phase to the product development phase is to be established. The bidirectional flow of information between the two phases forms the basis for faster product improvements and a more efficient service (e.g. more dependable maintenance forecasts, more efficient fault analyses and thus higher availability). This is achieved through the target-oriented integration of so far unused data from the product use phase. There are numerous reasons for feeding back the so-called field data from the product use phase in the product development. On the one hand, the underlying data can essentially be seen to be realistic. On the other hand the obtained information from the field comes directly from the relevant markets for the product and therefore they mirror the customer's "mood". The use of field data in product development is also a source for customer requirements, that has an entirely different quality of information than, for example, market research or cost and time-consuming customer questioning, because it based upon concrete experiences related to the use and handling of existing products [Edl01].

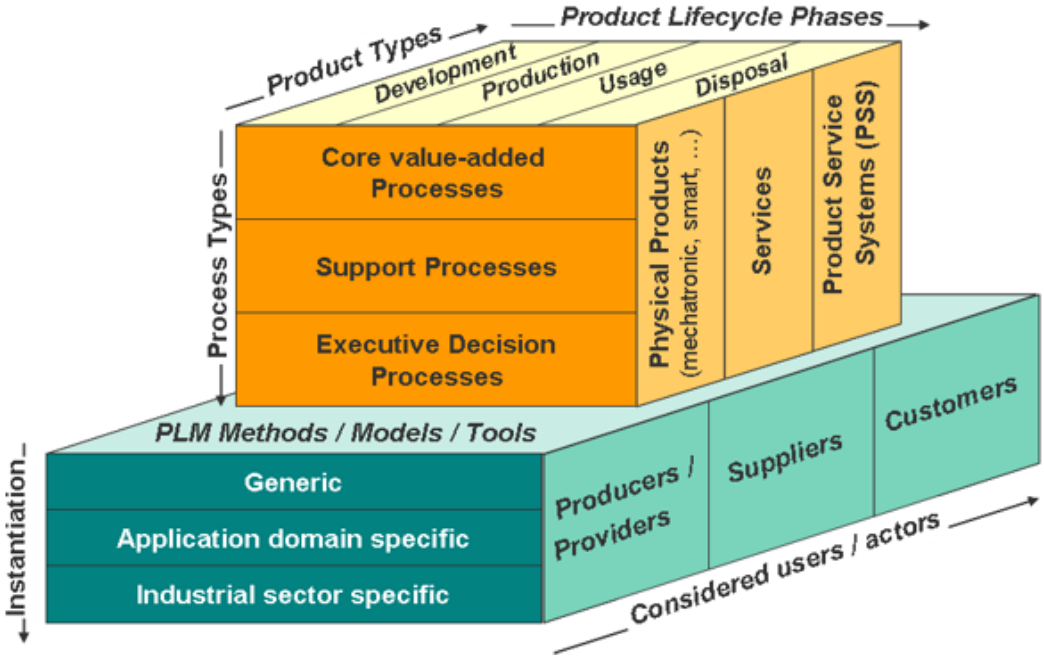


Figure 7.2.: Development taxonomy for future PLM methods, models and tools [Abr07]

Figure 7.2 illustrates a development taxonomy for future PLM methods, models and tools including own coordinates for product lifecycle phases, product types, process types, instantiation and considered users and actors, where customer requirements can be represented very efficiently and fruitfully.

Based on the development taxonomy and essential underlying information and process management supposable and expected development directions are derivable, which are also used as fundament for collaboration engineering.

7.2. Condition Monitoring

Today's production machines are extremely cost-intense and cause high fixed cost, whereby their efficient use is defined by low operating cost and low maintenance cost [Wan07]. It is possible to measure informative machine parameters for safety, efficiency and maintenance purposes using sensors to detect component relationships by regularly acquiring the machine parameters during operation [BKKL05] for condition monitoring. The sensors need to be capable of delivering, alongside exact control variables, blurred, flawed, vague or inexact data [Wan06] and representing this downstream.
Condition monitoring and diagnosis of machinery and processes [Dav98] have become established by reducing sudden failure of machinery, reducing downtimes for repair and improving reliability [For05]. Maintenance work often includes major machine replacements or upgrades, which are really capital intensive work projects. Fundamental distinctions are the two types of maintenance. Breakdown maintenance operates as repair at failure, whereas preventive maintenance types take some actions with the aim of preventing failure occurring or at least minimize the chance of failure. Reliability of a machine measures whether it does what it is required to do whenever it is required to do so [Mou01]. Reliability is the probability that a machine will remain online producing as required for a desired time period in a statistical manner. The application of quantified and qualified methodologies and techniques in condition monitoring and diagnosis plays a leading role in the development of intelligent manufacturing systems.
Although the technology and implementation of condition monitoring and diagnosis are continually developing, its fundamental principles can be traced back to the usage of a priori information by using environmental parameters, machine parameters and also human senses of machine experts to monitor the state of machines and to find their failures. Monitoring processes (see Figure 7.3) are neither automatisms for parameter cause-effect relationships, nor do they form models that predict incalculable or sudden events regarding the considered machine [Hol06b]. The usage of representations dealing with a priori information allows quantifying the condition of industrial equipment. In that case early diagnosis can be executed and corrected by suitable maintenance steps before causing plant breakdown [CdC07].
Condition monitoring therefore involves designing and using sensing arrangements of production machines, alongside with data acquisition, analysis and decision making methods [WCC06] with the objective of implementing equipment maintenance in a planned way using actual knowledge. This is the reason why the application of a modern monitoring and diagnosis system for production is becoming one of the urgent requirements that is necessary to improve the level of system intelligence through the utilization of graphical models and

systems. This enables the equipment to survive in an uncertain environment. Conventional techniques for mechanical equipment monitoring usually involve establishing certain kinds of quantified mathematical and statistical descriptions about the detected symptoms and the possible problems, which tend to be difficult and time consuming when the object's behaviour and detected signals become complicated.

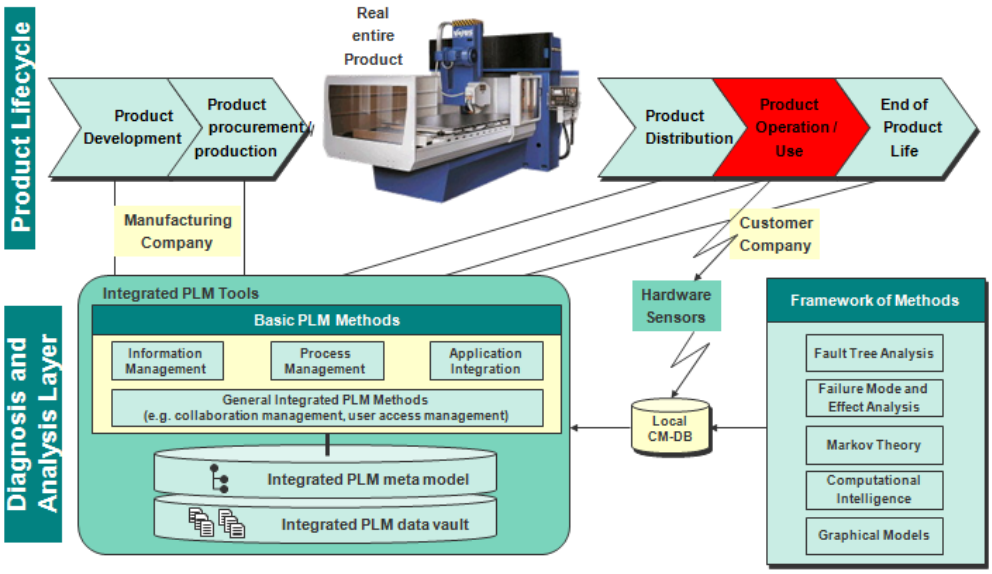


Figure 7.3.: Condition monitoring results and their coupling with integrated PLM tools for decision making

The primary fields of application for preventing unplanned downtime and diagnosing damage are, for example, pumps ([Bee04],[GZ06]), turbines [TK04], clutches or gearboxes. Clutches suffer from abrasion, lack of lubricants, fouling, corrosion or the consequences due to operating errors such as temperature rises or increased friction. Further areas of application for condition monitoring are wafer production ([GRFM06],[KKG+06],[MT06]) in the semiconductor industry ([CSHL05],[MGI06],[MS06]), casting processes, power machines ([Wol05],[HMS+06]), machine tools, engines or railcars [Guo05]. Microsystems technology in industrial engineering makes it possible to acquire information in real-time [LCZL07] and to make this available for application processes [LY05] through IT-based networking. New services, such as remote diagnostics, supplying spare parts or autonomous sensor network integration can be used to increase availability, flexibility, accuracy or reliability which permit new lines of business [SGGB07]. Several steps are required for implementing CM measures:

- significant production machine state variables and redundancies are to be established,
- singular states need to be measured and documented,
- current states have to be matched with nominal values and thresholds,

- methods and processes for diagnosing error sources precociously and determining the causes [EJM06], to optimally control and carry out maintenance processes [Mai06] must be implemented.

The result of optimized maintenance processes based on machine parameter analysis processes can reduce downtime as well as the following impact on further intermediate and end products in downstream production stages [Wei06] with the according follow-up costs regarding the product's reliability [WP06]. Quantitative methods - based on mathematical statistics, probability theory and qualitative methods based on a systematic, experimental localization of weak spots and their following impacts - can be implemented for reliability analyses [AFHN07].

7.2.1. Condition Monitoring Method Framework

In the past years, different effective monitoring techniques have been developed [RJO05] for mechanical machinery, monitoring and diagnosis, such as visual inspection or surface defect detection. Condition monitoring techniques focused on how to extract the pertinent signals of features from the equipment health information. The main question is how to analyze and utilize this information patterns in modern mechanical equipment. The main characteristics of modern mechanical equipment are high working speed with complex and flexible structures and dynamic working environment. Explicit and reasonable monitoring and diagnostics decisions are often required in real time, where the available mechanical equipment information is incomplete, uncertain or conflicting in nature. Under such situations, human decisions and the consideration of a priori machinery and environmental information is a possible solution. The usage of a priori information in Bayesian networks or other graphical models (e.g. influence diagrams with action and utility nodes) is established [KN04]. In recent years considerable progress has been made in the area of probabilistic graphical models. They have become mainstream in the area of uncertainty in artificial intelligence [RN03], where human decision is usually one of the best solutions in the field of mechanical equipment health information. Experts can initialize network structures with a priori information and learn from data and probabilistic inference. This includes themes such as the characterization of conditional independence in mechanical equipment and the sensitivity of the underlying probability distribution of a Bayesian network to variation in its parameters. A new approach by the author of this thesis makes use of the learning of graphical models with latent variables and extensions to the influence diagram formalism [Hol06a]. The new condition monitoring framework consists of analytical methods, statistical methods and graphical methods [HFAN07] for detecting abnormal situations. Critical operating parameters occur as warnings or alarms for the production machine. Advanced functions for diagnosis are installed to test the truthfulness of alarming and warning signals to diagnose prior to failure the causes of negative trends in the equipment and to provide corrective actions to be taken. The condition monitoring framework consists of the following components illustrated in Figure 7.4:

Fault Tree Analysis: FTA methods were developed for system safety and reliability as analytic approach. An undesired effect is taken as root top event of a tree of logic. Each situation that could cause that effect is added to the tree as a series of logic expressions [AD00]. In the case of a critical failure mode, all possible ways that mode could occur must be discovered [ST07].
Failure Mode and Effect Analysis: A methodology for analyzing potential reliability problems early in the development cycle where it is easier to take actions to overcome these issues [TL06]. FMEA is used to identify potential failure modes, determine their effect on the operation of the product and identify actions to migrate the failures [Ise05].

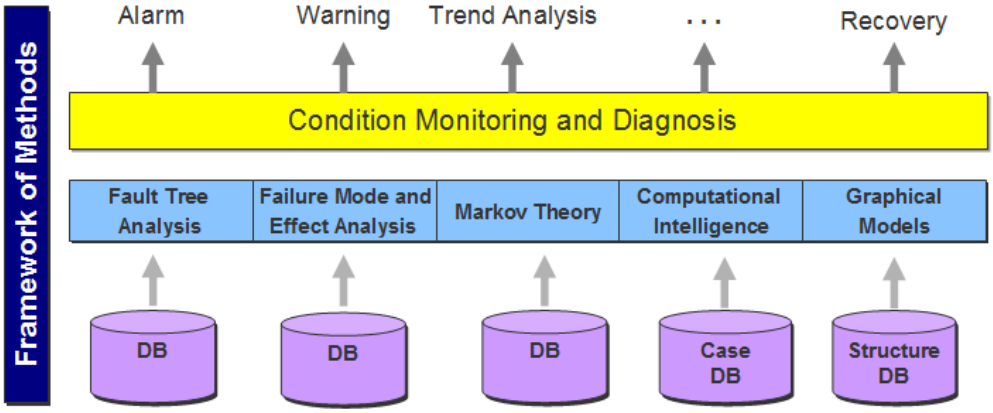


Figure 7.4.: Condition monitoring and diagnosis techniques as framework during product operation and usage

Markov Theory: A discrete time stochastic process with the Markov property. A Markov chain is a sequence of random variables X_i with the property, that given the present state, the future and past states are independent. Hidden states [LHQB06] can be determined by observable output symbols based on state-related probability distributions [JMS05]. Markovian systems ([GDX04],[KMM06]) appear so extensively in statistical mechanics as application of probability theory for modeling and analyzing dynamic systems, with which probabilities of future events are expressed.
Computational Intelligence: A rapidly growing area of fundamental and applied research in advanced information processing technology. The main components of Computational Intelligence (CI) encompass a number of technologies such as knowledge based systems, neural networks, fuzzy logic systems and genetic algorithms. CI provides methods and techniques [SWR06] for analyzing, constructing and developing intelligent systems [RN03]. The technologies have their origins in biological or behavioral phenomena related to humans using their senses (e.g. from seeing, touching). A neural network characterizes its significant learning abilities by utilizing examples from data and organizing the information into a useful form. This form composes a model that represents the relationship between the input and output variables. This behaviour can be interpreted as desired learning attribute to be applied in condition monitoring

and diagnosis environment. Neural networks are suitable in the CM context [Wan03] thanks to their ability to learn complex, non-linear functions, identify dynamic processes [XYZ97] that are tough to extract and thus model the behaviour of complex production systems [Mor05]. Fuzzy logic deals with uncertainty and ambiguity. Production monitoring tasks have often not the ability to cope with accurate mathematical models [Jan06]. In such situations there is the alternative to formalize a mapping between features and the machine status with fuzzy logic [YE06], where imprecise information imitates human reasoning processes [JH03]. The input and output variables are encoded in insecure representation [Pen04]. It is possible to deal with knowledge items on different levels of complexity through aggregation and granulation techniques. Genetic algorithms have the idea to evolve a generation of possible candidate solutions to a problem using crossover and mutation operators based on the natural selection and evolution theory.

Graphical Models: Graphical models [BK02] play an important role in representing and processing applications with uncertainty [KN04] and for characterizing conditional (in)dependencies [OC07]. Graphical models were developed as means to build models of a problem domain (e.g. production machine) of interest [Mit97]. It is necessary to decompose the available information in high-dimensional domains [BK02]. In graphical modeling such a decomposition is based on conditional dependence and independence relations between the machine attributes used to describe the domain under consideration [MKT07]. The structure of these relations is represented as a graphical network including nodes for the attributes of the problem domain and edges representing a direct dependence between two attributes. Using probabilities for giving an estimation about a problem domain (e.g. machine domain) is an intuitive approach. A knowledge expert gives an estimation about the domain knowledge on variable Y by calculating the conditional probability that Y is known under the condition evidence [HF06c], where evidence is the previously detected information about this problem domain. An advantage of constructing and using such network types for CM is the inference step on top of the complete knowledge model of the machine application domain. This knowledge model contains all necessary prerequisites (e.g. environmental knowledge, machine parameters) for a particular knowledge item, model dependencies between knowledge items and the ability to infer that prerequisite knowledge has already been acquired [GH01]. Another advantage of using Bayesian networks as CM technique is the management of uncertainty in the user's observations [HF06b] by using every degree of information on different abstraction levels about the user's knowledge. For CM applications machine parameter data can be modeled [CdC07], graphical structures learned [Nea04] and probabilistic inference engines applied. In this context Bayesian networks and influence diagrams [Jen01] can be found in medical diagnoses, logistic applications [CdC07], financial applications or expert systems ([Wit02],[CDL07]).

7.2.2. Product Data and Condition Monitoring Architecture

A product data management and condition monitoring architecture (PDMCM) is designed for gathering conventional PLM data and field data from the operation phase of the product instances which is gained by monitoring and diagnosis of mechanical equipment. PDMCM can be made up of the parts sensor array, signal processing, feature extraction and monitoring, meta data management and data analytics & decision support for the aggregated machinery data.

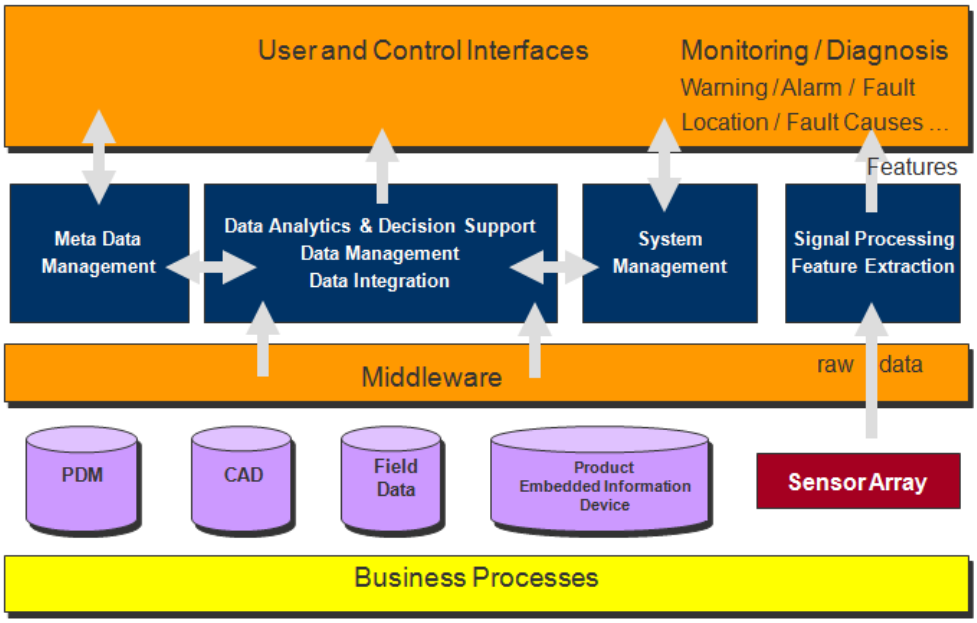


Figure 7.5.: Product data management and condition monitoring architecture [AFHN07]

Figure 7.5 shows the data and meta data flows in the overall PDMCM architecture including the condition monitoring components based on sensors, signal processing and monitoring subsystems. The main advantage of current PLM-solutions is the strong integration of the operation phase, which is neglected in existing PLM-solutions. The condition monitoring and diagnosis components allow the coupling [KSP05] between the integrated PLM meta model and integrated PLM data vault and the condition monitoring output data (e.g. production machine warning, alarm, performance evaluation) [SHRW05]. Sensor arrays allow to convert physical quantities into electrical quantities for the sake of computerized post processing [Bee04]. The selection of right production machine sensors is the key to effective condition monitoring and diagnosis because without the ability to acquire accurate information the quantitative monitoring and accurate diagnosis of equipment would be very difficult [GL06]. The signal processing and feature extraction part is necessary to receive sensed measurements and human observations and operating parameters from the controller and process them on the basis of the signal nature and requirements from the monitoring and diagnosis task [HMW+07]. Feature extraction is here funda-

mental to map incoming signals into useable features ([JHRW02],[WLRH96]). The monitoring system part should detect any forms of deviations from normal situations and diagnose any potential problem. In order to manage data about different product instances of a product type we have to distinguish between classes (which are used in the early phases of the product lifecycle) and instances of these classes (for which CM-data is collected and diagnosed during the operation phase of the product). Therefore the PLM-system has to be extended in a way that it is not only able to manage data about conventional product classes but also about instances of these classes. The condition monitoring results (gained in the operation phase) may then be incorporated in the development of the next generation of a given product using an advanced PLM-system based on the described architecture.

7.2.3. Integration Concept for Capturing and Processing Feedback Data

The traditional interpretation of PLM is the management of various products' design and manufacturing data and their versions and variants. The product lifecycle consists of the design/development, manufacturing and sales/marketing phases. In this case PLM is characterized by managing product information regarding the product type, i.e. digital models (e.g. the BMW 3 series) of the physical product, which is produced and used in later phases of the product lifecycle.
This thesis, however, aims at integrating the product use phase into PLM. The information in this phase no longer refers to the general product type, but to its precise instances, called product items. The conventional PLM concept thus needs to be expanded with respect to the management of product information on the item level. The traditional product type PLM manages the manufacturer's product information, as a large amount of the information is generated here.
A great deal of information is generated during the product use phase, outside of the companies developing and manufacturing the product, and has to be managed on the item level. Therefore gathering and utilizing this product information poses a challenge to product item PLM. The problems identified in product item PLM are as follows:

- Data holding backend systems are unavoidable due to the impracticality of saving all product information on the product itself. Every product item must be globally uniquely identifiable, to create a relationship between the actual item and its product data on the backend system [FHB06].

- Product items generate information in the product use phase that have to be passed on to the backend system. Depending on the product, its location can regularly change, making concepts for synchronizing product information in the item level necessary [FR06].

- Lack of an integration concept for product type and item PLM. Product item information from product use has to be adequately combined and

managed based on an extended meta-data model in the PLM system to be useful for product development. This is done by means of aggregation and fusion methods.

It can be seen from the above described integration of the product use phase that it is mandatory to develop an extended PLM meta-data model. This model is the basis for saving, managing and linking all product and process related data in the product lifecycle. The proposed model was developed based on the UML (Unified Modeling Language) object-oriented notation. Distinguish between product type and item data as follows:

- The classical product meta-data objects for product information management on the type level are described by the classes *Product_Type, Virtual_Part, Document* (incl. associated File) as well as *Virtual_Structure*.
- The additional classes *Product_Item, Real_Structure, History_Entry, Diagnosis_Model* and *Condition_Monitoring_Data* need to be introduced to manage feedback information from the product use phase.

In Figure 7.6 the conventional classes for managing product type knowledge are arranged on the left hand side and the new, additional classes for managing product item knowledge on the right hand side.

Product_Type and *Product_Item* inherit from the super-class *PLM_Item*, which bundles and provides fundamental attributes from every element in the PLM system. The main class for product information management on the item level, *Product_Item*, represents the real states of a physical product. The real product is associated via the serial number. An instance of the *Product_Type* class can be associated with several instances of the *Product_Item* class, while a concrete product item must always be assigned to a certain product type. *History_Entrys* offer the possibility of linking special maintenance incidents to the corresponding product item. The *Condition_Monitoring_Data* class captures concrete sensor data in the associated product item. Diagnosis models can furthermore be associated to product items that, for example, can be used to efficiently locate errors or to forecast upcoming maintenance work. These models can either be constructed manually or automatically, based on suitable learning algorithms due to CM data obtained in the product use phase [Nea04], [LC06], [CRAMBMNF06]. They are structured recursively, making it possible to assemble complex models from individual, hierarchically arranged, partial models (such as certain generic modules, e.g. a specific type of bearing). Various diagnosis models, which are assigned to different product items, but the same product type / module, can be united to increase model accuracy, thanks to aggregation methods [CSK04], [RLNR06]. These aggregation models (compare *Aggregated_Diagnosis_Model* class) are particularly interesting for product development and can provide a basis for deriving future product improvements. New product services optimizing the knowledge flow between customers, manufacturers and service providers, can be generated from the presented integration concept. This concept can be used as a basis for a holistic product lifecycle concept. While in the early phases of the product lifecycle (product development,

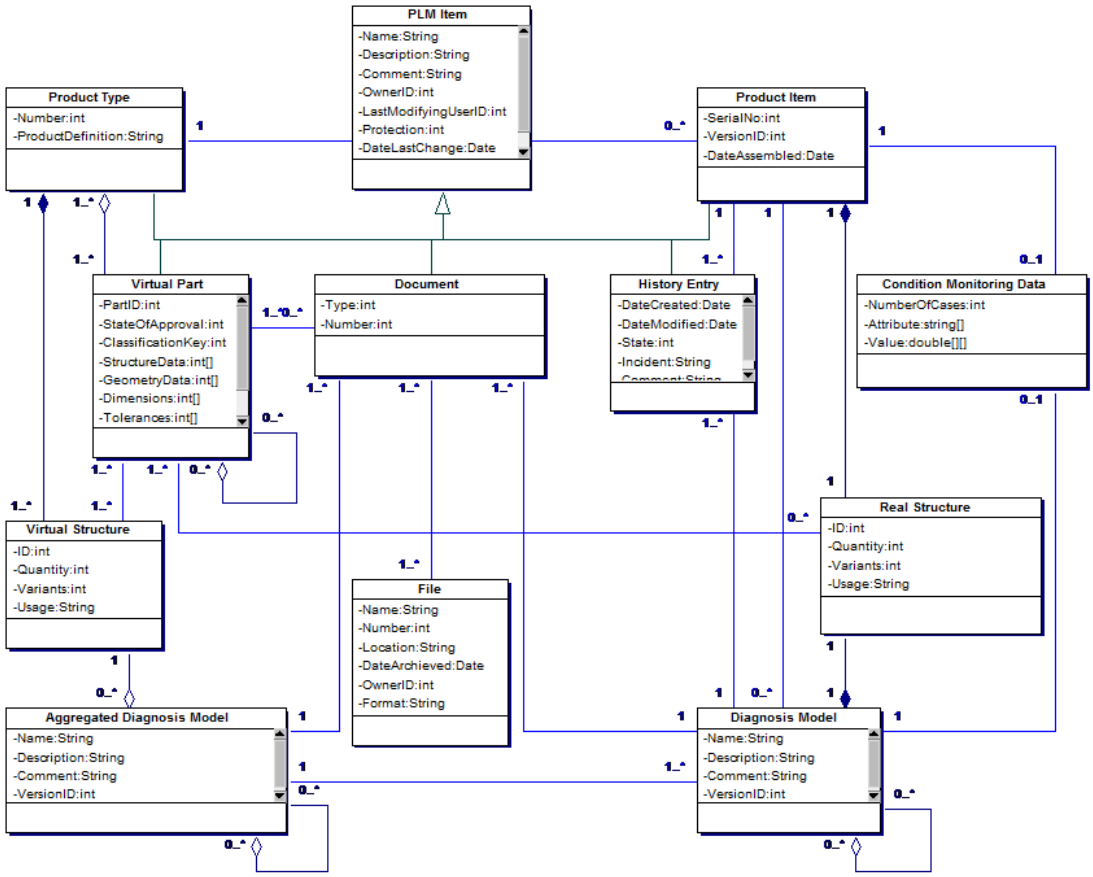


Figure 7.6.: Basic extended meta-data model UML class diagram for the integrated management of product type and item data [AFHN08a]

manufacturing and distribution) PLM is distinguished by managing product information related to the product type, information coming from the product use phase is related to concrete instances of products developed in earlier phases [TYJL06]. The PLM concept was extended so that the meta-data model could serve as a basis for managing product type and item data. The three main participants manufacturer, user and service provider access the data with different views and rights (see Figure 7.7).

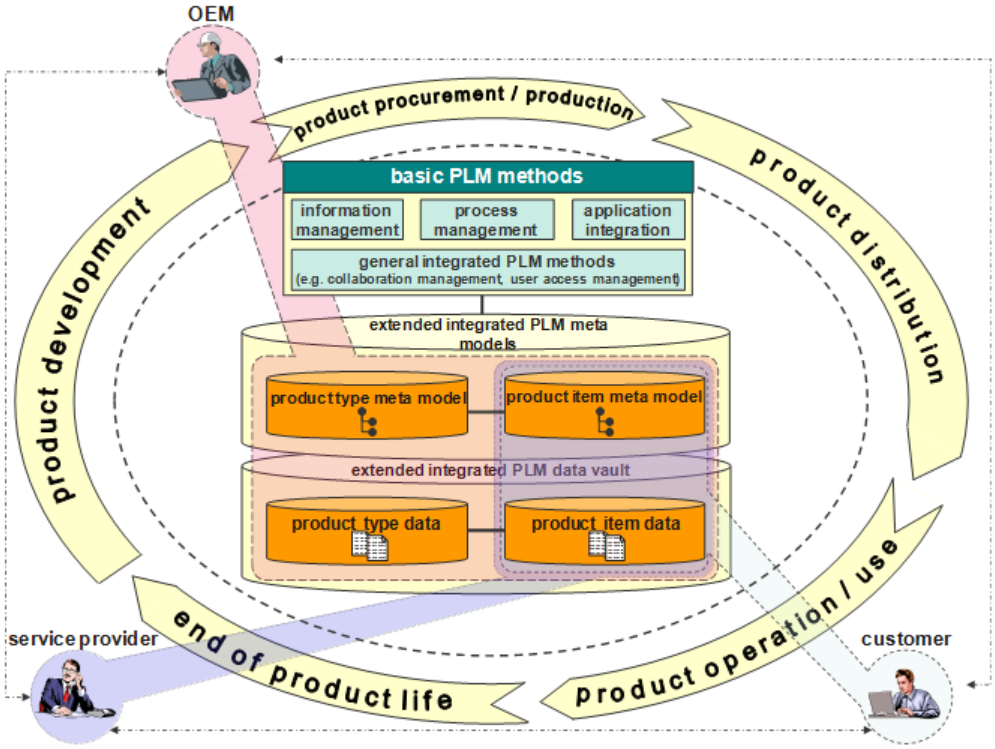


Figure 7.7.: Integration concept as a basis for new product services

New fields of business in service or in collaborating [VH05] with external service providers open up to the manufacturer, who until now mainly generated and managed product type knowledge [WTL07]. Furthermore the aggregated product item knowledge may be incorporated into the development of following generations in the scope of a feedback cycle to improve customer loyalty and customer relationship management services. Future PLM approaches will therefore integrate development and service partners as well as customers within different stages of the product lifecycle [Mye05]. Customers can provide preferences, wishes and requirements to the producer with prospective feedback. During the product usage and operation phase the customer can also provide feedback on his experiences, his satisfaction with the product use and more improvement proposals, which can be used by the producer for the development of the next product generation as retrospective feedback. PLM models and methods were extended in order to integrate this customer feedback into the product development processes (e.g. change management processes), into the

product structure and into the classical PLM configuration management. Specialized PLM models, methods and tools will better support the management of other downstream product life phases like operation monitoring, optimization and maintenance of the product during its usage [Qiu06]. CM acts as a major component of predictive maintenance and as extension of the PLM approach supports the development of new product generations. Customer feedback information and condition monitoring method framework results carry over information and experiences from the product operation phase of generation n to the product development phase of the next product generation $n+1$.

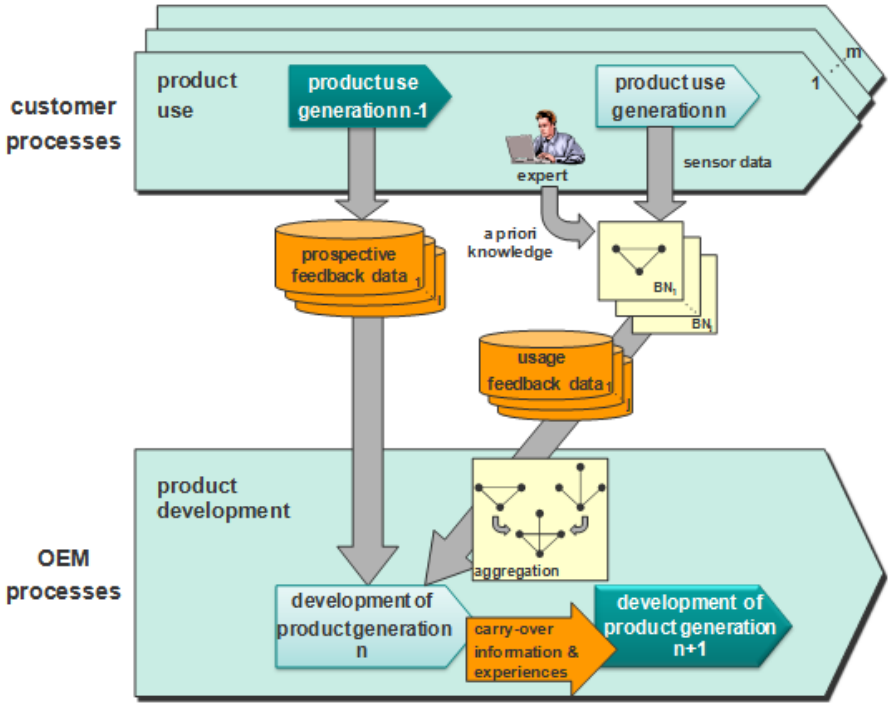


Figure 7.8.: Feedback of information from the product use phase to the development phase of the next product generation

It is imperative to represent and process data, compressed through aggregation and fusion methods [CSK04], in numerous granularity levels when integrating distributed data models. The integration is based on the diverse views and access rights of the users. Advanced aggregation methods work with all available source information at aggregation time and allow extensions to take the user's prior knowledge into account [RLNR06]. The CM results, collected in the product use phase from various customers are aggregated at original equipment manufacturer (OEM) side (see Figure 7.8). In this context more complete and exact knowledge representation models can be implemented for developing following product generations as well as for use in the manufacturer's or external service providers' service departments.

7.2.4. PLM Feedback Cycle as Enhancement of the Support Knowledge Engineering Process

Supporting business processes with knowledge management technologies is one of the key factors in today's industry. The former technologically related organizations now have to turn to the business strategy of the company, which they must support with their offered IT services. IT processes have to become more transparent and better manageable. It is fundamental to visualize service propositions to satisfy customer needs. Different business processes and the potential of supporting them need knowledge management measures. The illustration of applicability and possibility of IT service management and appropriate support services in business processes especially in the field of PLM require a novel feedback cycle for customers, service providers and OEM to assist such service support processes.

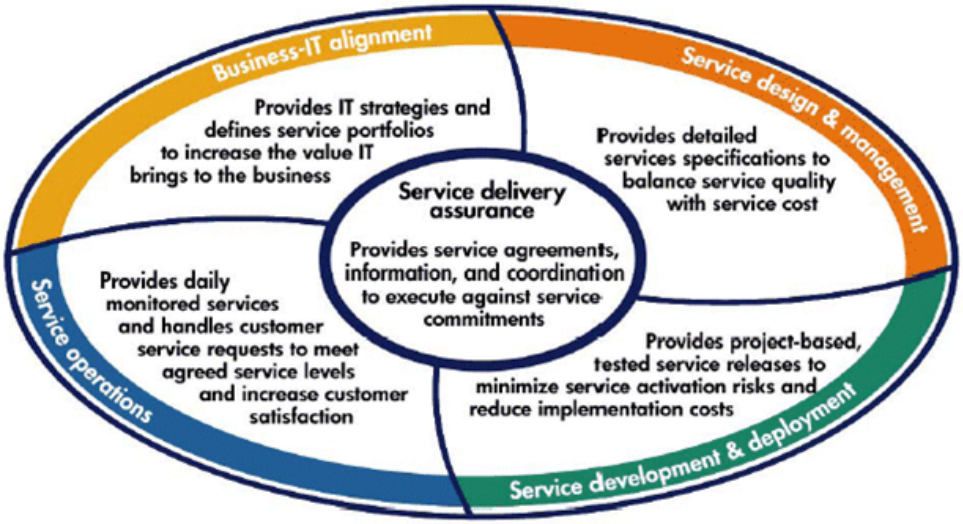


Figure 7.9.: HP ITSM reference model structure including five different groups

New digital technology is improving business efficiency by radically increasing the quality and quantity of information available to PLM knowledge workers. Networks can be built if some sort of catalogue of knowledge (knowledge bank) exists that can be used to connect people [RES+00]. Best practices were found in the four main areas product support, product sales, workflow and project management. This means in a condensed form a structured knowledge [All98] and information creation process project for customer product related support. The effective knowledge creation, maintenance and sharing becomes more and more a crucial factor for support product organizations and especially for the support business [Mai04]. IT Service Management (ITSM) helps delivering and supporting IT services that are appropriate to the business requirements of the product organization, and it achieves this by leveraging IT Infrastructure Library (ITIL)-based best practices that promote business effectiveness and efficiency. The ultimate goal of ITSM is to provide quality services to customers.

The IT company Hewlett Packard has organized IT processes into five different groups that focus on different aspects of the service lifecycle. This approach allows product users to follow a complete service lifecycle [Grü05]. HP ITSM enables to consistently deliver product services in a way that balances performance, quality and cost.
Starting at the upper left and proceeding clockwise around the model (see Figure 7.9), users can follow the progress of an IT service from initial conception to delivery, eventual obsolescence, and updating or replacement by a new service. A support knowledge engineering process within product service operations describes the call flow inside organizational structures and the knowledge creation and retrieval action. Most of the standard product requests and feedback issues were solved by using support knowledge tools such as knowledge databases, retrieval systems or knowledge trees implemented and realized in a knowledge desk framework. A product domain architect (DAR) develops the product knowledge tree structure and creates the knowledge content for a domain. Figure 7.10 illustrates two domains of a mobile phone manufacturer who offers diagnosis trees for the problem domains *SIM card* and *mobile phone battery*.

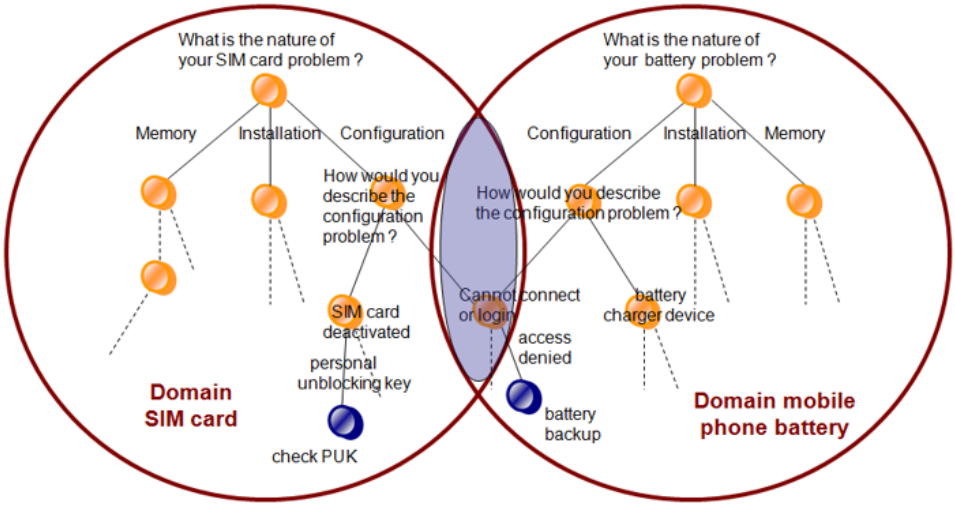


Figure 7.10.: Knowledge desk decision tree structure including two domains of the product mobile phone

A knowledge desk (KD) is technically a system portal solution that can create inside product organizations a case-based reasoning and advisory system for guided retrieval, maintenance and creation of knowledge [HF07a]. KDs are aimed to enable and encourage structured, seamless integration of process and solution information from a variety of data sources and legacy systems. It also offers a fast linking and learning decision tree system integration to all users in the online process of content and structure maintenance. KD users are able to focus on handling the issues and feedback, while the knowledge tool takes care of the complexity of the systems, networks and processes for them. It is more than a knowledge tool. It is also a strategy for complete integration of the processes

of knowledge retrieval and maintenance, and workflow management. It is an instrument for a fast learning product organization as fast linking of knowledge about people, organizations, products, problems, customers and processes is a key to success. KD contains a powerful feedback process which transforms it into learning and growing organism. Figure 7.11 illustrates a screenshot of the knowledge desk. As can be seen top left, there is the tree structure with the last questions that where asked for visualized. The first question, the entry point is connected with the experience level of the questioner. There the user has basically three different possibilities, high, medium and low. The user can also visualize the tree structure using hyperbolic trees.

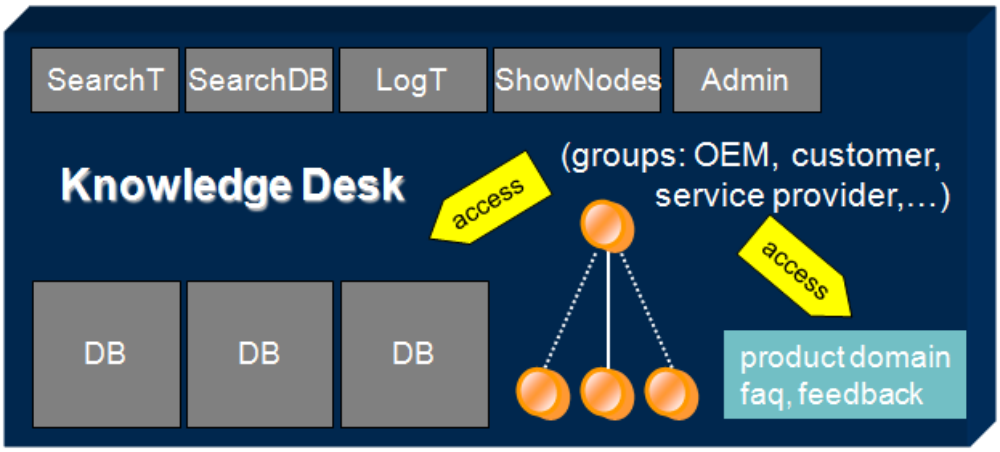


Figure 7.11.: Knowledge desk structure including the tree structure, user navigation panel, hyperbolic tree and node search

A direct jump and link to a specific KD node is available via the syntax *product-domain-name/question.x*. If the customer chooses a solution node he will get detailed solution information concerning the initial request and also several helpful links to other web pages or other tools. Figure 7.12 illustrates a vision for the support knowledge chain including product support and feedback processes to increase the benefit of involved groups based on their observations.

Next follows a description of the main tasks to implement the product feedback process. A feedback process is necessary to make a knowledge desk a powerful tool [HF06d]. It also helps to grow together within a knowledge support community as support network. Furthermore, the feedback process has to be easy and simple to handle because otherwise nobody will use it again [JHS01]. A feedback process and its use have to be communicated to employees to improve acceptance and to be daily business. The access to feedback has to be more problem oriented [KHS03], otherwise support users experience feedback as useless. A necessity is the common understanding of the various definitions, a clear communication flow, a fast processing of the feedback and fast reaction dependent on the request, obvious and clear structure of the feedback to support the easy handling, a well known owner of the feedback for further contact, definition of priorities and a mechanism to easily analyze feedback. It should also be possible to give general product-related feedback which is not related to

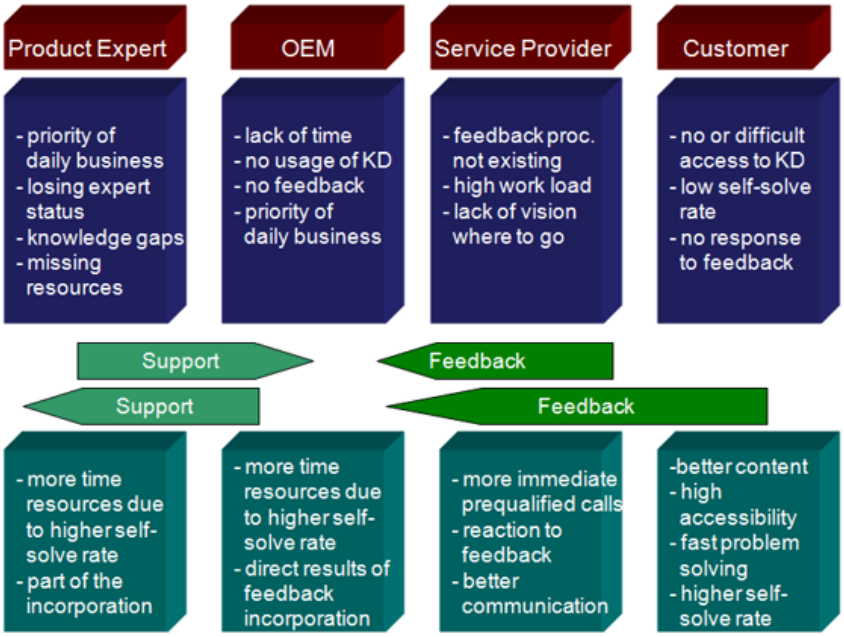


Figure 7.12.: Vision for the knowledge chain with observations for the product support

the KD. In this case, it makes sense to develop a feedback process within an organization collaboration involving the groups service provider, product expert, OEM and customer as illustrated in Figure 7.13 in detail.

After the user submits a new feedback in the support queue (status assigned), the product responsible takes over the case (status open) and follow up. Answered requests have a specific waiting time (based on the priority) before closing. The user has the possibility to check the feedback status online at any time. The content feedback influences the technical infrastructure to grow to an underlying powerful support tool. After the feedback process was implemented it had to be communicated [WC03] to the different groups and divisions to get their commitment. The product domain responsible verifies the feedback, updates content and makes status updates to aid feedback incorporation. Incident Management's goal is to minimize the adverse impact of technology problems on business operations, ensuring that the highest levels of service quality and availability are maintained [HFPS07]. A way that supports this process is by bringing together management data from across the infrastructure and giving IT [Thi05] a single place to find and fix problems [ZL05]. Next consider the quality and performance management of the product feedback process to assist the support product management in generating business support measurements for support innovation, risk and efficiency [HF07b]. Support innovation touch e.g. growth in support market, percentage of revenue from new products and processes, profits resulting from new business operations or percentage of research and development in the support business field. Risk measures the number of

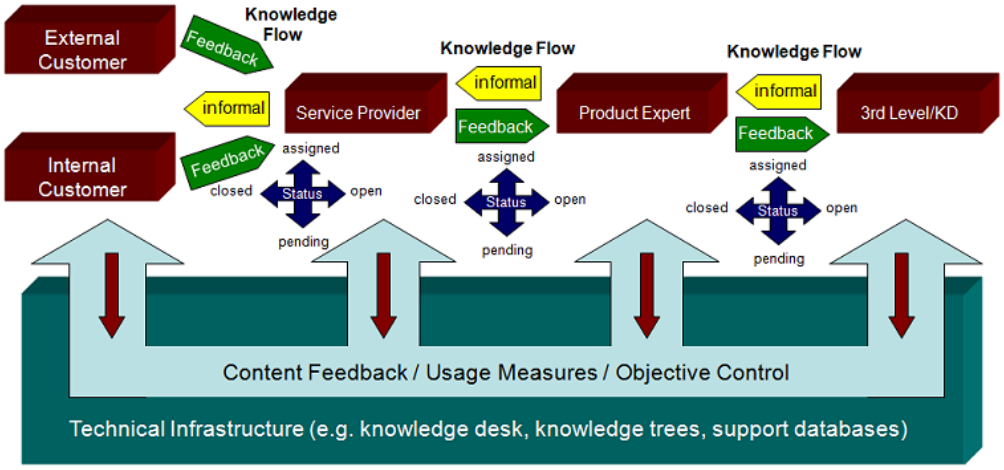


Figure 7.13.: Product feedback flow with feedback states and the underlying technical infrastructure framework

customer complaints, percentage of repeat customers as of total, the employee turnover ratio or return on research and development spending [Hol04b]. Efficiency is an indicator for the profit per customer, value added per employee, usage of support tools and knowledge bases, time to market of new products and services or the revenues per customer. The quality and performance measures for the complete feedback cycle with embedded product support groups and functionalities like service provider leads to a Quality & Performance Management (QPM) metric for knowledge creation and maintaining, management and support knowledge tools.

Table 7.1.: QPMs for the product feedback

QPM
number of new cookbooks
number of unique knowledge trees
knowledge tree traffic
number of customer feedback
time to feedback response
customer satisfaction
product handling (e.g. menu navigation)
product design (e.g. size, weight)
product function (e.g. camera)

Table 7.1 specifies customer oriented product feedback helpful for the mapping onto new products and the integration into next product generations. Customer feedback contains information about customer requirements, product preferences, technical requirements and the product structure from a prospective and retrospective point of view [Sch07].

7.3. Adaptive Condition Monitoring for a Rotation Spindle

The prototypical implementation and validation of the introduced concept is realized as integrated PLM suite. In this regard Siemens' PLM system *Teamcenter Engineering* serves as a test bed for the implementation. *Teamcenter Engineering* provides a comprehensive, collaborative work environment optimized for product development involving large project teams. In situations where project team members may be widely distributed and a wide range of tasks must be efficiently synchronized, this solution offers the tools and coordinated workspace to support the required operations.
The Teamcenter Engineering solution may be seen as a core platform for continuing PLM improvements, enhancements and new developments. Its strengths are in managing CAD models and technical documents, supporting design releases and change processes as well as the close integration with CAD and ERP systems. However, the clear focus of the solution lies on product development activities, whilst the integration of later phases in the product lifecycle, especially the product use phase are disregarded. This is the point where this extension applies. By implementing the introduced concept, Teamcenter Engineering's functionality has been enhanced by the following aspects:

- **Creating / editing product items:** In order to manage product use information, it is necessary to differentiate between product types and product items. Information management at the product type level (release management for CAD-models, change management for technical documents etc.) is already supported by Teamcenter Engineering. However, the information from the product use phase (e.g. condition monitoring data, diagnosis models and maintenance histories) do not refer to a general product type, but to individual product items. By implementing a product item class with adequate attributes (e.g. serial number, date assembled, etc.) and appropriately designed forms for the data acquisition, we are able to create and edit product items and attach development-relevant product use information like CM data, diagnosis models and maintenance histories to the appropriate product item. Figure 7.14 shows a screenshot of the enhanced Teamcenter Engineering solution applying a rotation spindle as use case. Product type data like CAD models of the rotation spindle is tagged with red icons, while the newly-integrated product item data is tagged with yellow icons.

- **Linking maintenance events:** In order to track the maintenance events in the product use phase of a given machine, it was necessary to enhance Teamcenter Engineering's functionality with regard to the management of maintenance histories and their association to product items. Therefore appropriate classes and attributes were defined and linked with newly designed forms for data acquisition.

- **Linking condition monitoring data:** Condition monitoring data collected in the product use phase may be linked to an individual product

item distinguished by its unique serial number. In Figure 7.14 an associated ARFF file is shown. An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of the University of Waikato [WF05]. In our case they store large amounts of condition monitoring data previously recorded by observing an industrial machine through sensors.

- **Embedding individual and aggregated diagnosis models:** The extraction of useful information from CM data stored in databases or huge case datasets like ARFF files is usually very difficult for humans. In this regard data mining methods may be incorporated to reveal the knowledge hidden in large case datasets and derive relevant interdependences between sensor data, environmental parameters and product failures. We have developed machine learning algorithms [HF06a], which transform the data sets into Bayesian networks. Another paper highlights the topic of learning Bayesian networks in detail [HFAN08a]. Coming back to Teamcenter Engineering, we extended the functionality in terms of an integrated creation and modification of Bayesian networks, their visualization and the possibility of executing what-if-analyses on the basis of an inference engine applying the currently most efficient method for exact belief updating in Bayesian networks, the junction-tree method [JLO90]. However, the multiple individual diagnosis models on the product item level [HFAN08b] in order to set up a representative knowledge model on the product type level remains to be aggregated.

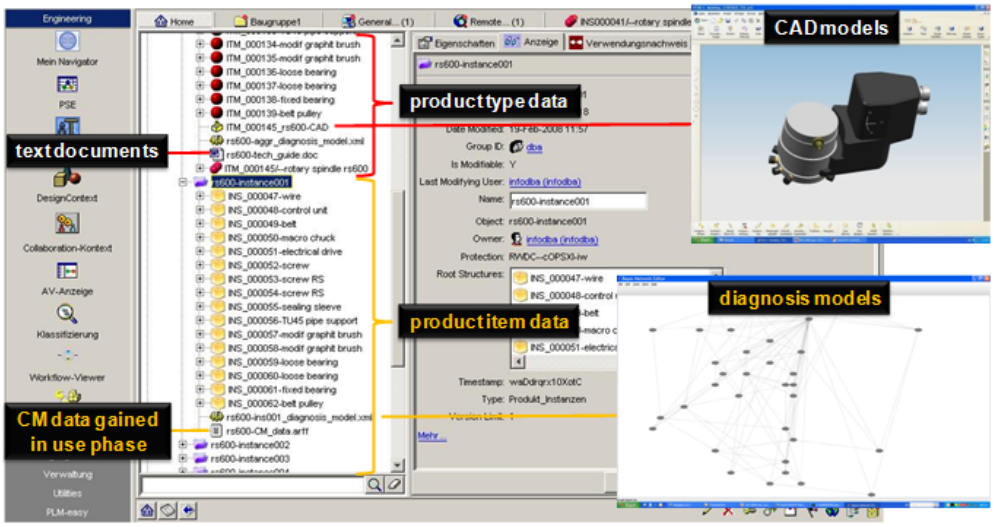


Figure 7.14.: Enhanced version of Teamcenter Engineering showing managed product type and product item data of a rotation spindle [AFHN08b]

The Teamcenter Engineering functionality was enhanced in regard to the creation and modification of product items, their association to the appropriate product type, the linking of maintenance events, CM data and diagnosis models, including support for visualization and what-if-analyses. The strong integration of the product use phase can be seen as an important step towards a new generation of PLM systems, which integrate data from all lifecycle phases in a holistic approach. It furthermore paves the way for new business models for manufacturers as well as service providers. The product use phase examination allows to bring product item, process, manufacturing, and service information in a single source of product knowledge. The enhanced version enables to employ this knowledge in workflow driven processes to synchronize activities with other team participants. Security settings enable to work with suppliers, partners, and trusted customers in a collaborative framework on the product item level to facilitate concept studies, program, design, and change reviews based on product use experiences. It is also possible to plan and deliver maintenance and repair tasks on the product item level. Exchanged experiences out of the product use phase facilitate maintenance planning and enable service organizations to define and plan activities for assets ranging in complexity from components to entire engines. These advanced and detailed planning opportunities enable to track and manage part and equipment inventories used to repair, maintain, or overhaul assets. The product item usage history facilitates in-service and service event management including access to configuration knowledge that comprehensively describes each service state and capture results from service activities performed anywhere in the service value chain.

7.3.1. Classification of the Spark Erosion Process

After outlining and evaluating the product type and product item data system integration using the example of Teamcenter, now an example for a monitoring and its integration into the PLM is presented. As a monitoring system a wire-electro discharge machine is used, thus being represented in a Bayesian network. Beforehand a strict description of the electrical discharge machining is given, on which the presented machine's machining principle is actually based. Following this the general construction of the wire-electro discharge machine is outlined so that all relevant components are clear to see. On the basis of this information the condition monitoring of the wire-electro discharge machine is described afterwards and represented with a Bayesian network. This Bayesian network shall be populated into the PLM software solution Teamcenter. The Bayesian network is maintained as product-instance information for the product-model of the wire-electro discharge machine.
In engineering a vast number of manufacturing methods is existing, which are applied by several machine tools to handle a work piece. Before describing spark erosion in detail, therefore an overview concerning the adapted procedures and the classification of spark erosion is helpful. In general parts are formed following the DIN 8580 by archetyping, transforming, disconnecting, assembling, coating or changing certain material properties (compare Figure 7.15).
The procedure of erosion belongs to the disconnecting procedures, since during

the process of manufacturing a work piece its cohesion gets changed [WB05]. Within the disconnecting procedures a further breakdown is carried out so that certain type of modification of cohesion is reflected. Thereby the spark erosion is assigned to the erosive procedures, which are circumstantiated in DIN 8590.

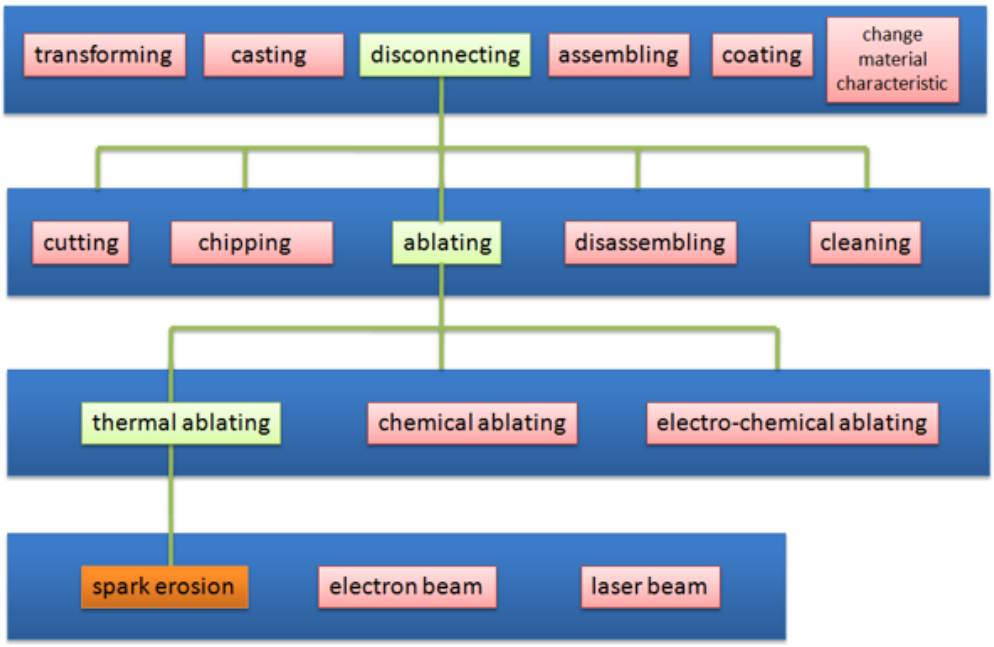


Figure 7.15.: Classification of manufacturing procedures referring to DIN 8580

Erosive procedures are characterized by the way, that they do not perform any mechanical action during the adaptation of the work piece, thus meaning that they are contact-free removal. This enables a handling completely independent from any attributes of the working piece, so for example without consideration of a work piece's hardness or benignity. This procedure is especially fitting for constructing micro-structured work pieces [WB07] with the application of micro spark erosion [UDP01]. Besides further erosive procedures the spark erosion, applied in the wire-electro discharge machine, nevertheless has the limitation that merely metallic work pieces can be adapted.

7.3.2. Principle of the Spark Erosion

The principle of spark erosion is based, as the name suggests, on sparks and their thermal consequences on a work piece. Particles are separated by the sparks while being in a solid, liquid or gaseous condition and afterwards removed by mechanical and/or electro-magnetic power. This process is also called Electrical Discharge Machining (EDM) [WB05]. Thereby, the sparks emerge by electrical discharges between certain tools and the work piece and furthermore create a high temperature at the point of working. Besides the material removal on a work piece there is additionally a noticeable metal removal on the working tool. For cooling purposes and for the removal of segregated material there is

a dielectric fluid, thus consisting of carbon compound or deionised water. It is characterized by an especially poor conductivity and so isolates the electric wire (tool) and the work piece [Feu83].

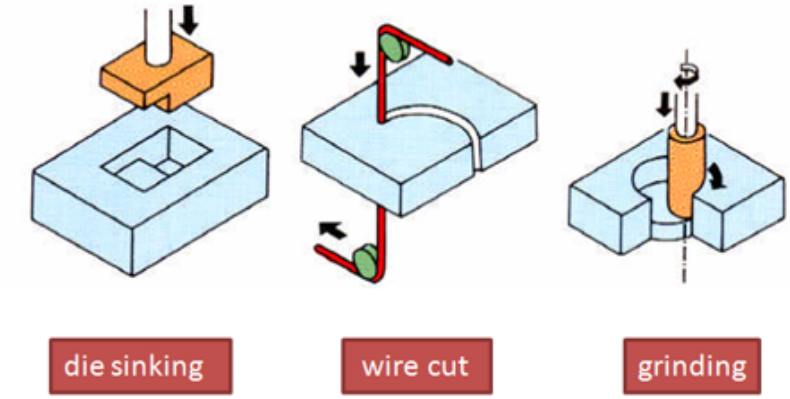


Figure 7.16.: Spark erosive procedures

In addition to that, the dielectric has the task to provide a channel for discharging. This merely happens by a temporal ionisation. Figure 7.16 presents several spark erosive procedures, also comprising the Wire Electrical Discharge Machining (WEDM). For instance, the micro die sinking EDM process enables to machine complex cavities and inner structures with nearly sharp corners and very tiny structure elements [BOM06]. In modern micro mechanical manufacturing technologies, we cannot pass on the die sinking process. The advantages of the die sinking process with pre-shaped electrodes are mainly used to machine micro injection moulds. Complex 3D cavities can easily be realized with high precision in e.g. hardened steel work pieces.

7.3.3. Structure of a Wire-Electro Discharge Machine

In the following, the general structure of a wire-electro discharge machine is presented. In these machines, an implementing electrode in form of a wire is used, thus contact-free consigning its image on a work piece. Since the wire itself thereby gets certain material removal, it is continuously replenished by an engine, to provide a constant material removal on the work piece. A generator thereby supplies the working tool and the work piece with required voltage, so that a discharge and the associated appearance of erosive sparks is possible at all at the working point. The relative movement of wire guide and work piece is taken over by a separate control. The dielectric is thereby continually fed to fulfill the tasks mentioned above.
Figure 7.17 shows the structure of a wire-electro discharge machine from Precision Design Lab[1]. The monitored wire-electro discharge machine basically is similar to the one presented above. Though, the work piece is not fixed on the machine-unit, as presented in Figure 7.17, but rather fixed in a rotation spindle (see Figure 7.18). The resulting piece is therefore always axially symmetric.

[1]Department of Mechanical Engineering, University of Utah

The researched wire-electro discharge machine contains a rotation spindle from the provider System 3R [SR08]. EDM machines, working with rotation spindles, are commonly called Wire Electrical Discharge Grinding (WEDG). These machines are mostly used in the manufacturing of micro-structured work pieces. The erosion wire therefore has a cross section dimension of about 30μm. The so called constancy in contour explains fluctuations in the working process and is measured with 1μm.

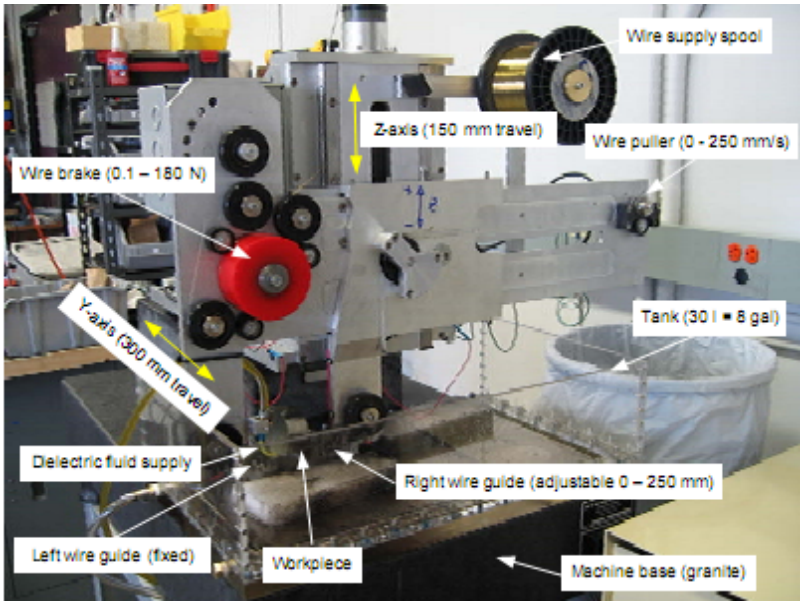


Figure 7.17.: Precision machine design example wire electrical discharge machine

With this, this procedure allows highest accuracy and surface quality for the finally produced work pieces. The structure of the researched machine is drafted in Figure 7.18. The rotation spindle is assembled on the machine base. This spindle moves into gear by a drive motor with a driving belt. The electrode (work piece) is fixed to the spindle with some kind of chuck and therefore rotates with exactly the same speed. This speed is therefore specified by the drive motor, which, by its own, is regulated by a control instance. During this, the erosion wire continually runs by a wire guide. This represents the antipode for the electrode. To enable the accruement of erosion sparks, the working point continually is supplied with dielectrics. The electrode on the other side gets its power by graphite brushes. With this, the current conduction inside the ball bearing of the rotation spindle gets decreased. The drive motor's electric circuit is strictly disconnected from the electric circuit used for the erosion by an insulating plate. Rotation spindle and drive motor are continually provided with compressed air, so that any intrusion of liquids or removed material is avoided.

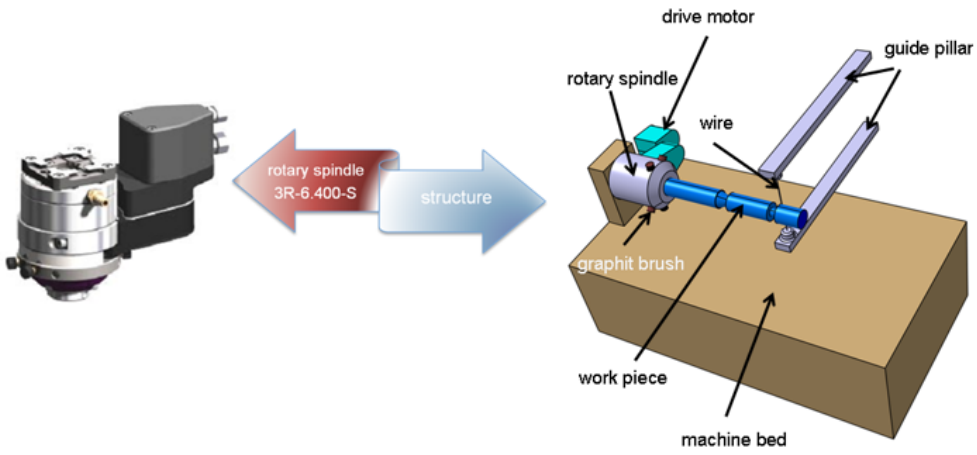


Figure 7.18.: Rotary spindle 3R and structure

7.3.4. Condition Monitoring of the Wire-Electro Discharge Machine

To provide the researched wire-electro discharge machine with a condition monitoring and therefore enable knowledge representation in the PLM, the phases data acquisition, data preparation, diagnostics and decision-making have to be passed through.

Data Acquisition:
The actual machine state is determined by a set of meaningful parameters. For this, different reading recorders are available, each one monitoring single components (compare Figure 7.19). The following sensors are used in Condition Monitoring in a wire-electro discharge machine:

- Vibration sensors are used in the controls of the trolley's rotation inside the rotation spindle. They are able to recognize the erosion of the trolley at an early state, since a worn out trolley is causing higher vibrations compared with an undamaged one. This parameter is measured in *hertz*.
- Deduce the state of the drive belt from two rotational speed sensors, which compare the velocities of spindle and motor. If the velocities are different, the impairment of the belt is the cause.
- A proximity switch allows the determination of the remaining coating of the graphite brushes. If all brushes are used up, they are no longer able to transfuse erosion electricity. This electricity then would unintentionally flow across the trolley and there cause several unintended effects. The remaining coating is measured in *millimetres*.
- The pressure inside the rotation spindle and inside the drive motor is detected by a pressure sensor. If the pressure decreases, there is the danger that the dielectric or even threadbare material might access machine components and by this may cause several failures. The pressure is measured

in *bar*.

- An insulation monitoring of the drive motor can be achieved by measuring its insulating resistance. In the case of an insulating failure on the insulating plate between drive motor and spindle this can be detected by decreasing insulating resistance values. The unit of measurement is Ω/V.

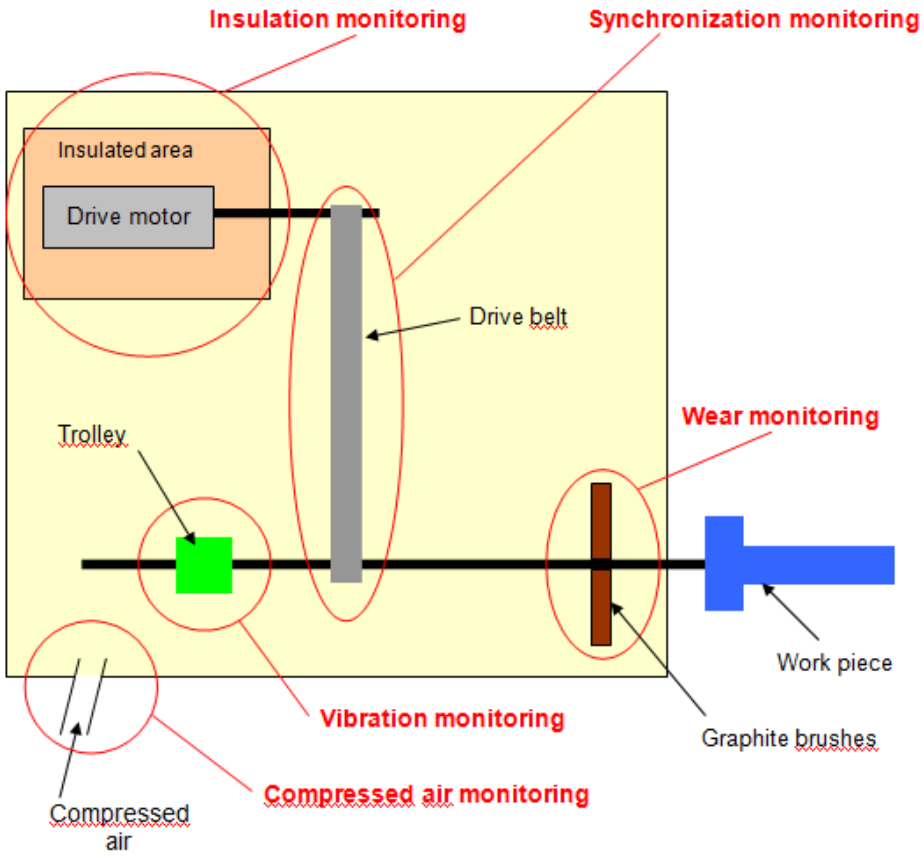


Figure 7.19.: Sensors in a wire electrical discharge machine

The shown machine parameters can be measured continually as well as discretionally. This difference, however, is not important for the following process of knowledge representation in a Bayesian network. There, every random variable obtains several states, thus comprising all gathered data inside a certain sector in the actual state of time. A discretional research has ultimately the disadvantage that no short-time fluctuations can be measured and the measurement might be done in a moment, when everything seems to be all right functionally, though a failure is still occurring.

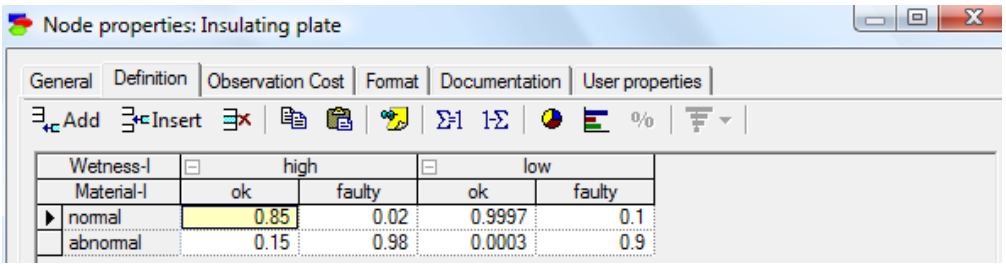
Besides the directly measured data in the machine, collected by several sensors, also some external data have to be considered. So there are, for example, environmental factors like temperature, wetness and quality of air as well as time delays, engine speed or material condition of certain machine components.

Data Preparation:
All gathered data now has to be brought into a certain structure, so that it is available for further diagnostics. This happens by a strict allocation of measured data to certain domains. So measured values like temperature can be divided into the sections “low“, “ok“ and “high“. For every machine component two differently noticed situations “normal“ and “faulty“ have to be distinguished. In addition to that several components’ breakdown can be detected. Table 7.2 therefore presents the monitored parameters and their possible states.

Table 7.2.: Monitored WEDG parameters and possible states

Parameter	State
Wetness on the Insulating plate	high,low
Material attributes of the insulating plate	ok,faulty
Function of the insulating plate	normal,faulty
Short circuit on the insulating plate	yes,no
Temperature of compressed air	low,ok,high
Air quality of compressed air	good,bad
Function of compressed air	normal,abnormal
Speed	to 4000 rpm,from 4000 rpm
Delay of the drive motor	to 1000 h,from 1000 h
Function of the drive motor	normal,abnormal
Damage on the drive motor	yes,no
Idleness of the drive motor	yes,no
Runtime of graphite brushes	to 1000 h,from 1000 h
Function of graphite brushes	normal,abnormal
Damage on graphite brushes	yes,no
Runtime of trolleys	to 1000 h,from 1000 h
Function of the trolley	normal,abnormal
Break on the trolley	yes,no
Function of spindle	normal,abnormal
Idleness of spindle	yes,no
Runtime of the drive belt	to 1000 h,from 1000 h
Function of the drive belt	normal,abnormal
Draft of the drive belt	yes,no
Material attributes of the erosion wire	ok,faulty
Function of the erosion wire	normal,abnormal
Draft of the erosion wire	yes,no
Quality of dielectric	good,bad
Function of the electrode	normal,abnormal
Failure in working process	yes,no

Diagnosis:
In the phase of diagnosis a Bayesian network is applied for knowledge representation (compare Figure 7.21). To learn the Bayesian network structure, the new conceptualized and aforementioned LAGD hill climbing algorithm has been chosen, to visualize this learned Bayesian network representation the GeNIe (Graphical Network Interface) software has been used. The figure referring to the Root Cause Analysis (RCA) is applied using coloured identification of the nodes. Green nodes represent causes and thereby the influence coefficients inside the process. Turquoise nodes show the monitored state of the process and orange-coloured nodes the possible resulting failures in components. The feasibility of the single nodes is given as a priori probability and is put down in the respective conditional probability table of the node. Every node has a different number of entries referring to its parental node in its CPT. The node *Insulating plate* for example has the two parent nodes *Wetness_I* and *Material_I*. With this the connected CPT has eight entries (compare Figure 7.20). To calculate the overall probability that node I of the *Insulating plate* has the



Wetness-I	high		low	
Material-I	ok	faulty	ok	faulty
normal	0.85	0.02	0.9997	0.1
abnormal	0.15	0.98	0.0003	0.9

Figure 7.20.: CPT of the WEDG note insulating plate

state n (normal) the CPT of the *Insulating plate* and also the two parent nodes are required (*Wetness_I* high/low 0.02/0.98, *Material_I* ok/faulty 0.997,0.003). With this, now $P_I(n)$ can be calculated:

$$\begin{aligned} P_I(n) &= P_I(Wetness_I\ high, Material_I\ ok) \cdot P_F\ (high) \cdot P_M\ (ok) \\ &+ P_I(Wetness_I\ high, Material_I\ faulty) \cdot P_F\ (high) \cdot P_M\ (faulty) \\ &+ P_I(Wetness_I\ low, Material_I\ ok) \cdot P_F\ (low) \cdot P_M\ (ok) \\ &+ P_I(Wetness_I\ low, Material_I\ faulty) \cdot P_F\ (low) \cdot P_M\ (faulty) \\ &= 0.85 \cdot 0.02 \cdot 0.997 + 0.02 \cdot 0.02 \cdot 0.003 \\ &\quad +0.9997 \cdot 0.98 \cdot 0.997 + 0.1 \cdot 0.98 \cdot 0.003 \approx 0.994. \end{aligned}$$

The probability that the insulating plate is operational therefore is 99,4%. In the same way the probabilities for all other nodes are determined. With this the probability for abnormal processing for all components can be calculated against the parental nodes. Therefore the analysis of weak components inside the system gets enabled, thus being possibly the cause for several failures in processing.

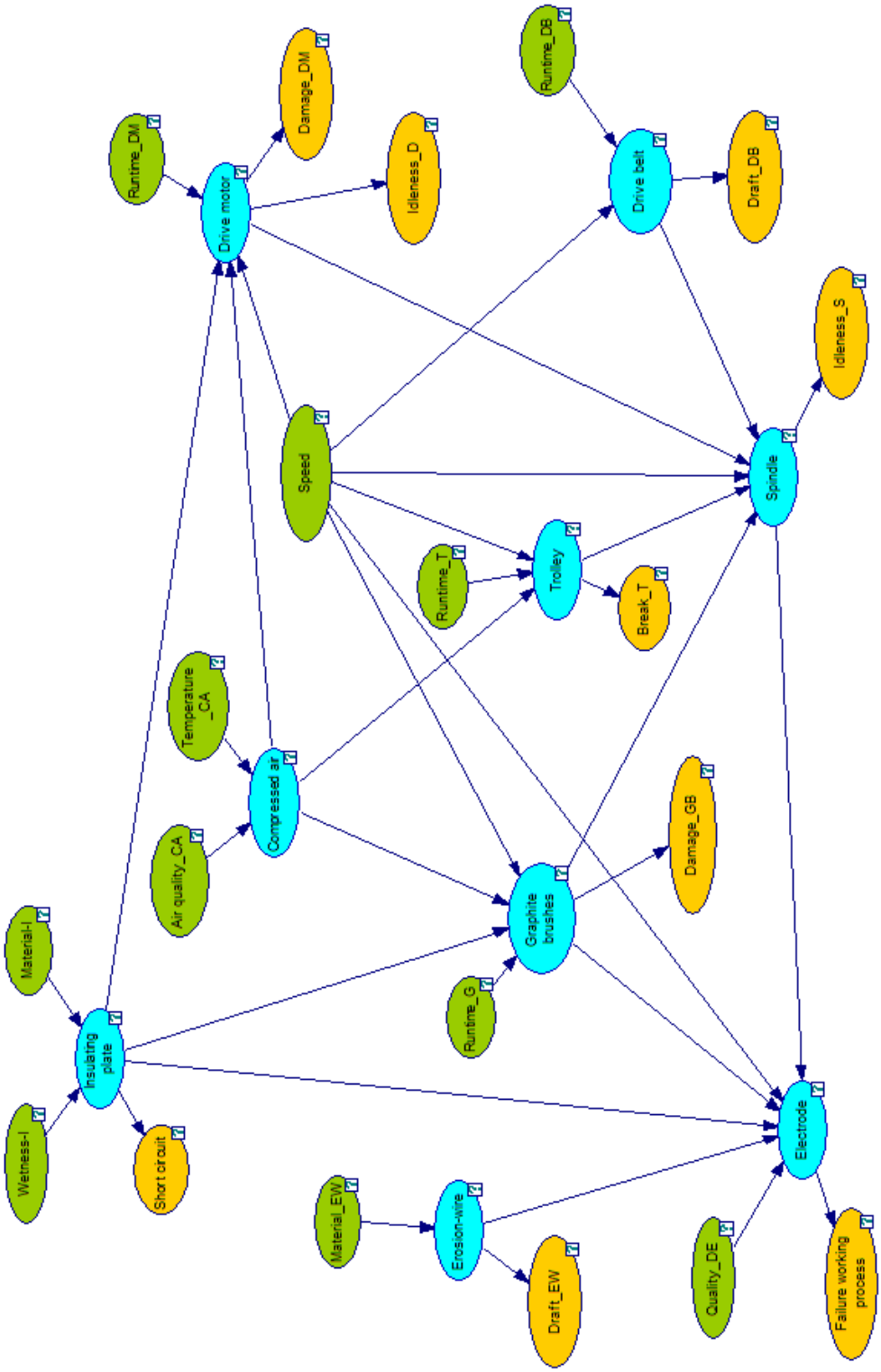


Figure 7.21.: Bayesian network of the wire-electro discharge machine

Decision Making:
Compared with conventional condition monitoring, right here the determination of rest-delays of single machine components, that can be used for the final planning for maintenance, is not of interest, but rather there is now the possibility, to make use of gained product-instance-knowledge to attain conclusions for technical product improvements. So, for example, the insulating plate is affected by the environmental influences wetness and material consistence (compare Figure 7.21). Assuming a failure in material of the insulating plate with $P_M(\text{faulty}) = 1$ and $P_M(\text{ok}) = 0$, the result is the probability of a normal function of the insulating plate as follows:

$$\begin{aligned} P_I(n) &= P_I(Wetness_I\ high, Material_I\ faulty) \cdot\ P_F\ (high) \cdot P_M\ (faulty) \\ &+ P_I(Wetness_I\ low, Material_I\ faulty) \cdot\ P_F\ (low) \cdot P_M\ (faulty) \\ &\approx 0.098. \end{aligned}$$

If furthermore there is an additional failure in material with $P_F(\text{high}) = 1$, $P_F(\text{low}) = 0$, $P_M(\text{faulty}) = 1$ and $P_M(\text{ok}) = 0$, the probability of a normal function decreases again:

$$\begin{aligned} P_I(n) &= P_I(Wetness_I\ high, Material_I\ faulty) \cdot\ P_F\ (high) \\ &\cdot P_F\ (high) \cdot P_M\ (faulty) = 0.02. \end{aligned}$$

Assuming on the other hand that the probability of a high wetness in the domain of the insulating plate with $P_F(\text{high}) = 1$ and $P_F(\text{low}) = 0$, the probability of a functional insulating plate is decreasing less strong.

$$\begin{aligned} P_I(n) &= P_I(Wetness_I\ high, Material_I\ ok) \cdot\ P_F\ (high) \cdot P_M\ (ok) \\ &+ P_I(Wetness_I\ high, Material_I\ faulty) \cdot\ P_F\ (high) \cdot P_M\ (faulty) \\ &0,85 \cdot 1 \cdot 0.997 + 0.02 \cdot 1 \cdot 0,003 \approx 0.848. \end{aligned}$$

The developer of the wire-electro discharge machine also has the possibility to assume evidence values and then is able to apply the following calculation of inference. Following this he is able to detect factors effecting machine components. Respective components can then be improved in a way that they get more resistant against changes of certain effecting factors. Optionally following failures and misses of certain components can be avoided. In the case of the insulating plate shown above it can be seen that the normal functionality especially depends on the certain state of the material. Here certain improvement demands in materials' quality can be noticed. The construction of the insulating plate can, in a further step, be changed in a way that it is less vulnerable for wetness. Both improvements have the common target to increase the probability of a well working insulating plate and therefore antagonize a possible

failure of a following short circuit.
The product changes done in this phase finally serve the fact to maximize the availability of the machine, so that they fit the underlying target of Condition Monitoring. Concerning the influence of component's delays and the respective probabilities of component failures a cost optimized planning of maintenance dates can be attained. The created Bayesian network can also be used for bug fixing. So if, for example, it is detected that the electrode does not work correctly, the network provides the information that this, among other things, is caused by a malfunction of the spindle. The spindle, on the other side, is depending on several other components. So it can be concluded, starting with a malfunction on the electrode, on all causes of this failure to be able to remove the problem consequently in time.

Next consider the topic of analyzing the spindle in detail. For a variety of products there is a need for rotation-symmetric units. Micro structured components are used for example in process engineering as a piston of a micro pump or in medical technology as a micro needle [UP05]. The assembly of rotation-symmetric parts is primarily accomplished by means of turning or frictional processes. These manufacturing processes have only limited use with further miniaturization influenced by a great strain the production puts on the components. The production of rotation-symmetric micro components poses special problems for the manufacturing processes, as mechanical processing is mostly excluded for the structuring of these components [Rat03]. A combination of wire erosion and rotating movement is given for the production of rotation-symmetric components [QSS02]. The quality of the component with respect to geometric accuracy depends primarily on the quality of this equipment.
An interesting question for a knowledge engineer is the determination of the influence of the process characteristics on the quality of the work piece by monitoring the machining process. Therefore we made experiments in sensor-based tool condition monitoring with the chair *IT in Mechanical Engineering* from the University of Bochum. Diagnosing the root cause of process modifications when multiple potential sources of variations exist plays a key role in determining the influence of process characteristics. The intent is to explore the possibility of using multiple sensors and data obtained from sequential machining operations to diagnose the influence of process characteristics.
In a first step, a Bayesian network should visualize the measurement structure for modeling and evaluating uncertainty. In a second step, the influence of the rotation speed on the process can be augmented with utility and action nodes to evaluate and measure the service and quality inspection influence for decision making. The developed transformation framework produces sharp rule-based outputs for representing decision functions. A fuzzy inference tool leads to a fuzzy rule-based system which is generally intelligible. The machining process and the measurement of the influence of the process characteristics is illustrated in the flow chart in Figure 7.22. In the first step, appropriate measurements for recording during the experiment are identified and evaluation measures are designated. For data analysis a training data set is given as basis for the calculation of the underlying network structure. If available, additional *a priori* information

about causal relationships between random variables are determined. Next follows the determination of whether a causal link between particular nodes exists in the network. We apply the newly developed LAGD hill climbing algorithm for determining and evaluating the network structure and conditional probability table settings, which is meanwhile completely integrated in the WEKA environment. Once the network structure has been determined and values for

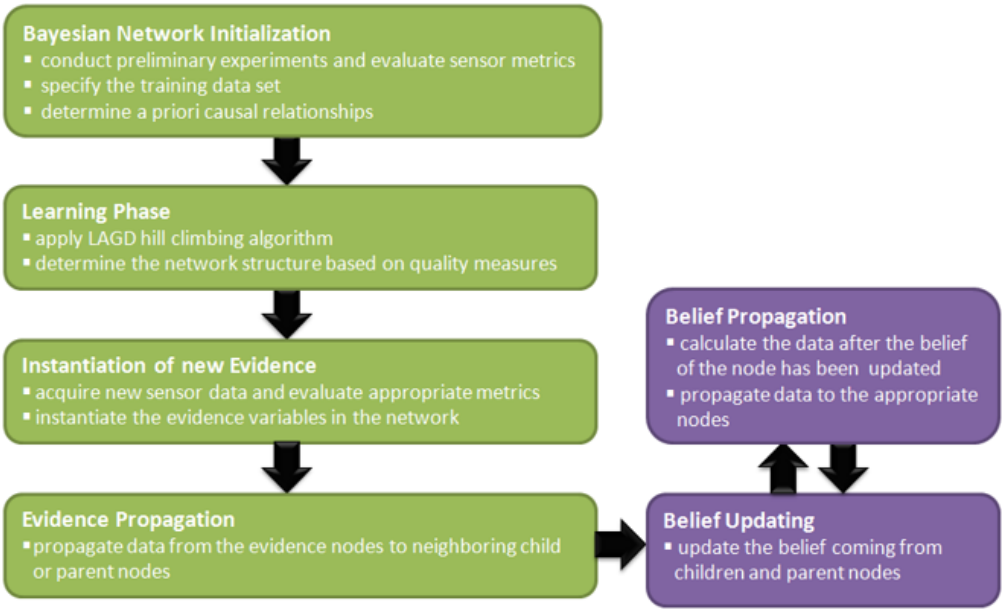


Figure 7.22.: BN evaluation representing the influence of the process characteristics

the conditional probability tables have been initialized, the network is ready for application. [DS05] have developed a process monitoring and diagnosis approach based on a Bayesian network for considering multiple process metrics from multiple sensor sources in sequential machining operations to identify the root cause of process variations and to provide with this a probabilistic confidence level of the diagnosis. They use the violet labeled process steps illustrated in Figure 7.22 for belief updating and belief propagation, when the evidence that has been instantiated into the network representation is propagated to the different nodes. It follows the updating of the beliefs of the evidence received from parent and child nodes. This procedure carried out until saturation is reached in the case, when there are no other nodes to propagate the evidence.

The electrical discharge machine under consideration is equipped with an adaptive feed control that realizes the motion of the machine table by means of impulse analysis of the discharge process. The determination of process parameters must be accomplished by research data [UP05]. Due to the fact that the rotation spindle is an accessory, the control of the rotating movement is not integrated into the machine control. The influence of rotation speed on the quality of the component is very high [QSS02]. Therefore the specifica-

tions for the rotation spindle must be in terms of a constant rotation speed without variation. Wear and tear of components of the rotation spindle like drive belt or bearings leads to such changes and therefore to the deterioration of performance. For instance, the radial run-out is influenced by the rotation speed and therefore, a varying rotation speed will lead to undesirable process results. Measurements of the process characteristics depends strongly on the condition of spindle components such as increased wear on the bearings or a loss of belt tension. Apply condition monitoring aims at the recognition of wear of mechanically stressed machine components. An interesting point of view is studying interrelationships between process signals and the wear state of machine components. For this purpose the dependence between the speed of rotation and the results of the process is presented. The achieved experimental results based on gathered rotation speed and process data received by the chair IT in mechanical engineering.

Table 7.3.: Influence of the rotation speed on the process without service

rotation speed n [1/min.]	max. discharge current ie[A]	discharge voltage Ue[V]	discharge duration te[ns]	off-load voltage U0[V]	off-load time td[μ,s]
none	15	16	342	93	2.0
30,90	0,0519	13,05	403,67	63,23	3,05
29,60	0,0497	13,18	379,45	64,08	3,05
38,80	0,0689	14,46	367,17	89,99	2,73
40,00	0,0704	15,03	395,24	87,00	2,69
50,10	0,1143	14,12	365,87	88,77	1,77
49,70	0,1122	14,43	361,12	94,51	1,83
101,70	0,0476	14,35	366,14	71,69	3,32
99,30	0,0505	14,55	395,25	66,59	3,06
298,80	0,0477	14,87	385,37	80,77	1,35
302,20	0,0516	14,67	389,67	80,53	1,26
502,20	0,0672	13,48	373,39	69,79	1,00
507,70	0,0680	12,69	399,87	69,68	1,02

For the analysis of the conditions of wire electrical discharge grinding measurements of these values, the electrical values, current and voltage, and the mechanical values, adaptive feed control and rotation speed, have been carried out. In the case of value *none* concerning the rotation speed a static test happens.

In this situation the work piece does not rotate. Two different types of influences are tested. Figure 7.23 illustrates the state without the needed service on the rotation spindle. In Figure 7.24 the service was done and the parameters of different rotation speeds are also tested. Both states were tested with rotation speed adjusted with 30 rpm, 40 rpm, 50 rpm, 100 rpm, 300 rpm and 500 rpm. As equipment we have chosen a 3R-6.600-S rotation spindle from the manufacturer System 3R International. [SR08]. The rotation speed is between 2 [1/min.]

Table 7.4.: Influence of the rotation speed on the process after service

rotation speed n [1/min.]	max. discharge current ie[A]	discharge voltage Ue[V]	discharge duration te[ns]	off-load voltage U0[V]	off-load time td[μ,s]
none	15	16	342	93	2.0
30,90	0,0704	13,04	296,76	87,43	1,83
29,60	0,0692	12,81	299,38	84,57	1,78
38,80	0,0707	12,96	345,44	76,42	3,60
40,00	0,0700	12,45	324,66	73,40	3,65
50,30	0,0820	12,84	379,44	81,45	1,75
49,60	0,0827	12,54	386,36	83,99	1,81
101,60	0,0811	11,68	364,79	79,27	4,11
99,80	0,0800	12,06	355,28	73,43	4,05
298,80	0,1000	11,43	359,14	74,91	2,18
302,80	0,0963	12,11	374,74	80,46	2,20
502,30	0,0886	12,13	394,34	78,08	1,54
508,90	0,0924	12,28	392,10	78,95	1,61

and 1800 [1/min.], whereas the spindle is powered with an electric motor that is driven by an alternating current.

The rotation speed of the spindle is regulated with the support of the control unit of the rotation spindle. The feed motion during the production of a circular groove occurs in this erosion machine in y-direction. The axes are adjusted by the adaptive feed control based on the process characteristics. The process parameters current and voltage are evaluated for this purpose. In the case of a short circuit discharge on account of the work piece and the tool electrode coming into contact with each other, the axis drives the work piece away from the erosion wire. If a greater gap occurs, the work piece is brought closer to the electrode wire until discharges in accordance with the preadjust process values are reached. The feed then accelerates. Periodic oscillation in the feed movement can caused by the relative movement of the work piece and electrode wire on account of the rotation and feed movement.

It is assumed that the process profile of the feed axis with known control parameters can be used as a characteristic value for the estimation of the rotation speed performance. Table 7.3 illustrates an extraction from the given data set, which includes rotation speed and process data without service on the rotation spindle. We have learned the Bayesian network structure as visualized in Figure 7.23 based on the LAGD hill climbing algorithm. The average of the electrical values in Table 7.3 and Table 7.4 are in the static case for the discharge current 15 ampere, for the discharge voltage 16 volt and for the discharge duration 342 nanoseconds. The open circuit voltage is 93 volt and the off-load duration is roughly 2.0 μs. The variation of the open circuit voltage and the off-load duration are minor. This is followed by the constant discharge with the rotation of the work piece the electronic characteristics changes with

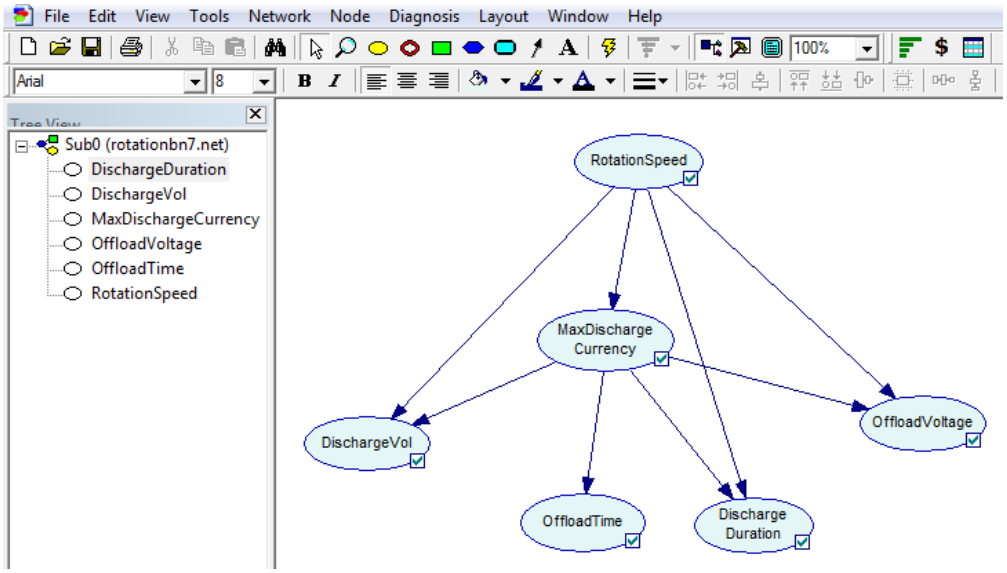


Figure 7.23.: Bayesian network for the process of WEDG

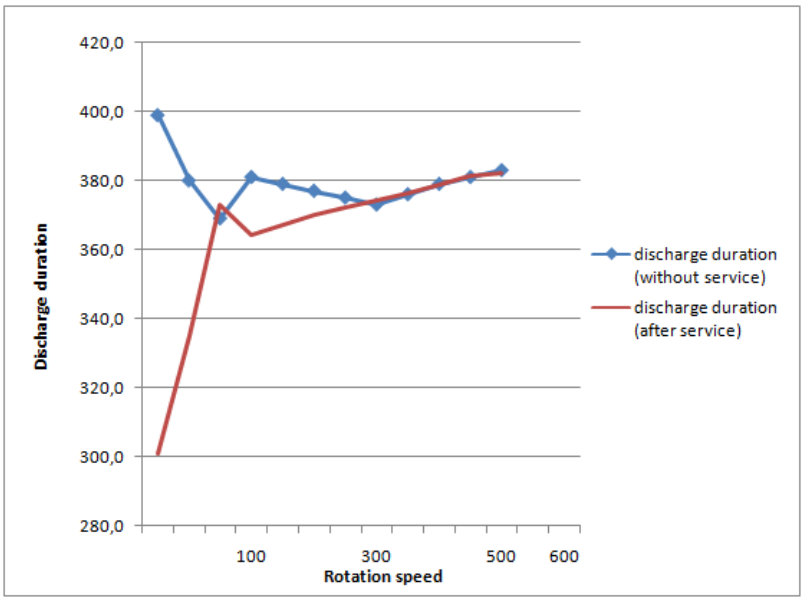


Figure 7.24.: Discharge duration as a function of the rotation speed

the rotation speed. The observed rotation speed has a strong influence on the machining. Before servicing the discharge voltage sinks with a rising rotation speed. The characteristics of the discharge duration and the discharge current at lower rotation speed less than 100 [1/min.] are temporary. With increasing rotation speed both variables rise (see Figure 7.24). After servicing similar characteristics can be evaluated (see also Table 7.4). The discharge voltage sinks with a rising speed and lies under the values of the discharge voltage before service. The amount is less than the values before servicing. The influence of the rotation speed can therefore said to be substantially less. Discharge duration and discharge current rise with rotational speed. The off-load duration tends to be longer with higher speeds and is longer in comparison to the values before servicing.

It was shown that maintaining the spindle leads to more favourable process conditions. The process efficiency increases. Correlations between revolutions per minute and process parameters were experimentally determined for the process of WEDG. Variations of the rotational speed have an impact on the characteristics of process parameters. Amongst others wear and tear of the rotation spindle may cause such effects.

7.4. Product Lifecycle Management Review

Today's PLM systems focus on supporting the early phases of the product lifecycle. Downstream phases, such as the product use phase, are only rudimentarily flanked and supported. We have presented a new approach for integrating the product use phase into the PLM concept. This concept stresses the possibility of using condition monitoring results collected in the operation phase when developing the next generation of a product. In addition, it propagates expanding the conventional product type PLM in view of managing product item data. In this regard, an extended metadata model for the integrated management of product type and item data in PLM systems was introduced and the resulting potential benefits were shown. The observable shift of manufacturers from selling only physical products to providing value added services is a factor underlining the importance of the product use phase for product development. With this in mind it gets more and more critical for product developers to consider in additional to conventional design parameters also the requirements of the later phases in the product lifecycle, e.g. in respect of efficient maintenance in the product use phase. In order to reduce the maintenance costs of an engine throughout its whole lifecycle, it is imperative for product developers to have instant access to maintenance histories and condition monitoring data for identifying critical engine parts. In this context, the transformation methodology applied to the problem of representing condition monitoring results for a rotation spindle using rules generated from a Decision network. For this particular case study, the quality measure indices have shown its best performance compared to two other Decision network representations discussed in Section 6.5. Furthermore, a fuzzy rule base representation created by the fuzzy inference tool FuzzIT is given and analyzed by the use of granularity measures

evaluating the position, core and size of linguistic terms. It makes sense to provide this rule-based representation for product developers easy accessible within the PLM system they use regularly. Within the scope of a feedback cycle product use information of the last product generation can be incorporated in an oriented fashion in the development of the next product generation and thus providing faster product improvements, lower development cost, increased product quality and lower maintenance expenses in the use phase.

Part V.

Conclusions and Outlook

8. Conclusions

This dissertation aimed at combining the advantages of decision-theoretic models with rule-based systems. The objective behind this is that serious drawbacks of both existing approaches became evident.

Decision-theoretic models can mimic the decision making mechanism from a declarative point of view based on an underlying rationality principle. Decision-theoretic models are not always suitable in time critical situations. In this case computational complexity becomes intractably large because of the fine discretization of both the probability distributions and the time axis. Hence we need another representation form which is more suitable for online-use and which produces decision outcomes a decision maker is able to make use of. Fuzzy logic provides a basis for representing uncertain and vague knowledge and forms a basis for human reasoning. The linguistic approach characterizes the system behaviour using rules of IF-THEN form with linguistic variables. We saw a need for establishing a new transformation process to generate rule-based representations from decision models, which provide an efficient execution architecture and represent knowledge in an explicit, intelligible way. This transformation process guides knowledge engineers through the transformation. This includes creating a rule-based system founded on a decision model which meets the aims of the decision process.

As opposed to the process of knowledge engineering with Bayesian networks, the new transformation process introduced in this thesis takes a step forward. It integrates various techniques we have previously introduced for building decision models that encompass decision making. The first transformation process step deals with learning the structure of Bayesian networks, for which we have developed a new parameterized structure learning algorithm named LAGD (Look-ahead in Good Directions). In addition to the generated Bayesian structure we need decision and utility nodes in order to be able to conceive a Decision-theoretic model. Utility functions are often used for the modeling of preferences. Procedures for modeling and learning preferences are useful for generating the utility nodes which are to be assessed in the present decision process. By specifying the problem-specific probability information we are able to eliminate alternatives in advance via a virtual dialogue with the decision maker and thus directly affect the number of decision nodes which have to be taken into account. The use of existing learning approaches to determine preferences and the specification of probability information subsequently enables us to model decision and utility nodes and generate a consolidated Decision network.

In the case of multiple decision nodes, we have to instantiate the first decision node with the first choice followed by the second node and so on. Algorithms which solve Decision networks convert the structure to decision trees or uses

clustering for inference in multiply connected networks. Jensen [Jen01] developed an object-oriented computational scheme for clustering implemented in the HUGIN system, an efficient and widely used tool for uncertain reasoning.

The availability of this cost free software package dealt as motivation for us to ask for an alternative approach in dealing with Decision networks on another level of abstraction. As a solution transforming decision networks in a rule-based representation was suggested and subsequently realized as AGoRuB algorithm. The idea here was to measure the utility loss, which occurs, if instead of the decision network the rule base is used for the determination of the decision. After the utility loss has been summed up over all relevant decision situations, the rule base is returned along with evaluation measures, which specify its overall quality.
The human-readable form of the achieved rule allows knowledge engineers to easily exchanges knowledge and interaction. Finally, fuzzy controllers allow to model uncertainty which can be evaluated with granulation measures that have been introduced and explored in this thesis. It should be noted that it becomes more challenging to determine the set of rules and membership functions to describe system behaviour, as the system complexity increases. Additional effort is needed to correctly tune membership functions and adjust rules.

All the previously mentioned techniques dealt with graphical models as basis. But, as has already been hinted, with the transformation framework there exist another, self-explanatory mechanism. After investigations into the innovative principles of graphical structure learning and Decision network transformation, both concepts were validated in the field of mechanical engineering. Today product lifecycle management focuses on supporting the early phases of the product lifecycle. Downstream phases, such as the product use phase, are currently not or only rudimentarily flanked and supported. Using data supplied from the product use phase is useful to verify the described mechanism, which enables us to extract product information and make decisions. In this context, we have presented an extension of the conventional product type PLM with regard to the management of product item data. The transformation process has been successfully applied to the problem of representing condition monitoring results for a rotation spindle in an intelligible form.

9. Outlook

In this thesis the development of a framework for approximating decision networks by fuzzy rule-based systems has been presented. The advantages that arise from this representation include information granulation and direct human readability, at least if the involved membership functions are restricted to semantically meaningful ones. Apart from aspects of efficiency, approximating a decision network by a fuzzy rule base is interesting for another reasonable point of view. Furthermore a fuzzy rule base can be evaluated for vague observations, whereas the decision network requires the evidence variables to be precisely specified. Thus, the compilation of a decision network into a fuzzy rule-base extends the applicability of a given decision model.

Naturally there remain several open questions some of which will be listed here as topics for future work. Since the transformation process is clearly supporting structure learning, automated reasoning, knowledge representation as well as semantic understanding the considered domain, further attention will have to focus on developing a general method for compiling Decision networks.

In order to further reduce the complexity of the rule base, a kind of feature selection could be useful. Selecting a subset of relevant features for building robust learning models is commonly used in data mining. Only the most important evidence variables are selected, whereas the other ones are ignored.

A next aspect to considerably improve the specification of probability information is the presence of uncertainty. Interacting with the decision maker to eliminate redundant actions can be achieved with fuzzy probability information which should express the degree of confidence for each probability distribution.

Instead of measuring the granularity of fuzzy rule-bases on the basis of core, size and position analysis, other criteria are of special interest for future work. Robustness measures the capability of coping well with modifications of membership functions, whereas relevance picks out significant rules which guarantee satisfactory solutions.

The use of the transformation process in product lifecycle management stressed the possibility of making use of condition monitoring results when developing the next generation of a product in view of managing product item data. The high integration of the product use phase can be seen as an important step towards a new generation of product lifecycle systems, which integrate data from all lifecycle phases in a holistic approach. Based on this new business models for manufacturers and service providers can be developed.

What remained undone are theoretical investigations into the complexity of a rule base, which can be defined in different ways. A general reasonable measure of complexity has to be determined and integrated in an overall quality function. With this, the transformation problem can be formalized as a general problem.

List of Figures

List of Tables

A. List of own Publications

A.1. Journal Contributions

[HFAN08b] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Competing fusion methods for electrical discharge machining. *Int. Journal on Multi-Sensor, Multi-Source Information Fusion, Elsevier [submitted]*, 2008.

[HFAN07] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Adaptive condition monitoring and diagnosis techniques for product and service improvement. *Journal Computers in Industry, Elsevier [submitted]*, 2007.

[Hol06a] A. Holland. Analysis and transformation of graphical models. *Int. Journal of Computational Intelligence*, 1(1):1–8, 2006.

A.2. Conference Contributions

[HFAN08a] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Competing fusion for bayesian applications. In B. Bouchon-Meunier and R. R. Yager, editors, *Proc. of the 12th Int. Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2008, Jun. 22-27, Málaga, Spain*, 2008.

[AFHN08a] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Integration of product use information into product lifecycle management. In H. Kaebernick, editor, *Proc. of the 15th CIRP Int. Conference on Life Cycle Engineering, LCE 2008, Mar. 17-19, University of New South Wales, Sydney, Australia*, 2008.

[AFHN08b] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Integration von Feedbackdaten aus der Produktnutzungsphase im Rahmen des PLM Konzepts. In M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, and A. Picot, editors, *Proc. zur Multikonferenz Wirtschaftsinformatik, MKWI 2008, Feb. 26-28, München, Germany*, pages 531–542, 2008.

[HF07b] A. Holland and M. Fathi. Quantitative and qualitative risk in IT portfolio management. In *Proc. of the IEEE Int. Conference on Systems, Man, and Cybernetics, SMC 2007, Oct. 7-10, Montreal, Canada*. IEEE Systems, Man, and Cybernetics Society (SMC), Piscataway, USA, 2007.

[HF07a] A. Holland and M. Fathi. Improving IT project portfolio prioritization by establishing a knowledge decision map. In J. Viedma, editor,

Proc. of the 8th European Conference on Knowledge Management, ECKM 2007, Sep. 6-7, Barcelona, Spain, 2007.

[AFHN07] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Advanced condition monitoring services in product lifecycle management. In W. Chang and J. Joshi, editors, *Proc. of the IEEE Int. Conference on Information Reuse and Integration, IRI 2007, Aug. 13-15, Las Vegas, USA*, pages 245–250. IEEE Systems, Man, and Cybernetics Society (SMC), Piscataway, USA, 2007.

[HFPS07] A. Holland, M. Fathi, H. Peuser, and T. Schmidt. Calculating risk of integration relations in application landscapes. In *Proc. of the 2007 IEEE Electro/Information Technology Conference, EIT 2007, May 17-20, Chicago, USA*. IEEE Press, 2007.

[HF06d] A. Holland and M. Fathi. Measuring business feedback cycles as enhancement of the support knowledge engineering process. In U. Reimer and D. Karagiannis, editors, *Proc. of the 6th Int. Conference on Practical Aspects of Knowledge Management, PAKM 2006, Nov. 30 - Dec. 1, Vienna, Austria*, volume 4333, pages 106–118. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin Heidelberg, 2006.

[HBF06] A. Holland, S. Berlik, and M. Fathi. Rule-based compilation of decision networks. In *Proc. of the Int. Conference on Systems, Man and Cybernetics, IEEE SMC 2006, Oct. 6-11, Taipei, Taiwan.* IEEE Press, 2006.

[HF06b] A. Holland and M. Fathi. Dynamic information granulation and information processing with imprecision and uncertainty. In H. V. Topping, G. Montero, and R. Montenegro, editors, *Proc. of the 5th Int. Conference on Engineering Computational Technology, ECT 2006, Sep. 12-15, Las Palmas, Spain.* Civil-Comp Press, 2006.

[HF06c] A. Holland and M. Fathi. Experience fusion as integration of distributed structured knowledge. In T. Vámos and P. Michelberger, editors, *Proc. of the World Automation Congress, WAC 2006, Jul. 24-26, Budapest, Hungary*, 2006.

[HB06] A. Holland and S. Berlik. Representation techniques for distributed knowledge models. In *Proc. of the Int. Conference on Artificial Intelligence, ICAI 2006, Jun. 26-29, Las Vegas, USA*. CSREA Press, 2006.

[HF06a] A. Holland and M. Fathi. Concurrent fusion via sampling and linOP aggregation. In V. Torra, Y. Narukawa, A. Valls, and J. Domingo-Ferrer, editors, *Proc. of the 3rd Int. Conference on Modeling Decisions for Artificial Intelligence, MDAI 2006, Apr. 3-5, Tarragona, Spain*, volume 3885. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin Heidelberg, 2006.

[Hol04a] A. Holland. A bayesian approach to model uncertainty in network balanced scorecards. In R. Matoušek and P. Ošmera, editors, *Proc. of the 10th Int. Conference on Soft Computing, MENDEL, Jun. 16-18, Brno, Czech Republic*, pages 134–138, 2004.

[Hol04b] A. Holland. Modeling uncertainty in decision support systems for customer call center. In B. Reusch, editor, *Computational Intelligence in Theory and Practice, Int. Conference 8th Fuzzy Days, Dortmund, Germany*, pages 580–588. Springer Series Advances in Soft Computing, 2004.

[Hol03] A. Holland. Planning time restricted logistic tours with fuzzy logic. In M. Wagenknecht and R. Hampel, editors, *Proc. of the Int. Conference on in Fuzzy Logic and Technology, EUSFLAT, Sep. 10-12, Zittau, Germany*, pages 329–336. European Society for Fuzzy Logic and Technology, EUSFLAT, 2003.

Bibliography

[Abr06] M. Abramovici. Evolution des Produkt Lifecycle Managements in einer veränderten Industrielandschaft. In *Proc. of Product Life live*, pages 13–20. VDE Verlag, Berlin, Offenbach, 2006.

[Abr07] M. Abramovici. Future trends in product lifecycle management. In *Proc. of the 17th CIRP Design Conference*, pages 311–320. Springer-Verlag, Berlin, 2007.

[AD00] J. D. Andrews and S. J. Dunnett. Event-tree analysis using binary decision diagrams. *Journal IEEE Transactions on Reliability*, 49(2):230–238, 2000.

[ADEK05] V. Arnold, H. Dettmering, T. Engel, and A. Karcher. *Product Lifecycle Management beherrschen.* Springer-Verlag, Berlin Heidelberg New York, 1st edition, 2005.

[ADF02] F. Abbattista, M. Degemmis, and N. Fanizzi. Learning user profiles for content-based filtering in e-commerce. In *Proc. Associazione Italiana per l'Intelligenza Artificiale, AIIA 2002, Siena, Italy*, 2002.

[AFHN07] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Advanced condition monitoring services in product lifecycle management. In W. Chang and J. Joshi, editors, *Proc. of the IEEE Int. Conference on Information Reuse and Integration, IRI 2007, Aug. 13-15, Las Vegas, USA*, pages 245–250. IEEE Systems, Man, and Cybernetics Society (SMC), Piscataway, USA, 2007.

[AFHN08a] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Integration of product use information into product lifecycle management. In H. Kaebernick, editor, *Proc. of the 15th CIRP Int. Conference on Life Cycle Engineering, LCE 2008, Mar. 17-19, University of New South Wales, Sydney, Australia*, 2008.

[AFHN08b] M. Abramovici, M. Fathi, A. Holland, and M. Neubach. Integration von Feedbackdaten aus der Produktnutzungsphase im Rahmen des PLM Konzepts. In M. Bichler, T. Hess, H. Krcmar, U. Lechner, F. Matthes, and A. Picot, editors, *Proc. zur Multikonferenz Wirtschaftsinformatik, MKWI 2008, Feb. 26-28, München, Germany*, pages 531–542, 2008.

[All98] T. Allweyer. Modellbasiertes Wissensmanagement. *IM Information Management*, 13(1):37–45, 1998.

[And96] P. T. Andreeva. Fuzzy sets. *Information and Control*, 2(2):70–76, 1996.

[Arn01] C. Arndt. *Information Measures. Information and its Description in Science and Engineering.* Springer Series in Reliability Engineering, Springer-Verlag, Berlin, 2001.

[AS04] M. Abramovici and S. Schulte. Study Benefits of PLM - The potential benefits of product lifecycle management in the automotive industry. volume 17. ZWF, Carl Hanser Verlag, 2004.

[AS06] M. Abramovici and S. Schulte. Plm - State of the art and trends. In K. Schützer, editor, *11th Seminario Internacional de Alta Tecnologia, Oct. 2006, Piracicaba, Brazil.* Inovacoes Tecnologicas no Desenolvimento do Produto, 2006.

[ASI07] A.-Saaksvuori and A. Immonen. *Product Lifecycle Management.* Springer-Verlag, Berlin Heidelberg New York, 2nd edition, 2007.

[Aug02] T. Augustin. Expected utility within a generalized concept of probability – a comprehensive framework for decision making under ambiguity. *Statistical Papers*, 43(1):5–22, 2002.

[Avo95] N. Avouris. Cooperating knowledge-based systems for environmental decision support. *Knowledge Based Systems*, 8:39–54, 1995.

[Ban98] H. Bandemer. *Ratschläge zum mathematischen Umgang mit Ungewißheit. Reasonable Computing.* Teubner Verlag, 1998.

[Bay58] T. Bayes. An essay towards solving a problem in the doctrine of chances. *Biometrica*, 45:296–315, 1958.

[BD97] P. J. Costa Branco and J. A. Dente. The application of fuzzy logic in automatic modeling electromechanical systems. *Fuzzy Sets and Systems*, 95(3):273–293, 1997.

[BDK98] J. S. Breese, D.+Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. Uncertainty in Artificial Intelligence, UAI-98, Madison, WI*, 1998.

[BDS04] R. Brafman, C. Domshlak, and S. Shimony. Qualitative decision making in adaptive presentation of structured information. *ACM Transactions on Information Systems*, 22(4):503–539, 2004.

[Bea03] M. J. Beal. *Variational algorithms for approximate bayesian inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, UK, 2003.

[Bee04] R. S. Beebe. *Predictive Maintenance of Pumps Using Condition Monitoring*. Elsevier Advanced Technology, Kidlington, Oxford, United Kingdom, 2004.

[Bil98] T. Bilgic. Interval-valued preference structures. *European Journal of Operational Research*, 105:162–183, 1998.

[Bis08] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics, Springer-Verlag, Berlin, 2nd edition, 2008.

[BK02] C. Borgelt and R. Kruse. *Graphical Models. Methods for Data Analysis and Mining*. John Wiley & Sons, West Sussex, United Kingdom, 2002.

[BKKL05] H. Bae, S. Kim, Y. T. Kim, and S. H. Lee. Application of time-series data mining for fault diagnosis of induction motors. In *Computational Science and Its Applications, ICCSA 2005*, volume 3483 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Heidelberg, 2005.

[BL04] R. J. Brachman and H. J. Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann, San Francisco, USA, 2004.

[Bly98] J. Blythe. *Planning under uncertainty in dynamic domains*. PhD thesis, Carnegie Mellon University, Pittsburgh, Computer Science Department, 1998.

[BM03] B. Bouchon-Meunier. *Aggregation and Fusion of Imperfect Information*, volume 12. Studies in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg, 2003.

[BOM06] O. Blatnik, H. Orbanič, and C. Masclet. Water jet machining of micro sinking electrical discharge machining tools. In *2nd Int. Conference on Multi-Material Micro manufacture, 4M Network of Excellence 2006, Grenoble, September 20-22, 2006*, Lecture Notes in Computer Science, pages 385–388, 2006.

[Bor00] C. Borgelt. *Data mining with graphical models*. PhD thesis, Department of Computer Science, University of Magdeburg, Germany, 2000.

[BP98] D. Billsus and M. Pazzani. Learning collaborative information filters. In *Proc. 15th Int. Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA*, pages 46–54, 1998.

[Bür01] R. Bürgel. *Prozessanalyse an spannenden Werkzeugmaschinen mit digital geregelten Antrieben*. PhD thesis, Technische Universität München, Fakultät für Maschinenwesen, 2001.

[Bra83] R. J. Brachman. What IS-A is and isn't. an analysis of taxonomic links in semantic networks. *Journal IEEE Computer*, 16(10), 1983.

[BS84] B. G. Buchanan and E. H. Shortliffe. Uncertainty and evidential support. In B. D. Buchanan and E. H. Shortliffe, editors, *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading, MA: Addison-Wesley, 1984.

[BSCC96] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Chooper. The ALARM monitoring system. A case study with two probabilistic inference techniques for belief networks. In *Proc. Reasoning with Uncertainty in Robotics, RUR 1995*, volume 1093 of *Lecture Notes in Computer Science*, pages 52–90. Springer-Verlag, Berlin Heidelberg, 1996.

[Bur02] R. Burke. Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[CdC07] B. M. Colosimo and E. del Castillo. *Bayesian Process Monitoring, Control and Optimization*. Chapman & Hall, Taylor & Francis Group, Boca Raton, USA, 2007.

[CDL07] R. G. Cowell, P. David, and S. L. Lauritzen. *Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks*. Springer-Verlag, Berlin Heidelberg, 2nd edition, 2007.

[CF01] C. Carlsson and R. Fuller. On possibilistic mean value and variance of fuzzy numbers. *Fuzzy Sets and Systems*, 142:315–326, 2001.

[CG01] J. Cheng and R. Greiner. Learning bayesian belief network classifiers: algorithms and system. In *Lecture Notes in Artificial Intelligence*, volume 2056, AI 2001, pages 141–151. Springer-Verlag, Berlin Heidelberg, 2001.

[CKO01] U. Chajewska, D. Koller, and D. Ormoneit. Learning an agent's utility function by observing behaviour. In *Proc. of the 18th Int. Conf. on Machine Learning, ICML-01*, pages 35–42, 2001.

[Cou98] J. B. Couch. Disease management: An overview. In J. B. Couch, editor, *The Health Care Professional's Guide to Dis-*

ease Management: Patient-Centered Care for the 21st Century. Aspen Publishers, 1998.

[CPL+03] M. Campos, J. Palma, B. Llamas, A. Gonzales, M. Menarguez, and R. Marin. Temporal data management and knowledge acquisition issues in medical decision support systems. In *Computer Aided Systems Theory, EUROCAST 2003*, volume 2809 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Heidelberg, 2003.

[CRAMBMNF06] N. Cruz-Ramirez, H. G. Acosta-Mesa, R.-E. Barrientos-Martinez, and L.-A. Nava-Fernandez. How good are the bayesian information creation and minimum description length principle for model selection? A bayesian network analysis. In *Lecture Notes in Artificial Intelligence*, volume 4293, MICAI 2006, pages 494–504. Springer-Verlag, Berlin Heidelberg, 2006.

[CSHL05] M. C. Chen, C. T. Su, C. C. Hsu, and Y. W. Liu. Data transformation in spc semiconductor machinery control: A case of monitoring particles. *International Journal of Production Research*, 43(13):2759–2773, 2005.

[CSK04] R. Chen, K. Sivakumar, and H. Kargupta. Collective mining of bayesian networks from distributed heterogeneous data. In *Knowledge and Information Systems*, volume 6, pages 164–187. Springer-Verlag, London, 2004.

[CT93] C. Chieh-Tien. On strong consistency of the fuzzy generalized nearest neighbor rule. *Fuzzy Sets and Systems*, 60:273–281, 1993.

[Dav98] A. Davies. *Handbook of Condition Monitoring. Techniques and Methodology*. Springer Netherland, 1st edition, 1998.

[dBK94] B. de Baets and E. Kerre. Fuzzy relations and applications. *Advances in Electronics and electron Physics, Academic Press*, 89:255–324, 1994.

[dBvdWK97] B. de Baets, B. van de Walle, and E. Kerre. Minimal definitions of classical and fuzzy preference structures. In *Proc. of the Annual Meeting of the North American Fuzzy Information Processing Society, NAFIPS '97, Syracuse, New York, USA*, pages 299–304, 1997.

[DGL97] T. Dean, R. Givan, and S. Leach. Model reduction techniques for computing approximately optimal solutions for markov decision processes. In *Proc. 13th Annual Conference on Uncertainty in Artificial Intelligence*, pages 124–131, 1997.

[DHR07] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An Introduction to Fuzzy Control*. Springer-Verlag, Berlin Heidelberg New York, 2nd edition, 2007.

[DL93] P. Dagum and M. Luby. Approximating probabilistic inference in bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.

[DLP94] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In D. M. Gabbay, C. H. Hogger, and J. .A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3. Oxford University Press, 1994.

[DP93] D. Dubois and H. Prade. *Readings in Fuzzy Sets for Intelligent Systems*. Morgan Kaufmann, San Mateo, California, USA, 1993.

[DR82] H. Dinges and H. Rost. *Prinzipien der Stochastik*. Teubner Verlag, Stuttgart, 1st edition, 1982.

[DS05] S. Dey and J. A. Stori. A bayesian network approach to root cause diagnosis of process variations. *Int. Journal of Machine Tools & Manufacture*, 45:75–91, 2005.

[Edl01] A. Edler. *Nutzung von Felddaten in der qualitätsgetriebenen Produktentwicklung und im Service*. PhD thesis, Fakultät Verkehrs- und Maschinensysteme, Technische Universität Berlin, 2001.

[Edw00] D. Edwards. *Introduction to Graphical Modelling*. Springer Texts in Statistics, Springer Verlag, Berlin, 2nd edition, 2000.

[EJM06] C. Emmanouilidis, E. Jantunen, and J. MacIntyre. Flexible software for condition monitoring, incorporating novelty detection and diagnostics. *Journal Computers in Industry*, 57(6):516–527, 2006.

[ES08] M. Eigner and R. Stelzer. *Produktdatenmanagement-Systeme*. Springer-Verlag, Berlin, 2nd edition, 2008.

[EVW05] J. Espinosa, J. Vandewalle, and V. Wertz. *Fuzzy Logic, Identification and Predictive Control*. Springer-Verlag, Advances in Industrial Control AIC, London Berlin Heidelberg, 2005.

[EW03] F. Eisenführ and M. Weber. *Rationales Entscheiden*. Springer-Verlag, Berlin Heidelberg New York, 4th edition, 2003.

[FDS93] M. Fathi, D. Danebrock, and J. P. Stöck. A fuzzy expert system for online diagnosis. *Uncertainty in Intelligent Systems, Elsevier Science Publishers*, 1993.

[FEJ00] A. Flycht-Eriksson and A. Jönsson. Dialogue and domain knowledge management in dialogue systems. In *Proceedings of the 1st SIGdial Workshop on Discourse and Dialogue*, volume 10, pages 121–130, 2000.

[Feu83] M. Feurer. *Elektroerosive Metallbearbeitung. Materialabtrag durch Funkenerosion.* Vogel Buchverlag, Würzburg, 1983.

[FH97] M. Fathi and L. Hildebrand. Model-free optimization of fuzzy rule-based systems using evolution strategies. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 27(2):270–277, 1997.

[FH00] A. Friis-Hansen. *Bayesian networks as a decision support tool in marine applications.* PhD thesis, Department of Naval Architecture and Offshore Engineering, Technical University of Denmark, Lyngby, Denmark, 2000.

[FHB06] K. Främling, M. Harrison, and J. Brusey. Globally unique product identifiers - requirements and solutions to product lifecycle management. In A. Dolgui, G. Morel, and C. E. Pereira, editors, *Proc. of the 12th IFAC Symposium on Information Control Problems in Manufacturing, INCOM 2006, May 17-19, Saint-Etienne, France*, pages 855–860, 2006.

[FHMV95] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge.* MIT Press, 1st edition, 1995.

[FMS05] A. Freßmann, R. Maximini, and T. Sauer. Towards collaborative agent-based knowledge support for time-critical and business-critical processes. In *Lecture Notes in Artificial Intelligence*, volume 3782, 3rd Conf. Professional Knowledge Management, pages 420–430. Springer-Verlag, Berlin Heidelberg, 2005.

[For05] W. Forsthoffer. *Reliability Optimization through Component Condition Monitoring and Root Cause Analysis.* Elsevier Advanced Technology, 2005.

[FR94a] J. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multi-Criteria Decision Support.* Kluwer Academic Publishers, Dodrecht Boston London, 1994.

[FR94b] J. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support.* Kluwer Academic Publishers, 1994.

[FR06] K. Främling and L. Rabe. Enriching product information during the product lifecycle. In A. Dolgui, G. Morel, and C. E. Pereira, editors, *Proc. of the 12th IFAC Symposium on*

Information Control Problems in Manufacturing, INCOM 2006, May 17-19, Saint-Etienne, France, pages 861–866, 2006.

[Fri98] N. Friedman. The bayesian structural em algorithm. In *Proc. 14th Conf. Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, CA*, 1998.

[GD93] A. J. Gonzales and D. D. Dankel. *The Engineering of Knowledge-Based Systems. Theory and Practice.* Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[GDX04] M. Ge, R. Du, and Y. S. Xu. Condition monitoring using marginal energy and hidden markov model. *Journal Control and Intelligent Systems*, 32(1):1322–1332, 2004.

[GH01] B. A. Gran and A. Helminen. A bayesian belief network for reliability assessment. In *Lecture Notes in Artificial Intelligence*, volume 2187, Safecomp 2001, pages 35–45. Springer-Verlag, Berlin Heidelberg, 2001.

[GH02] H. Guo and W. Hsu. A survey of algorithms for real-time bayesian network inference. 2002.

[Gha98] Z. Ghahramani. Learning dynamic bayesian networks. In *Lecture Notes in Artificial Intelligence*, volume 1387, Adaptive Processing, pages 168–197. Springer-Verlag, Berlin Heidelberg, 1998.

[GL06] A. Gupta and C. Lawsirirat. Strategically optimum maintenance of monitoring-enabled multicomponent systems using continuous-time jump deterioration models. *Journal of Quality in Maintenance Engineering*, 12(3):306–329, 2006.

[GNOT92] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave and information tapestry. *ECommunications of the ACM*, 35(12):61–70, 1992.

[Grü05] R. Grütter. IT Servicemanagement nach ITIL - Unterschiede zwischen Anbietern und Kunden. Master's thesis, Universität Zürich, Institut für Informatik, 2005.

[GRFM06] J. N. D. Gupta, R. Ruiz, J. W. Fowler, and J. S. Mason. Operational planning and control of semiconductor wafer production. *Production Planning and Control*, 17(7):639–647, 2006.

[GS04] G. Grimmett and D. Stirzaker. *Probability and Random Processes.* Oxford University Press, 3rd edition, 2004.

[Guo05] Y. Guo. *Algorithmen zur On-Bord Diagnose von Fahrwerksschäden an Schienenfahrzeugen.* PhD thesis, Technische Universität Berlin, Fakultät III, Prozesswissenschaften, 2005.

[GW06] R. X. Gao and L. Wang. *Condition Monitoring and Control for Intelligent Manufacturing.* Springer Series in Advanced Manufacturing, Springer-Verlag London, UK, 2006.

[GZ06] Y. Gao and Q. Zhang. A wavelet packet and residual analysis based method for hydraulic pump health diagnosis. In *Proc. of the I Mech E Part D Journal of Automobile Engineering*, volume 220, number 6, pages 735–745, 2006.

[HA08] HUGIN-API. HUGIN API reference manual version 6.7. *Homepage http://developer.hugin.com/documentation/API_Manuals/*, 2008.

[HB06] A. Holland and S. Berlik. Representation techniques for distributed knowledge models. In *Proc. of the Int. Conference on Artificial Intelligence, ICAI 2006, Jun. 26-29, Las Vegas, USA*. CSREA Press, 2006.

[HBF06] A. Holland, S. Berlik, and M. Fathi. Rule-based compilation of decision networks. In *Proc. of the Int. Conference on Systems, Man and Cybernetics, IEEE SMC 2006, Oct. 6-11, Taipei, Taiwan.* IEEE Press, 2006.

[HBH98] E. Horvitz, J. Breese, and D. Heckerman. Bayesian user modeling for inferring goals of needs of software users. In *Proc. of the 14th Conference on Uncertainty in Artificial Intelligence, UAI-98*, pages 256–265, 1998.

[HE04] E. R. Hruschka and N. Ebecken. Feature selection by bayesian networks. In *Lecture Notes in Artificial Intelligence*, volume 3060, Canadian AI 2004, pages 370–379. Springer-Verlag, Berlin Heidelberg, 2004.

[Hec95] D. Heckerman. A tutorial on learning bayesian networks. Technical report, MSR-TR-95-06, Microsoft Research, USA, 1995.

[Hen88] M. Henrion. Propagating uncertainty in bayesian networks by probabilistic logic sampling. *Uncertainty in Artificial Intelligence*, 2:149–163, 1988.

[HF06a] A. Holland and M. Fathi. Concurrent fusion via sampling and linOP aggregation. In V. Torra, Y. Narukawa, A. Valls, and J. Domingo-Ferrer, editors, *Proc. of the 3rd Int. Conference on Modeling Decisions for Artificial Intelligence, MDAI*

2006, Apr. 3-5, Tarragona, Spain, volume 3885. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin Heidelberg, 2006.

[HF06b] A. Holland and M. Fathi. Dynamic information granulation and information processing with imprecision and uncertainty. In H. V. Topping, G. Montero, and R. Montenegro, editors, *Proc. of the 5th Int. Conference on Engineering Computational Technology, ECT 2006, Sep. 12-15, Las Palmas, Spain*. Civil-Comp Press, 2006.

[HF06c] A. Holland and M. Fathi. Experience fusion as integration of distributed structured knowledge. In T. Vámos and P. Michelberger, editors, *Proc. of the World Automation Congress, WAC 2006, Jul. 24-26, Budapest, Hungary*, 2006.

[HF06d] A. Holland and M. Fathi. Measuring business feedback cycles as enhancement of the support knowledge engineering process. In U. Reimer and D. Karagiannis, editors, *Proc. of the 6th Int. Conference on Practical Aspects of Knowledge Management, PAKM 2006, Nov. 30 - Dec. 1, Vienna, Austria*, volume 4333, pages 106–118. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin Heidelberg, 2006.

[HF07a] A. Holland and M. Fathi. Improving IT project portfolio prioritization by establishing a knowledge decision map. In J. Viedma, editor, *Proc. of the 8th European Conference on Knowledge Management, ECKM 2007, Sep. 6-7, Barcelona, Spain*, 2007.

[HF07b] A. Holland and M. Fathi. Quantitative and qualitative risk in IT portfolio management. In *Proc. of the IEEE Int. Conference on Systems, Man, and Cybernetics, SMC 2007, Oct. 7-10, Montreal, Canada*. IEEE Systems, Man, and Cybernetics Society (SMC), Piscataway, USA, 2007.

[HFAN07] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Adaptive condition monitoring and diagnosis techniques for product and service improvement. *Journal Computers in Industry, Elsevier [submitted]*, 2007.

[HFAN08a] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Competing fusion for bayesian applications. In B. Bouchon-Meunier and R. R. Yager, editors, *Proc. of the 12th Int. Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2008, Jun. 22-27, Málaga, Spain*, 2008.

[HFAN08b] A. Holland, M. Fathi, M. Abramovici, and M. Neubach. Competing fusion methods for electrical discharge machining. *Int. Journal on Multi-Sensor, Multi-Source Information Fusion, Elsevier [submitted]*, 2008.

[HFPS07] A. Holland, M. Fathi, H. Peuser, and T. Schmidt. Calculating risk of integration relations in application landscapes. In *Proc. of the 2007 IEEE Electro/Information Technology Conference, EIT 2007, May 17-20, Chicago, USA*. IEEE Press, 2007.

[HH03] V. Ha and P. Haddawy. Similarity of personal preferences: theoretical foundations and empirical analysis. *Artificial Intelligence*, 146:149–173, 2003.

[HHG02] C. Harris, X. Hong, and Q. Gan. *Adaptive Modelling, Estimation and Fusion from Data. A Neurofuzzy Approach.* Springer-Verlag, Berlin Heidelberg New York, 1st edition, 2002.

[HKBR99] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. of the 1999 Conf. on Research and Development in Information Retrieval*, 1999.

[HM04] D. L. Hall and S. A.H. McMullen. *Mathematical Techniques in Multisensor Data Fusion.* Artech House Information Warfare Library, 2004.

[HMS+06] R. W. Hyers, J. G. Mcgowan, K. L. Sullivan, J. F. Manwell, and B. C. Syrett. Condition monitoring and prognosis of utility scale wind turbines. *Energy Materials: Materials Science and Engineering for Energy Systems*, 1(3):187–203, 2006.

[HMW+07] T. J. Harvey, S. Morris, L. Wang, R. J. K. Wood, and H. E. G. Powrie. Real-time monitoring of wear debris using electrostatic sensing techniques. In *Proc. of the I Mech E Part J Journal of Engineering Tribology*, volume 221, number 1, pages 27–40, 2007.

[Hol03] A. Holland. Planning time restricted logistic tours with fuzzy logic. In M. Wagenknecht and R. Hampel, editors, *Proc. of the Int. Conference on in Fuzzy Logic and Technology, EUSFLAT, Sep. 10-12, Zittau, Germany*, pages 329–336. European Society for Fuzzy Logic and Technology, EUSFLAT, 2003.

[Hol04a] A. Holland. A bayesian approach to model uncertainty in network balanced scorecards. In R. Matoušek and

P. Ošmera, editors, *Proc. of the 10th Int. Conference on Soft Computing, MENDEL, Jun. 16-18, Brno, Czech Republic*, pages 134–138, 2004.

[Hol04b] A. Holland. Modeling uncertainty in decision support systems for customer call center. In B. Reusch, editor, *Computational Intelligence in Theory and Practice, Int. Conference 8th Fuzzy Days, Dortmund, Germany*, pages 580–588. Springer Series Advances in Soft Computing, 2004.

[Hol06a] A. Holland. Analysis and transformation of graphical models. *Int. Journal of Computational Intelligence*, 1(1):1–8, 2006.

[Hol06b] K. Holmberg. Prognostics of industrial machinery availability. Technical report, Project Dynamic Decisions in Maintenance, EC Sixth Framework Programme, VTT Technical Research Centre of Finland, 9 2006.

[HPRZ02] S. Har-Pered, D. Roth, and D. Zimak. Constraint classification: a new aproach to multiclass classification. In *Proc. 13th Int. Conf. on Algorithmic Learning Theory*, pages 365–379, 2002.

[HR92] J. Hirshleifer and J. G. Riley. *An analytics of uncertainty and information.* Cambridge University Press, Cambridge, UK, 1992.

[HTB02] E. Hering, J. Triemel, and H. P. Blank. *Qualitätsmanagement für Ingenieure.* VDI Buch, Springer-Verlag, Berlin Heidelberg New York, 5th edition, 2002.

[HUG08] HUGIN. Hugin expert. *Homepage http://www.hugin.com/*, 2008.

[Ise05] R. Isermann. *Fault Diagnosis Systems. An Introduction from Fault Detection to Fault Tolerance.* Springer-Verlag, Berlin Heidelberg, 1st edition, 2005.

[Jan06] E. Jantunen. Diagnosis of tool wear based on regression analysis and fuzzy logic. *IMA Journal of Management Mathematics*, 17(1):47–60, 2006.

[Jen01] F. V. Jensen. *Bayesian Networks and Decision Graphs.* Statistics for Engineering and Information Science, Springer-Verlag, Berlin Heidelberg New York, 1st edition, 2001.

[JH03] B. C. Jiang and C. H. Hsu. Development of a fuzzy decision model for manufacturability evaluation. *Journal of Intelligent Manufacturing*, 14:169–181, 2003.

[JHRW02] S. Jie, G. S. Hong, M. Rahman, and Y. S. Wong. Feature extraction and selection in tool condition monitoring system. In *Lecture Notes in Artificial Intelligence*, volume 2557, AI 2002, pages 487–497. Springer-Verlag, Berlin Heidelberg, 2002.

[JHS01] S. Jablonski, S. Horn, and M. Schlundt. Process oriented knowledge management. In *11th Int. Workshop on Research Issues in Data Engineering, Heidelberg, Germany*, pages 77–84. IEEE Comp. Society, 2001.

[JLO90] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4:269–280, 1990.

[JM99] L. C. Jain and N. M. Martin. *Fusion of Neural Networks, Fuzzy Sets, and Genetic Algorithms*. CRC Press, Boca Raton London New York, 1st edition, 1999.

[JMS05] L. Jin, T. Mashita, and K. Suzuki. An optimal policy for partially observable markov decision processes with non-independent monitors. *Journal of Quality in Maintenance Engineering*, 11(3):228–238, 2005.

[Joy99] J. M. Joyce. *The Foundations of Causal Decision Theory*. Cambridge University Press, Cambridge, UK, 1999.

[KD01] H. J. Kushner and P. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*. Springer-Verlag, New York, USA, 2nd edition, 2001.

[KF88] G. J. Klir and T. A. Folger. *Fuzzy Sets, Uncertainty, and Information*. Englewood Cliffs, NJ:Prentice-Hall, 1988.

[KGK95] R. Kruse, J. Gebhardt, and F. Klawonn. *Fuzzy-Systeme*. Teubner-Verlag, Stuttgart, 2nd edition, 1995.

[KHS03] S. Kim, H. Hwang, and E. Suh. A process-based approach to knowledge flow analysis. *Knowledge and Process Management*, 10(4):260–276, 2003.

[KKG+06] N. Kumar, K. Kennedy, K. Gildersleeve, R. Abelson, C. M. Mastrangelo, and D. C. Montgomery. A review on yield modelling techniques for semiconductor manufacturing. *International Journal of Production Research*, 44(23):5019–5036, 2006.

[Kla06] A. Klahold. *CRIC: Kontextbasierte Empfehlung unstrukturierter Texte in Echtzeitumgebungen*. PhD thesis, Department of Electrical Engineering and Computer Science, University of Siegen, 2006.

[Kle92] G. D. Kleiter. Bayesian diagnosis in expert systems. *Artificial Intelligence*, 54:1–32, 1992.

[KMM06] A. Kassim, Z. Mian, and M. Mannan. Condition classification using hidden markov model based on fractal analysis of machined surface textures. *Machine Vision and Applications*, 17(5):327–336, 2006.

[KN04] K. B. Korb and A. E. Nicholson. *Bayesian Artificial Intelligence*. Chapman & Hall, Series in Computer Science and Data Analysis, London, UK, 2004.

[Kor90] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42:189–211, 1990.

[Kos92] B. Kosko. *Neural Networks and Fuzzy Systems*. Prentice Hall, 3rd edition, 1992.

[KP93] N. I. Karacapilidis and C. P. Pappis. A comparative assessment of measures of similarity of fuzzy values. *Fuzzy Sets and Systems*, 56:171–174, 1993.

[Kra96] P. Krause. Learning probabilistic networks. Technical report, Philips Research Laboratories, United Kingdom, 1996.

[KSP05] P. Koomsap, N. I. Shaikh, and V. V. Prabhu. Integrated process control and condition-based maintenance scheduler for distributed manufacturing control systems. *International Journal of Production Research*, 43(8):1625–1641, 2005.

[Lau05] H. Laux. *Entscheidungstheorie*. Springer-Verlag, Berlin Heidelberg New York, 4th edition, 2005.

[LC06] S. Lim and S. B. Cho. Online learning of bayesian network parameters with incomplete data. In *Lecture Notes in Artificial Intelligence*, volume 4114, ICIC 2006, pages 309–314. Springer-Verlag, Berlin Heidelberg, 2006.

[LCZL07] H. Li, X. Chen, H. Zeng, and X. Li. Embedded tool condition monitoring for intelligent machining. *International Journal of Computer Applications in Technology*, 28(1):74–81, 2007.

[LGS07] P. Lucas, J. A. Gámez, and A. Salmerón. *Advances in Probabilistic Graphical Models*. Studies in Fuzziness and Soft Computing, Springer-Verlag, Berlin Heidelberg New York, 2007.

[LHQB06] T. Liao, G. Hua, J. Qu, and P. J. Blau. Grinding wheel condition monitoring with hidden markov model-based clustering methods. *Machining Science and Technology*, 10(4):511–538, 2006.

[LLLJ03] X. Luo, J. Lee, H. Leung, and N. Jennings. Prioritized fuzzy constraint satisfaction problems: axioms, instantiation and validation. *Fuzzy Sets and Systems*, 136:151–188, 2003.

[LS93] K. S. Leung and Y. T. So. Consistency checking for fuzzy expert systems. *Int. Journal of Approximate Reasonning*, 9:263–282, 1993.

[Luc01] P. Lucas. Expert knowledge and its role in learning bayesian networks in medicine. In *Lecture Notes in Artificial Intelligence*, volume 2101, AIME 2001, pages 156–166. Springer-Verlag, Berlin Heidelberg, 2001.

[Lui07] L. Luini. *Uncertain Decisions. Bridging Theory and Experiments.* Kluwer Academic Publishers, Dordrecht, Netherlands, 2nd edition, 2007.

[LY05] A. W. Labib and M. N. Yuniarto. Intelligent real time control of disturbances in manufacturing systems. *Journal of Manufacturing Technology Management*, 16(8):864–889, 2005.

[Mah07] R. P.S. Mahler. *Statistical Multisource-Multitarget Information Fusion.* Artech House Information Warfare Library, 2007.

[Mai04] R. Maier. *Knowledge Management Systems. Information and Communication Technologies for Knowledge Management.* Springer-Verlag, Berlin Heidelberg New York, 2nd edition, 2004.

[Mai06] L. Maillart. Maintenance policies for systems with condition monitoring and obvious failures. *IIE Transactions*, 38(6):463–475, 2006.

[MC04] S. Mani and G. F. Cooper. *Causal discovery using a bayesian local causal discovery algorithm.* Medinfo, IOS Press, 2004.

[MGI06] C. Marques, J. N. D. Gupta, and J. P. Ignizio. Improving preventive maintenance scheduling in semiconductor fabrication facilities. *Machining Science and Technology*, 17(7):742–754, 2006.

[Mil03] E. Miller. State of the product lifecycle management industry. In *Proc. of the CIMdata PLM-Conference, Dearborn, USA*, 2003.

[Mil07] N. R. Milton. *Knowledge Acquisition in Practice. A Step-by-Step Guide.* Springer-Verlag, London, 2007.

[Mit97] T. Mitchell. *Machine Learning.* McGraw-Hill, New York, USA, 1997.

[MKT07] A. Mittal, A. Kassim, and T. Tan. *Bayesian Network Technologies - Applications and Graphical Models*. IGI Global, Hershey, United States, 2007.

[MLM06] S. Meganck, P. Leray, and B. Manderick. Learning causal bayesian networks from observations and experiments: a decision theoretic approach. In V. Torra, Y. Narukawa, A. Valls, and J. Domingo-Ferrer, editors, *Proc. of the 3rd Int. Conference on Modeling Decisions for Artificial Intelligence, MDAI 2006, Apr. 3-5, Tarragona, Spain*, volume 3885. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin Heidelberg, 2006.

[Mor05] Z. Moravej. Evolving neural nets for protection and condition monitoring of power transformer. *Electric Power Components and Systems*, 33(11):1229–1236, 2005.

[Mou01] J. Moubray. *Reliability Centered Maintenance*. Industrial Press, Inc., 2nd edition, 2001.

[MQB07] H. Meier, N. Quade, and S. Binner. *Neue Wege zu einer höheren Verfügbarkeit*. VDI-Z, Springer-VDI Verlag, 5 2007.

[MS06] G. S. May and C. J. Spanos. *Fundamentals of Semiconductor Manufacturing and Process Control*. Wiley & Sons, 1st edition, 2006.

[MT06] J. R. Montoya-Torres. A literature survey on the design approaches and operational issues of automated wafer-transport systems for wafer fabs. *Production Planning and Control*, 17(7):648–663, 2006.

[Mye05] B. A. Myers. Using handhelds for wireless remote control of PCs and appliances. *Journal Interacting with Computers, Elsevier*, 17:251–264, 2005.

[Nea90] R. E. Neapolitan. *Probabilistic reasoning in Expert Systems*. John Wiley & Sons, New York, USA, 1990.

[Nea04] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, New Jersey, USA, 2004.

[NFN00] M. Neil, N. Fenton, and L. M. Nielsen. Building large-scale bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, 2000.

[NPP06] V. Namasivayam, A. Pathak, and V. Prasanna. Scalable parallel implementation of bayesian network to junction tree conversion for exact inference. In *Proc. of the 18th International Symposium on Computer Architecture and High Performance Computing, SBAC-PAD '06*, pages 167–176, 2006.

[OC07] M. Oaksford and N. Chater. *Bayesian Rationality: The Probabilistic Approach to Human Reasoning.* Oxford University Press, 2007.

[Paw04] Z. Pawlak. Decision networks. In *Lecture Notes in Artificial Intelligence*, volume 3066, RSCTC 2004, pages 1–7. Springer-Verlag, Berlin Heidelberg, 2004.

[PB00] R. Potharst and J. C. Bioch. Decision tree for ordinal classification. *Intelligent Data Analysis*, 4:97–112, 2000.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, San Mateo, CA, 1988.

[Pea00] J. Pearl. *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2000.

[Pen04] Y. Peng. Intelligent condition monitoring using fuzzy inductive learning. *Journal of Intelligent Manufacturing*, 15:373–380, 2004.

[PH00] A. D. Preece and K. Y. Hui. The kraft architecture for knowledge fusion and transformation. *Knowledge Based Systems*, 13:113–120, 2000.

[Poh82] G. Pohlmann. Rechnerintegrierte Objektdarstellung als Basis integrierter CAD-Systeme. *Produktionstechnik Berlin*, 27, 1982.

[PP02] A. Papoulis and S. Pillai. *Probability, Random Variables and Stochastic Processes.* McGraw Hill, Netherlands, 4th edition, 2002.

[PR92] P. Perny and B. Roy. The use of fuzzy outranking relations in preference modeling. *Fuzzy Sets and Systems*, 49:33–53, 1992.

[Pra97] M. Pradhan. *Focussing attention in anytime decision making.* PhD thesis, Section on Medical Informatics, Stanford University, USA, 1997.

[Pre02] G. Presser. *Lazy Decision Making - Entscheiden durch zielgerichtetes Präzisieren der Wahrscheinlichkeitsinformation.* PhD thesis, Fachbereich Informatik, Technische Universität Dortmund, 2002.

[PW02] S. Parsons and M. Wooldridge. Game theory and decision theory in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 5:243–254, 2002.

[PW05] D. M. Pennock and M. P. Wellman. Graphical models for groups: belief aggregation and risk sharing. *Decision Analysis*, 2(3):148–164, 2005.

[PZ06] M. Pantic and R. Zwitserloot. Active learning of introductory machine learning. *Frontiers in Education Conference, 36th Annual, October 2006*, pages 1–6, 2006.

[Qiu06] R. G. Qiu. A service-oriented integration framework for semiconductor manufacturing systems. *International Journal of Manufacturing Technology and Management*, 10(2–3):177–191, 2006.

[QSS02] J. Qu, A. Shih, and R. Scattergood. Development of the cylindrical wire discharge machining process. *Transactions of the ASME, Concept, Design, and Material Removal Rate, Part 1*, 124, 2002.

[Rai68] H. Raiffa. *Decision Analysis: Introductory Lectures of Choices under Uncertainty*. Addison WesleyPress, Reading, MA, 1968.

[Rao07] R. V. Rao. *Decision Making in the Manufacturing Environment. Using Graph Theory and Fuzzy multiple Attribute Decision Making Methods*. Springer Series in Advanced Manufacturing, Springer-Verlag, London, 8 2007.

[Rat03] B. Rattay. Untersuchung der Einflussgrößen auf die Formfüllung und die Werkzeugbelastungen beim Prägen von Mikrokanalstrukturen in metallischen Blechen. Technical report, Schriftreihe Produktionstechnik der Universität des Saarlandes, Saarbrücken, 2003.

[Reh06] A. J. Rehm. *UML for Developing Knowledge Management Systems*. Auerbach Publications, Taylor & Francis Group, Boca Raton New York, USA, 2006.

[RES+00] J. Raimann, E. Enkel, A. Seufert, G. von Krogh, and A. Back. Supporting business processes through knowledge management. A technology-based analysis. Technical report, University of St. Gallen, Switzerland, 2000.

[Rie00] D. Riecken. Personalized views of personalization. *Communications of the ACM*, 43(8):26–29, 2000.

[RJO05] A. G. Rehorn, J. Jiang, and P. E. Orban. State-of-the-art methods and results in tool condition monitoring: a review. *International Journal of Advanced Manufacturing Technology*, 26:693–710, 2005.

[RLNR06] D. Raheja, J. Llinas, R. Nagi, and C. Romanowski. Data fusion - data mining-based architecture for condition-based maintenance. *International Journal of Production Research*, 44(15):2869–2887, 2006.

[RN03] S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, New Jersey, 2nd edition, 2003.

[Ros00] S. M. Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. Elsevier Academic Press, Netherlands, 2nd edition, 2000.

[Rq03] L. Ru-qian. *Knowledge Science and Computation Science*. Tsighua University Press, Beijing, China, 2003.

[San96] T. W. Sandholm. *Negotiation among self-interested computationally limited agents*. PhD thesis, University of Massachusetts Amherst, Department of Computer Science, United States, 1996.

[SBM+05] A.-W. Scheer, M. Boczanski, M. Muth, W.-G. Schmitz, and U. Segelbacher. *Prozessorientiertes Product Lifecycle Management*. Springer-Verlag, Berlin Heidelberg New York, 1st edition, 2005.

[Sch99] J. Schöttner. *Produktdatenmanagement in der Fertigungsindustrie - Prinzip, Konzepte, Strategien*. Fachbuchverlag Leipzig, 1999.

[Sch00] S. Schreiber. *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, 2000.

[Sch07] S. Schulte. *Integration von Kundenfeedback in die Produktentwicklung zur Optimierung der Kundenzufriedenheit*. PhD thesis, Lehrstuhl für Maschinenbauinformatik, Ruhr-Universität Bochum, 2007.

[SGGB07] J. Z. Shi, F. Gu, P. Goulding, and A. Ball. Integration of multiple platforms for real-time remote model-based condition monitoring. *Journal Computers in Industry*, 58(6):531–538, 2007.

[Sha76] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.

[SHRW05] J. Sun, G. S. Hong, M. Rahman, and Y. S. Wong. Improved performance evaluation of tool condition identification by manufacturing loss consideration. *International Journal of Production Research*, 43(6):1185–1204, 2005.

[Sme96] P. Smets. Imperfect information: imprecision and uncertainty. In A. Motro and P. Smets, editors, *Uncertainty Management in Information Systems: From Needs to Solution, Piracicaba, Brazil*. Kluwer Academic Publishers, Boston, USA, 1996.

[SMMSPY95] C. Shyi-Ming, Y. Ming-Shiow, and H. Pei-Yung. A comparison of similarity measures of fuzzy values. *Fuzzy Sets and Systems*, 72:79–89, 1995.

[Sor00] G. Sorger. *Entscheidungstheorie bei Unsicherheit*. Lucius & Lucius, Stuttgart, 2000.

[SR08] SYSTEM-3R. System 3R Basic Hardware WEDM. *Homepage http://www.system3r.com/*, 2008.

[ST07] D. M. Shalev and J. Tiran. Condition-based fault tree analysis (cbfta): a new method for improved fault tree analysis (fta), reliability and safety calculations. *Journal Reliability Engineering & System Safety*, 92(9):1231–1241, 2007.

[Sug77] M. Sugeno. Fuzzy measures and fuzzy integrals- a survey. In *Fuzzy Automata and Decision Processes*, pages 89–102. Elsevier North-Holland, Amsterdam, 1977.

[SWR06] R. Silva, S. Wilcox, and R. Reuben. Development of a system for monitoring tool wear using artificial intelligence techniques. In *Proc. of the I Mech E Part B Journal of Engineering Manufacture*, volume 220, number 8, pages 1333–1346, 2006.

[SY07] D. Shi and J. You. Adaptive dynamic probabilistic networks for distributed uncertainty processing. *Journal of Experimental & Theoretical Artificial Intelligence*, 19(4):269–284, 2007.

[Thi05] T. Thiadens. *Manage IT! Organizing IT Demand and IT Supply*. Springer Publish, Dordrecht, Netherlands, 2005.

[TK04] H. A. Thompson and G. G. Kulikov. *Dynamic Modelling of Gas Turbines: Identification, Simulation, Condition Monitoring and Optimal Control*. Advances in Industrial Control, Springer-Verlag, Berlin Heidelberg, 1st edition, 2004.

[TL06] K. M. Tay and C. P. Lim. Fuzzy FMEA with a guided rules reduction system for prioritization of failures. *International Journal of Quality & Reliability Management*, 23(8):1047–1066, 2006.

[Tor03] V. Torra. *Information Fusion in Data Mining*. Studies in Fuzziness and Soft Computing, Springer-Verlag, Berlin Heidelberg New York, 2003.

[TYJL06] A. Tsang, W. K. Yeung, A. Jardine, and B. Leung. Data management for condition-based monitoring optimization. *Journal of Quality in Maintenance Engineering*, 12(1):37–51, 2006.

[Uch08] G. Uchyigit. *Personalization Techniques and Recommender Systems*, volume 70. World Scientific Publishing, Series in Machine Perception & Artificial Intelligence, Vol. 70, 2008.

[UDP01] E. Uhlmann, U. Doll, and S. Piltz. Anwendung funkenerosiver Verfahrensvarianten für die Herstellung feinst- und mikrostrukturierter Formwerkzeuge. Technical report, EDM-Funkey, 01 2001.

[UP05] E. Uhlmann and S. Piltz. Manufacturing of micro structured rotational devices by electro discharge technologies. In *Proc. of the 5th Euspen Int. Conference, May 2005, Montpellier, France*, 2005.

[Vau97] E. J. Vaughan. *Risk Management*. John Wiley & Sons, New York, USA, 1st edition, 1997.

[vdGR01] L. C. van der Gaag and S. Renooij. On the evaluation of probabilistic networks. In *Proceedings of the 8th Conference on Artificial Intelligence in Medicine, Lecture Notes in Computer Science*, volume 2101, pages 457–461, 2001.

[VH05] V. Vidqvist and J. Halme. Data communication for additional service in industrial and mobile machine networks. Technical report, Espoo 2005, Technical Research Centre of Finland, VTT Research Report, 2005.

[VH06] R. Viertl and D. Hareter. *Beschreibung und Analyse unscharfer Information. Statistische Methoden für unscharfe Daten*. Springer-Verlag, Wien New York, 1st edition, 2006.

[Voo96] F. Voorbraak. Reasoning with uncertainty in artificial intelligence. In *Proc. Reasoning with Uncertainty in Robotics, RUR 1995*, volume 1093 of *Lecture Notes in Computer Science*, pages 52–90. Springer-Verlag, Berlin Heidelberg, 1996.

[VR93] J. E. Vargas and S. Raj. Developing maintainable expert systems using case-based reasoning. *Expert Systems*, 10(4):219–225, 1993.

[Wan03] K. Wang. *Intelligent Condition Monitoring and Diagnosis Systems. A Computational Intelligence Approach*. Subseries Knowledge-Based Intelligent Engineering Systems, IOS Press, Amsterdam Berlin Oxford Tokyo Washington, 2003.

[Wan06] W. Wang. Modelling the probability assessment of system state prognosis using available condition monitoring information. *IMA Journal of Management Mathematics*, 17(3):225–233, 2006.

[Wan07] W. Wang. A two-stage prognosis model in condition based maintenance. *European Journal of Operational Research*, 182(3):1177–1187, 2007.

[WB05] M. Weck and C. Brecher. *Werkzeugmaschinen - Maschinenarten und Anwendungsbereiche*. Springer-Verlag, Berlin Heidelberg New York, 6th edition, 2005.

[WB07] M. Weck and C. Brecher. *Dubbel. Taschenbuch für den Maschinenbau*. Springer-Verlag, Berlin Heidelberg New York, 21st edition, 2007.

[WC03] V. Weerakkody and W. Currie. Integrating business process reengineering with information systems development. In *Proc. of the Business Process Management Conference, BPM 2003, Eindhoven, Netherlands*, 2003.

[WCC06] M. C. Wu, W. J. Chang, and C. W. Chiou. Product-mix decision in a mixed-yield wafer fabrication scenario. *International Journal of Advanced Manufacturing Technology*, 29(7–8):746–752, 2006.

[Wei00] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, Massachusetts, 2000.

[Wei02] G. Weidl. *Root Cause Analysis and Decision Support on Process Operation*. PhD thesis, Department of Public Technology, Mälardalen University, Sweden, 2002.

[Wei06] F. Weissbuch. *Entwicklung eines Überwachungssystems für Strangguss-Kokillen*. PhD thesis, Universität Duisburg-Essen, Fakultät für Ingenieurwissenschaften, Abteilung Maschinenbau, 2006.

[WF05] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*, volume 2nd Edition. Morgan Kaufmann, San Francisco, 2005.

[WH93] M. P. Wellman and M. Henrion. Explaining 'explaining-away'. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):287–292, 1993.

[Wit02] F. Wittig. *Maschinelles Lernen Bayes'scher Netze für benutzeradaptive Systeme*. PhD thesis, Fachbereich Informatik, Universität des Saarlandes, 2002.

[WLRH96] Z. Wang, S. Lawrenz, R. Rao, and T. Hope. Feature-filtered fuzzy clustering for condition monitoring of tool wear. *Journal of Intelligent Manufacturing*, 7:13–22, 1996.

[WMD03] G. Weidl, A. L. Madsen, and E. Dahlquist. Object oriented bayesian networks for industrial process operation. In *Proc. of the Bayesian Modeling Applications Workshop associated with the 19th Conference on Uncertainties in Artificial Intelligence, Acapulco, Mexico*, 2003.

[Wol05] R. Wolff. Praxiserfahrungen mit Condition Monitoring Systemen zur zustandsorientierten Instandhaltung von On- und Offshore Windparks. Technical report, Symposium Husumwind, 9 2005.

[WP06] H. Wang and H. Pham. *Reliability and Optimal Maintenance*. Springer Series in Reliability Engineering, Springer-Verlag, Berlin, 2006.

[WS06] D. A. Wooff and J. M. Schneider. A bayesian belief network for quality assessment: application to employment officer support. *Journal of Intellectual Disability Research*, 50(2):109–126, 2006.

[WTL07] W. Wang, P. Tse, and J. Lee. Remote machine maintenance system through internet and mobile communication. *International Journal of Advanced Manufacturing Technology*, 31(7–8):783–789, 2007.

[XYZ97] L. Xiaoli, Y. Yingxue, and Y. Zhejun. Online tool condition monitoring system with wavelet fuzzy neural networks. *Journal of Intelligent Manufacturing*, 8:271–276, 1997.

[Yao00] Y. Y. Yao. Granular computing: basic issues and possible solutions. In *Proceedings of the 5th Joint Conference on Information Sciences, Atlantic City, New Jersey*, 2000.

[YE06] O. Yumac and H. M. Ertunc. Tool wear condition monitoring in drilling processes using fuzzy logic. In *ICONIP 2006, Part III*, volume 4234 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Heidelberg, pages 508-517, 2006.

[YK03] X. Yu and J. Kacprzyk. *Applied Decision Support with Soft Computing*, volume 124. Studies in Fuzziness and Soft Computing, Springer-Verlag, Berlin Heidelberg, 1st edition, 2003.

[Yos01] Y. Yoshida. *Dynamical Aspects in Fuzzy Decision Making*, volume 73. Studies in Fuzziness and Soft Computing, Springer-Verlag, Berlin Heidelberg New York, 2001.

[Zad65] L. A. Zadeh. Inexact information systems and its application to approximate reasoning. *Journal of Universal Computer*, 8:338–353, 1965.

[Zad78] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

[ZL05] X. Zhao and C. Liu. An organisational perspective on collaborative business processes. In *3rd Int. Conference on Business Process Management*, volume 3649 of *Lecture Notes in Computer Science*, pages 17–31. Springer-Verlag, Berlin Heidelberg New York, 2005.

Index