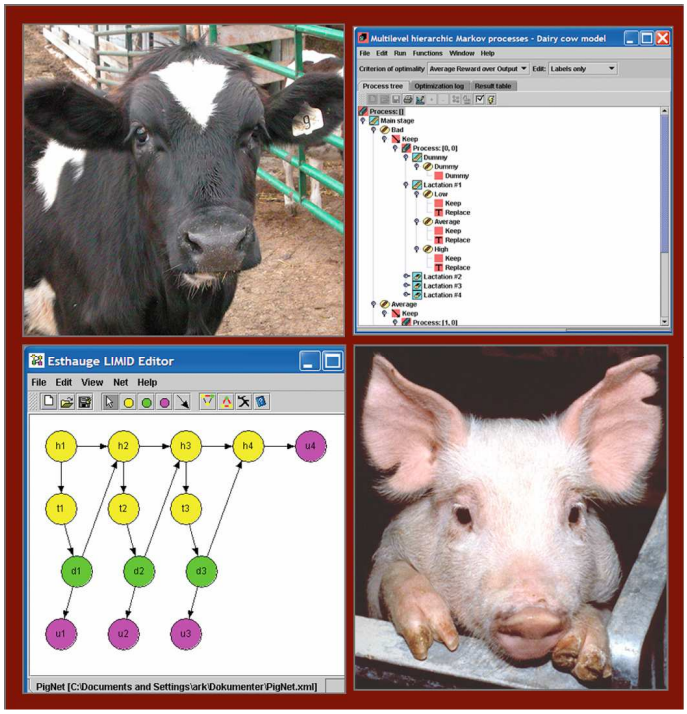
Herd Management Science



II. Advanced topics

2010 Edition

Compiled on October 31, 2013

Anders Ringgaard Kristensen
Erik Jørgensen
Nils Toft
University of Copenhagen
Faculty of Life Sciences

Preface

In 1996 two of the authors organized a Nordic PhD course on *Planning and Control of Animal Production at Farm Level*. The third author, Nils Toft, attended the course as a PhD student. As part of the material for the course, 5 textbook notes were written and published as Dina Notes.

The notes were successfully used at the course in 1996, and afterwards they have served as important input to the Master level course on *Advanced Herd Management* given at the Royal Veterinary and Agricultural University. A couple of the notes have been slightly updated over the years, but basically the content has not changed.

Over the years, the authors have had the wish to collect and update the material in order to create an authoritative textbook covering more or less all aspects of herd management from the basic to the advanced level. This preliminary edition is the first attempt to realize this wish. Even though there is still a long way to go before the work is done, it is our hope that the preliminary version will turn out to be useful for the participants of various Herd Management courses.

The book is organized in two parts with basic herd management principles and classical theories in part I, and the more advanced methods in part II. Furthermore, the necessary mathematical and statistical theory is summarized in appendices for easy reference.

Part I has been written with a bachelor level course in mind, and the contents reflect what we think that any animal scientist should master, whereas Part II has been written for a more advanced level for graduate students who wish to specialize in Herd Management. Given the structural development in modern agriculture with ever increasing herd sizes, we expect that the need for an advanced textbook focusing directly on herd management will increase.

Compared to previous edition, the 2010 edition has been extended with a new section on applications of linear programming in Chapter 10 and detailed computations in Example D.1. Finally, both volumes have been provided with comprehensive indexes for easy reference. Minor changes include correction of typos and correction of misleading errors in Example 5.8.

Copenhagen, October 31, 2013

Anders Ringgaard Kristensen
Erik Jørgensen
Nils Toft

This book has a home page at URL: http://www.dina.dk/∼ark/book.htm

Contents

Part II

Advanced topics

Chapter 9

Decisions and strategies: A survey of methods

9.1 Framework of the survey

In Chapter 1 we defined (Definition 1.2) the information needs for decision making to include

1. The current state of the unit.
2. A production function describing the immediate production given stage, state and decision and the distribution of the possible random term(s).
3. The distribution of the future state given stage, state and decision.
4. All attribute functions relevant to the farmer's preferences.
5. The utility function representing farmer's preferences.
6. All constraints of legal, economic, physical or personal kind.

The purpose of this chapter is to provide a general overview of techniques available in the determination of optimal decisions and strategies.

The techniques will be described according to their ability to represent the various kinds of necessary information listed above. Furthermore, their potentials for integration of decisions at different levels and time horizons defined in Chapter 1 are discussed. It is not the purpose to describe the various methods in details, but only to provide a general survey relating to the issues of the previous chapters. Details about the methods are left for separate chapters.

9.2 Rule based expert systems

Research concerning expert systems is a development within the area called Artificial Intelligence (AI). The British Computer Society has defined expert systems as

follows (cited from Dindorph, 1992): *An expert system is regarded as the embodiment within a computer of a knowledge-based component from an expert skill in such a form that the system can offer intelligent advice or take intelligent decision about a processing function. A desirable additional characteristic, which many would consider fundamental, is the capability of the system, on demand to justify its own line of reasoning in a manner directly intelligible to the enquirer. The style adopted to attain these characteristics is rule-based programming.* This is just one of many proposed definitions of expert systems.

The fundamental difference between the rule based systems, and the approach that we have presented until now, is that instead of trying to model a system, the rule based expert systems tries to model the expert, or rather the expert's approach to problem solving. Originally, the ambition within AI-research was to make general problem solvers that could be used for any problem area, but this was realized by most researchers within the area to be overambitious. The research efforts had, however, led to new approaches towards problem solving, and within narrow problem (expert) domains, the approach showed some promising results.

Rule based expert systems have three components: a knowledge base containing the expert's knowledge of the domain, an inference engine that decides how and when to use the knowledge, and a user interface.

The knowledge base contains knowledge of a problem domain, as it is described in text books, as well as expert knowledge, e.g. exceptions to general rules, experiences from previous problems, and time-efficient approaches on how to solve problems within the area. The knowledge base is an enhanced data base that apart from data also contains logic rules for the connection between the items in the knowledge base, e.g if x or y then z.

The inference machine contains the mechanism for deduction based on the logical rules in the knowledge base. The deduction can use different inference principles, such as backward chaining and forward chaining. In the rule showed above, the inference machine would start out by finding the value of z, and given the knowledge of z establish the value of x and y. Forward chaining would start out by establishing the values of x and y and deduce the value of z subsequently. In both cases the unknown values are found either by a question to the user or by combining other rules in the knowledge-base. The optimal choice of inference principle depends on the type of problem the expert system is supposed to solve. In very complex expert systems, neither forward nor backward chaining is fast enough, and the so-called heuristic search strategies are needed. These strategies work primarily by searching the knowledge base in an efficient order, focusing on areas, where a solution to the problem is most likely to be found. Both general and problem specific heuristic strategies exist.

The questioning mechanism is a standard part of the user interface. Besides posing questions, the user interface is usually able to explain, why it asks the question, i.e. *I am trying to establish the value of z because I want to know if either x or y is true*. Another facility is the explanation facility, i.e. *I know that neither x*

is true nor y is true because z is not true. Usually the phrases are formulated more user friendly.

Programs for maintaining knowledge bases in connection with inference machine and user interface as an integral part is sold as the so-called expert system shells. This concept originates from the medical diagnosis system, MYCIN. The knowledge base in MYCIN was emptied and the program sold as E-MYCIN (or empty mycin), and was thought to be applicable to any problem domain. Very often these shells are programmed in programming languages where logic deduction can easily be represented, such as LISP or PROLOG, but standard programming languages such as C and Pascal can of course be used.

Rule based expert system can be categorized into several areas (Hayes-Roth et al., 1996). Referring to Figure 1.1 they comprise the planning, check and analysis phase of the management cycle.

In developing rule based expert systems two "players" are essential, of course the expert, but in addition the so-called knowledge engineer. The role of the knowledge engineer is to "extract" the knowledge from the expert and to formulate the knowledge as rules that can serve as input to the knowledge base. Knowledge engineering has in fact become a research area in its own right.

To illustrate the problem of knowledge engineering, the first rule based expert systems were based on very simple "expert" rules very much inspired by the diagnostic systems, e.g. if indications a and b are observed then problem is probably c. Later on it was realized that the expert relies on many information sources and part of being an expert is to know when to draw on which knowledge sources. If we look at the information necessary to make optimal decisions as mentioned in Section 9.1 this can be seen as the result of an expert's problem solving. An expert system would therefore guide the user through obtaining the necessary information. If it is not possible to obtain the necessary information it would use other information and based on the expert's experience try to make a sufficiently good plan.

The current trend is that the rule-based system does not function as stand-alone systems, but rather as an integral part of other systems, the so-called knowledge based systems. A typical example could be that the expert system helps in establishing the user's utility function by asking questions and then uses this utility function when calling an optimizing program. The concept is now incorporated into the wizards and experts known from standard computer program, e.g. spreadsheets and word processors.

The method will not be discussed further in this book.

9.3 Linear programming with extensions

The general linear programming problem may in matrix notation be written as follows:

$$\begin{aligned} px &= \text{Min!} \\ &subject\ to \\ Ax &\leq b \\ x &\geq 0 \end{aligned} \tag{9.1}$$

where p is a constant row vector with m elements, A is a constant matrix of size $n \times m$, b is a column vector with n elements, and x is a vector of variables. The problem is to select a vector x that minimizes the linear objective function px and simultaneously meets the linear constraints $Ax \leq b$ and $x \geq 0$.

Eq. (9.1) represents the standard formulation of a linear programming problem. In applied models it is often convenient to define a maximization problem instead, and some of the restraints may be of the kind $a_i x \geq b_i$ or the kind $a_i x = b_i$ (where a_i is a row in A) , but any linear programming problem may be rearranged in accordance with the standard formulation of Eq. (9.1).

If we interpret the linear programming problem in relation to a herd management decision problem, then x is a vector of factor levels and $Ax \leq b$ is a set of constraints of legal, economic, physical or personal kind. It should particularly be noticed, that personal constraints may also include constraints on levels of attribute functions (for instance leisure time or monetary gain). This direct representation of constraints is probably the main force of the method. The objective function px has to represent the aggregate utility function.

If we compare the linear programming problem with the information needs of a decision problem (cf. Section 9.1) we observe that all random elements are missing. At least in the standard formulation, the method is deterministic. Also the dynamic linking to the future state of the production unit is missing. A consequence of the latter shortcoming is that only effects at the current stage are represented. In other words, the method is static of nature. Furthermore, we observe, that since the aggregate utility function has to be linear in the factor levels, x, also the production function, all attribute functions and the utility function have to be linear. Examples of linear attribute functions are shown as Eqs. (3.1), (3.4) and (3.7), and a linear utility function is shown as Eq. (3.17). This demand for linear functions and linear restraints is a serious weakness of the method.

Several of the shortcomings mentioned may be redressed or at least adapted by extensions to the method: The linear objective function may be replaced by a quadratic one (quadratic programming); the static nature may be modified by introduction of stages and additional constraints ensuring dynamic links (dynamic linear programming); random terms may be added to the elements of A and b, and the corresponding restraints may be expressed as probabilities (stochastic programming); and often, non-linear functions may be approximated by pieces of linear relations over short intervals. In particular, dynamic linear programming, may be used to link decisions at different levels with different time horizons.

Herd management applications of linear programming are numerous. The most frequent application is no doubt for ration formulation, where least-cost rations meeting the nutritional requirements of the animals in question are met. Most often such programs ignore the effect of feeding on production.

Also examples of application of linear programming for whole-farm planning may be found in literature. Refer for instance to Hansen (1992) and Hardie (1996). Such models are often very large containing thousands of variables and restraints, but since very efficient standard software is available this is hardly a problem.

Due to the shortcomings of Linear Programming in dealing with important aspects (like dynamics and uncertainty) of herd management, the method is not given high priority in this book. Nevertheless, a short intuition based description is given in Chapter 10, together with practical aspects of modeling by use of linear programming.

9.4 Dynamic programming and Markov decision processes

Consider a production unit which is observed over a number of stages $n = 1, \ldots, N$. At the beginning of each stage, we observe the state, $i \in \omega_n$, of the unit. Having observed the state, we have to take an action, $d \in D_n$, concerning the production unit. Usually, the state space, ω_n, and the action space, D_n, are assumed to be finite sets. Depending on the stage, state and action, a reward is gained. The reward may very well be a random variable, but the expected value, $r_i^d(n)$, has to be known. Also the state to be observed at the next stage is a random variable. We shall denote as $p_{ij}^d(n)$ the conditional probability of observing state j at stage $n+1$ given that state i has been observed and action d taken at stage n. Finally, a strategy, s, is defined as a map assigning to each combination of stage and state an action $s(n,i) \in D_n$. We have now defined the elements of a Markov decision process (or a dynamic programming problem).

The purpose of dynamic programming is to determine a strategy which (in some sense) is optimal. Several optimization techniques are available. The most commonly applied method is called *value iteration* where a value function representing the expected total rewards from the present stage until the end of the planning horizon (i.e. stage N) is maximized. Optimal decisions depending on stage and state are determined backwards step by step as those maximizing the value function. This way of determining an optimal policy is based on the Bellman principle of optimality which says: *An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision* (Bellman, 1957, p. 83). Value iteration is often just denoted as dynamic programming.

If N is large, an infinite planning horizon is often assumed. A relevant optimization technique for infinite stage problems is *policy iteration*. This method was introduced by Howard (1960), who combined the dynamic programming technique with the mathematically well established notion of a Markov chain. A natural con-

sequence of the combination was to use the term Markov decision process to describe the notion. The policy iteration method was a result of the application of the Markov chain environment and it was an important contribution to the development of optimization techniques.

The objective function being maximized during optimization depends on the circumstances. It may represent the total expected rewards, the total expected discounted rewards, the average rewards per stage or the average rewards over some kind of physical output.

If we compare the dynamic programming problem with the information needs of a decision problem (cf. Section 9.1) we observe that most aspects are covered. The current state is simply i, and the rewards directly correspond to production functions; the conditional probabilities $p_{ij}^{d}(n)$ represent the dynamic random links to future stages; and the objective function represents the farmer's utility function. There are, however, some restrictions on the kind of utility function which can be represented in a dynamic programming model. The restrictions concern the way in which individual stage attributes are aggregated into the over-all utility function as described by Eq. (1.5). In order to be able to use dynamic programming, we implicitly assume that the aggregation may be performed in such a way that we first aggregate attributes at the same stage into a stage specific utility v_n of the kind

$$v_n = g_n(u_{1,t_n}, ..., u_{k,t_n}),$$

where g_n is a stage specific utility function of arbitrary kind. In the dynamic programming context, v_n is identical to the reward $r_i^d(n)$. The over-all utility (i.e. aggregation over stages) in turn must be calculated as a simple sum of the stage specific utilities v_n, as a discounted sum (cf. Eq. (3.2)), as the average value over stages or as the average value over some kind of physical output or input from production.

The most difficult kind of information to represent in dynamic programming models is the information on constraints. There is no general solution to that problem, but some times it may be solved by using an objective function maximizing average rewards relative to the most limiting restriction. An example is maximization of average net returns per unit of milk produced under a milk quota (Kristensen, 1989). In other cases combination of the method with methods like simulation (Ben-Ari and Gal, 1986; Kristensen, 1992) or genetic algorithms (Houben et al., 1995) may circumvent the constraint problem.

A major problem in relation to dynamic programming models is the so-called curse of dimensionality. Since the state space is represented by discrete levels of a set of traits (state variables), models tend to become very big. Thus a model presented by Houben et al. (1994) contained 6.8 million states. Despite the size of the model, optimization was still possible due to a new notion of a hierarchical Markov process described by Kristensen (1987, 1991).

Later the concept was further developed by Kristensen and Jørgensen (2000) into multi-level hierarchical Markov processes in order to allow for simultaneous

optimization of decisions on multiple time scales (at the founder level as well as at the child levels). A general Java software system, MLHMP, for representation and solution of multi-level hierarchical Markov processes has been developed by Kristensen (2003a).

Numerous applications of dynamic programming are described in literature. A relevant textbook concerning application in agriculture has been written by Kennedy (1986). The book also contains a survey of agricultural applications. In herd management, the technique has most often been applied for operational decisions like replacement, insemination and medical treatment of animals.

Despite the numerous successful applications, the very concept of a Markov decision process has some built-in limitations:

State space representation: Most often the state of the system (e.g. an animal) being modeled is defined by the values of a number of state variables each representing a trait of the system. The state space is then defined as the cartesian product of all value sets of individual state variables. This leads to very large transition matrices which - even though they are often sparse - are inefficient from a numerical point of view.

Observability: It is generally assumed that the parameters of a Markov decision process are known and that the state space is fully observable. In particular when we are dealing with production assets, it would be more logical to distinguish between directly observable state variables like for instance number of items produced and underlying unobservable asset dependent potential for production capacity. This kind of modeling is referred to as POMDP *Partially Observable Markov Decision Processes*. Refer for instance to Lovejoy (1991) for a survey or Kaelbling et al. (1998) for an introduction.

Markov property: The Markov property implies that the state space at any stage must contain sufficient information for determination of the transition probabilities. In a straight forward formulation of a decision problem this is rarely the case. The trick used most often in order to make the process Markovian is to include *memory variables* in the state space. Even though this solves the problem from a theoretical point of view it contributes significantly to the *curse of dimensionality* of such models.

Even though the multi-level hierarchical Markov decision processes to some extent compensate for these problems by partitioning the state space according to temporal considerations and also an attempt to combine hierarchic Markov processes with Bayesian updating in special cases has been done by Kristensen (1993), the state space representation, the lacking observability and the Markov property remain important limitations for the use of Markov decision processes.

Further details of the technique are given in Chapter 13. A very good general textbook has been published by Puterman (1994).

9.5 Probabilistic Expert systems: Bayesian networks

Another part of the research area named Artificial Intelligence are the so called probabilistic expert systems that rely on the Bayesian network. The following description is based on Lauritzen (1995).

In some areas where expert systems are appropriate, the task involves performing a sequence of steps according to specified logical rules. However, other expert systems work in domains that are characterized by inherent uncertainty. This uncertainty is either due to imperfect understanding of the domain, incomplete knowledge of the state of the domain at the time where the task has to be performed, randomness in the mechanisms governing the behavior of the domain or a combination of these. Within these domains probability and statistics can serve to represent and manipulate the uncertain aspect of domains having these characteristics. Probabilistic methods were for some time discarded in this context as requiring too complex specification and computation. However, the work of Pearl (1988) and Lauritzen and Spiegelhalter (1988) demonstrated that these difficulties could be overcome, based on causal networks or as it is now usually termed Bayesian networks. There exist other formalisms for handling uncertainty in expert system, such as the fuzzy sets, but these will not be discussed in the present context.

The rule based systems were mainly constructed through modeling of the behavior of the expert and the encoding of this behavior in terms of rules of various kind. In contrast, probabilistic expert systems are constructed by modeling the domain rather than the expert. The method is thus in close correspondence with the approach used in this book, where the domain is modeled using production functions etc. The probabilistic expert systems specify a graphical model for the variables. The reasoning is then performed by updating the probabilities of the domain in the light of the specific knowledge according to the laws of conditional probability.

The graphical model captures associations between entities in the domain, or rather lack thereof, in terms of conditional independence that in a systematic fashion are encoded in a graph or network with nodes representing the entities themselves and edges representing associations between them. The nodes are represented as dots or circles. The edges are usually directed corresponding to influences of a causal nature and represented as arrows, or, sometimes, undirected corresponding to symmetric associations (e.g. correlations) and represented as lines.

The use of the graphic specification in the probabilistic expert systems plays several roles. For example, it gives a visual picture of the domain information; it gives a concise presentation of domain information in terms of conditional independence relations, and it enables rapid computation and revision of interesting probabilities.

The graphic method can also be used for several important tasks in the specification process. It can be used to learn quantitative and structural aspects, or as it known within general statistics, estimation and model selection.

If we compare the probabilistic expert systems method with the information needs of the decision problem (Section 9.1), it is important to recognize that the method is inherently a static method, even though attempts have been made to model dynamic systems as well. For monitoring and analysis purposes it is ideal, i.e. the method can assign probabilities to observed deviations, whether they are random or not. It can also make a diagnosis in the analysis, that is, indicate probabilities for different causes of the deviation. This can in turn serve as the necessary basis for decisions concerning changes in production plan.

The expert systems can be build as recurrent time slices and can in this manner represent dynamic production functions, predicting the future state for given decisions. The constraints concerning the production function can be modeled, but constraints may cause the same problems as described under Dynamic Programming and Markov decision processes.

To represent decisions in Bayesian networks the decision can be included as a random variable in the model, with the different decisions as level of the variables. When the decision is made, the corresponding level is assigned a probability of 1. This approach does not make any search for optimal decisions.

If decisions have to be included, Decision Graphs should be used instead. They can in fact be fitted into the general framework of Bayesian Networks.

We shall not in this book discuss Bayesian networks further. Instead, readers are referred to an excellent textbook by Jensen (2001) where this important subject is convincingly introduced in Chapters 1 and 2. Readers interested in the algorithmic aspects of the method should consult Cowell et al. (1999).

9.6 Decision graphs

Decision graphs (or influence diagrams as they are also called) were introduced by Howard and Matheson (1981) as a formalism to model decision problems with uncertainty for a single decision maker. The influence diagrams gave a more compact graphical representation of a decision problem than the more traditional decision tree approach as illustrated in Figure 9.1.

A decision graph is very similar to a Bayesian network consisting of chance node and directed edges ("arrows") denoting causal effect. In addition two more node types are introduced, the decision node shown as a square, and the value node shown as a diamond (cf. Figure 9.1).

Originally, the decision graph was translated to a decision tree within the computer and the standard "average-out and fold-back" algorithm was applied on that tree. In Shachter (1986) a method was suggested for solving the decision problem represented by the decision graph directly, without the translation to a decision tree. This method transformed the decision graph by successively removing nodes in the graph, until at last only one final utility node remained, holding the utility of the optimal policy. In order to solve many similar problems one therefore had to start from scratch every time. The transactions performed on the graph consisted of

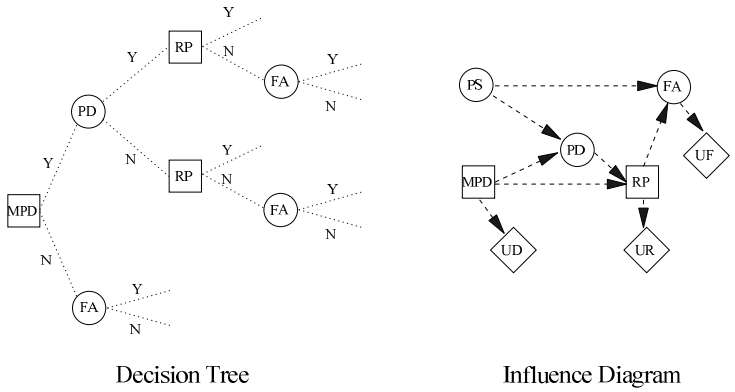


Figure 9.1: Decision tree and decision graph (influence diagram) representation of the pregnancy diagnosis and replacement problem.

four simple transactions, the arc reversal (application of Bayes Theorem), node removal by summing out, expectation of a value node with respect to a change node, and finally removal of a decision node into a value node by maximization. Initially problems were formulated with only one value node. By introducing the concept of separability of the utility function or value function, Tatman and Shachter (1990) showed that the Dynamic Programming Problems could be solved within the decision graph framework, by introducing the separability of the utility function. The requirement for separability is the same as the requirement that the overall utility is calculated as a simple sum of stage specific responses as mentioned in Section 9.4.

Shenoy (1992) proposed another algorithm that gave the solution to the decision graph without disrupting the structure of the graph. Then Jensen et al. (1994) showed how a similar approach could be incorporated within the general framework of Bayesian Networks. This approach has been implemented in the Hugin expert system shell. The similarity between decision graphs and Bayesian networks means, that several important improvements is to be expected. This comprises, e.g approximate solutions by techniques such as Monte Carlo methods (Bielza et al., 1999; Charnes and Shenoy, 1996), easy representation of dynamic models as in dHugin (Kjærulff, 1995) and object oriented design (Bangsø, 2004).

As decision graphs closely correspond to Dynamic Programming the same comments concerning the information needs of the decision problems can be made. In addition, the current version of decision graphs are inherently static, and no algorithm corresponding to policy iteration has been found (even though R.A. Howard's

research has been central for both developments). If stages of varying time length have to be modeled, time has to be included in the model and the discounting factor has to be incorporated directly in the utility. Furthermore decisions choosing between qualitatively different child processes, such as in Hierarchic Markov processes is currently not possible. It should, however, be noted that it is a very active research area, and continuous progress is to be expected.

Decision graphs at first glance seem to overcome all three problems listed for Markov decision processes in Section 9.4:

State space representation: Instead of a state space formed as a cartesian product, the state variables are represented one by one in decision graphs.

Observability: In decision graphs, variables may be observable or unobservable.

Markov property: The classical algorithms applied to decision graphs as presented by for instance Jensen et al. (1994), implicitly assume *no forgetting* implying that all previous observations done and decisions made are remembered and taken into account. This "no forgetting" assumption is today associated with the concept of "Influence diagrams" as a subclass of the more general concept of "Decision Graphs".

In practice, however, the performance of influence diagrams has been disappointing. Even though algorithms for optimization are available (refer for instance to Cowell et al., 1999, Chapter 8), the numerical calculations become prohibitive for real world decision problems because of the "no forgetting" assumption. A recent extension by Lauritzen and Nilsson (2001) to influence diagrams called LIMIDs or *LImited Memory Influence Diagrams* relaxes the "no forgetting" assumption thus providing a computationally tractable decision problem without assuming a Markov process. The algorithm presented is exact in some cases and only approximate in other cases.

Thus, the method has obvious possibilities for application within animal production, but so far the only example known to the author is a model for determination of optimal slaughter policies in slaughter pigs (Hansen, 2006). Within crop protection a system has been made for decision making concerning mildew management in winter wheat (Jensen, 1995).

We shall not in this book give details about the algorithms of decision graphs. Instead, we again direct the readers to good textbooks like Cowell et al. (1999); Jensen (2001). Chapter 12 deals with some practical aspects of modeling by use of decision graphs.

9.7 Simulation

As the name implies, a simulation model is simply a model of a system. The model is used for the study of the real system's behavior under different condi-

tions. Within animal production the term usually refers to computer based dynamic calculation models.

Formally, the simulation model is a computer representation of the production function, the attribute function, and/or the utility function. The degree of detail differs between the different models.

The input to the model consists of two elements, a set of parameters, Φ, and a set of decision rules, Θ. The decision rules specify the setting of input factors as well as other decisions in the system. The term "decision rule" is used rather than decision strategy, because usually no direct mapping between the rule and the state of the whole system exists. A decision rule can e.g be to use a dynamic programming model to specify a decision strategy every (simulated) year. Another example is to use some simple rule-of-thumb (heuristic) to make culling decisions. The set of parameters can be split in two, $\Phi = (\Phi_0, \Phi_{s\bullet})$, where Φ_0 are the initial values of the parameters at the start of the calculation (State of Nature) and $\Phi_{s\bullet}$ represents parameter values that change during simulations. The elements of $\Phi_{s\bullet}$ are often called state variables, and can be further split into time stages of the model i.e. $\Phi_{s\bullet} = (\Phi_{s1}, \Phi_{s2}, \ldots, \Phi_{st}, \ldots, \Phi_{sT})$, where T is the number of stages in the planning horizon. It is often convenient to refer to the set of output variables $\Omega_\bullet$ that contains calculated values of input factors, production functions, attributes etc. The distinction between the elements of $\Phi_{s\bullet}$ and $\Omega_\bullet$ is not clear, but usually, $\Omega_\bullet$ is a subset of $\Phi_{s\bullet}$. The elements in $\Omega_\bullet$ will usually be traits that at least in principle can be observed in the real system. The term *model input* usually refers to (Φ_0, Θ).

The purpose of the models is to calculate the expected utility, $\overline{U}(\Theta)$, under a given decision rule, Θ, i.e.,

$$\begin{aligned} \overline{U}(\Theta) &= \int_{-\infty}^{\infty} U(\Theta, \phi) f_\Phi(\phi) d\phi \qquad (9.2) \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} U(\Theta, \Phi_{s\bullet} \mid \Phi_0 = \phi_0) f_{(\Phi_{s\bullet}|\Phi_0)}(\phi_{s\bullet} \mid \phi_0) f_{\Phi_0}(\phi_0)\, d\phi_{s\bullet} d\phi_0, \end{aligned}$$

where U is the utility function, which in general can refer to any function of the output variables. The functions f_Φ, $f_{(\Phi_{s\bullet}|\Phi_0)}$ and f_{Φ_0} are the density functions of Φ, $(\Phi_{s\bullet}|\Phi_0)$ and Φ_0, respectively. *Simulation models are numerical methods for solving this integral.*

Two different categories of simulation models have been implemented within animal production. Stochastic models and deterministic models, where the stochastic nature of the system is ignored, i.e. the underlying assumption is that $\mathrm{P}(\Phi = \phi_c) = 1$ for some ϕ_c, e.g. estimated from various experiments. It is important to realize that such a simplification is only valid, if $U(\Theta, \Phi)$ *is linear in the parameters*. Since this is almost never the case, we will more or less ignore deterministic simulation models in this book. They may have some justification for system comprehension in animal physiology and nutrition (examples can be found in Whittemore and Fawcett, 1976; Black, 1995; Arnold and Bennet, 1991; Danfær, 1990), but hardly in herd management.

Stochastic models can be further subclassified into Probabilistic Models and Monte Carlo models. Probabilistic models are models such as Markov Chain models (see references under Dynamic Programming and in addition e.g. Jalvingh et al., 1992a,b) and Bayesian Networks. Within the probabilistic models the distribution of the output variables can be directly found within a single run of the model. Reasonable complex models can be specified within this context, at least if the parameters and the traits follow the Gaussian (normal) distribution. Capacity restrictions, interactions between system elements and the inclusion of decision variables will, however, make it impossible to specify the distribution in closed form.

Therefore, the Monte Carlo simulation technique is preferred. It relies on the drawing of random numbers. Every time the model encounters a stochastic variable, a (pseudo)-random variable is drawn from the appropriate distribution and this value is used in the subsequent calculations. Each completed calculation (simulation run) with the model represents a random drawing from the simultaneous distribution of input and output variables. By increasing the number of calculations the distribution of the output variables can be specified to any degree of precision. The expected utility is found from:

$$\overline{U}(\Theta) \approx \frac{1}{k} \sum_{i=1}^{k} U(\Theta, \phi_i). \tag{9.3}$$

where ϕ_i is a random drawing from the multidimensional distribution of the parameters, and k is the number of random drawings (simulation runs). In addition the standard error on the estimated utility can be found by calculating the variance of $U(\Theta, \phi_i)$. Thus, we can obtain a measure of the precision of the estimated utility, and an indication of how many iterations that are necessary. If, for instance, the standard error of the expected utility is 10% after 100 iterations, it will take 10,000 iterations to obtain a standard error of 1%.

Examples of Monte Carlo simulation models are (in pigs) Singh (1986); de Roo (1987), (in dairy cows) the *SimHerd* model (Sørensen et al., 1992; Østergaard et al., 2000, 2004, 2005) and the *Florida Dairy Computer Program* (de Vries et al., 2004), and (in hens) the *SimFlock* model (McAinsh and Kristensen, 2004; Kristensen and Pedersen, 2003).

Simulation models can also be divided between physiological models of single animals, physiological models of whole herds, and models of whole herds with emphasis on managements strategies. The physiological model of whole herds is e.g. Tess et al. (1983); Pettigrew et al. (1986); Finlayson et al. (1995), while examples of current physiological models is found under the deterministic models, and the whole herd approach under the stochastic models mentioned above. Obviously, it is within the last category that the likely candidates for decision support systems should be found. However, models from the first category have been adapted to serve as decision support systems. The first two approaches are often based on a description of the system with differential (or difference) equations, while the third

approach relies more on the theory behind stochastic processes such as queuing models.

Compared to the other techniques, simulation models are much more flexible, and there is no constraint on the degree of detail in the model. Especially when the so-called object oriented programming method is used, it is possible to achieve a very close correspondence between the elements of the real system and the model (see e.g. Chang et al., 1994; Skidmore and Beverly, 1995; Jørgensen and Kristensen, 1995; McAinsh and Kristensen, 2004). Any model variable can be used as output variable and it is easy to represent capacity restrictions.

Very often the purpose of simulation models is to improve the understanding of a system, i.e. to combine research results from different areas to obtain a comprehensive description of the system, the so-called holistic approach. This purpose should be seen as something different from decision support. When simulation models are used to improve the understanding of the complex system, a fixed and known set of parameters ϕ_0 are used for the initial state of nature, Φ_0, and the expected value of the utility function or any other output variable is calculated as:

$$\overline{U}(\Theta \mid \Phi_0 = \phi_0) = \frac{1}{k} \sum_{i=1}^{k} U(\Theta, \phi_{si} \mid \Phi_0 = \phi_0), \tag{9.4}$$

i.e. only the inner part of the integrand in (9.2) is solved.

The knowledge of the systems sensitivity to changes in the parameters is part of, what we call understanding of a system.

In contrast, when simulation models are used to determine "optimal" strategies we want to find the optimal set of decision rules given the precision in our current knowledge of the parameters (state of nature). The parameters used in each simulation run should therefore be a sample from the prior distribution of Φ_0 reflecting the precision in our current knowledge, and not fixed values.

The search for optimal strategies is included in linear programming, dynamic programming and decision graphs, (i.e. simplex algorithm, policy and value iteration). No such search facility is included in connection with simulation models. This is a major drawback of the method.

Within simulation models the search for the optimal set of decision rules is treated as a general problem of multidimensional optimization. Several numerical methods exist that can handle this (see e.g. Press et al., 1989). The choice of method should be made carefully. The flexible form of the simulation models means that the behavior of the expected utility function is unknown, for example if there exist discontinuities and local optima. Such phenomenons can make some of the methods go wrong. Another complication is that the expected utility is only estimated with a precision depending on the number of simulation runs within each treatment. The difference between two sets of decision rules may therefore be just a matter of sampling error, rather than a difference in expected utility. The solution to this problem is to do more simulation runs. But there is a trade off between time spent improving the precision in the estimate of one set of decisions rules and

the time spent searching for a better. Guide lines to handle this problem are not available.

The search procedures are most easily demonstrated by borrowing terms from experimental world. A set of decision rule is termed a treatment. Expected values from a given treatment are found by a number of replicates (N) or simulation runs. When searching for optimal decision rules, we have to repeatedly specify new treatments and calculate expected utility for the treatment. If we want to combine a set of treatments simultaneously, we design an experiment with the different treatments included.

A well-established technique for well-behaved expected utility functions, especially with continuous variables in the decision rules, is the gradient search technique. First an experiment is designed to initially explore the expected utility function, e.g. a response surface design. The result from this experiment is analyzed and the response surface estimated. If the optimum is outside the current design, the path of steepest ascent of the response surface is estimated. Then an experiment is made with treatments on the steepest ascent path, until the optimal treatment on this path is found. A new response surface design is made centered around this optimum point. This procedure is repeated until the optimum is found with sufficient precision. Using this procedure, an (at least local) optimum will be found.

Other promising techniques are stochastic search techniques, such as simulated annealing and genetic algorithms. These algorithms start with the selecting of a random initial set of decision rules (treatment) as the current. The expected utility of this is calculated. Then the following steps are carried out iteratively. Select a new treatment candidate based on the current treatment by random permutation. Calculate expected utility for the treatment candidate. Decide randomly according to a specific rule (depending on the technique), whether to use the treatment candidate as current candidate by drawing a random variable. Continue the iterations.

A third possibility is the group of so-called heuristic search strategies. Examples can be found in Reeves (1995).

If we compare the simulation method with the information needs of the decision problem, all the aspects can be covered, and the utility function and capacity restrictions can easily be handled. The curse of dimensionality is not felt immediately. The computation time of a single run of the model grows more or less linearly with the complexity of the model. The problem is the search for optimal solutions. The techniques mentioned are not as efficient as either the simplex, value iteration or policy iteration methods. With the same complexity in the decision rules as in e.g. dynamic programming the curse of dimensionality will be felt, e.g. if the decision to cull an animal should include the states of all other animals in the herd. If the rules are specified more heuristically, such as cull the worst animal, the problems become tractable, but no overall optimal solution is guarantied. Other decision rules might exist with higher utility.

Probably because of this problem, published results from simulation model research usually have only very few options in the decision rule, and the decision

rules are often of a very general nature. The use of simulation models for decision support is usually suggested to be of the what-if nature, i.e. the user of the model specifies some decision rules and the model calculates the expected output from these decision rules. This approach has advantages because there is no need to attempt to formulate the farmer's utility function. The user of the model can simply look at the list of output variables for different set of decision-rules and decide which set to prefer. Anyhow, it seems that some kind of optimality search within the simulation models would be the best.

The future developments within simulation modeling, will probably be in the area of estimating model parameters, perhaps by directly using the model calculations. More efficient strategies for sampling than the purely random approach, and improved search strategies are of interest, too. Finally, developments within the area of multi criteria optimization, to obtain a better reflection of the farmers utility in the object function should not be overlooked.

Chapter 12

Decision graphs: Potential use and current limitations

12.1 Introduction

The purpose of this chapter is to illustrate the potential use of the decision graph technique within herd management. Decision graphs were (under the name of *Influence Diagrams*) introduced by Howard and Matheson (1981) as a formalism to model decision problems with uncertainty for a single decision maker. The influence diagrams gave a more compact graphical representation of a decision problem than the more traditional decision tree approach. In Shachter (1986) a method was suggested for solving the decision problem represented by the influence diagram directly, without the translation to a decision tree. This method transformed the influence diagram by successively removing nodes in the graph, until at last only one final utility node, holding the utility of the optimal policy. In order to solve many similar problems one therefore had to start from scratch every time. Shenoy (1992) proposed another algorithm that gave the solution to the influence diagram without disrupting the structure of the diagram. Then Jensen et al. (1994) showed how a similar approach could be incorporated within the general framework of Bayesian Networks. This approach has been implemented in the Hugin expert system shell. We refer to Jensen et al. (1994) for a presentation of the method. In addition, we refer to the short introduction given in Section 9.6.

As concerns software for decision graphs, reference is made to the Hugin[1] system implementing classical influence diagrams under the "no forgetting" assumption by use of the strong junction tree approach (Jensen et al., 1994) and the Esthauge LIMID Software System[2] implementing Limited Memory Influence Diagrams as described by Lauritzen and Nilsson (2001).

[1] http://www.hugin.com
[2] http://www.esthauge.dk

12.2 From decision tree to influence diagram

The decision tree is a method for decision analysis. An excellent general description can be found in Hillier and Lieberman (1996). The concept is also illustrated in Example 4.5 of Chapter 4. In order to further illustrate the method we will elaborate on the pregnancy diagnoses decision tree in Figure 9.1 of Chapter 9.

The sequential decisions start from the left of the diagram. A decision node (a square) represents the decision to Make Pregnancy Diagnosis (MPD). From the decision node two branches originate, "Yes" and "No". If we follow the "yes branch", we encounter a chance node (a circle), the outcome of the pregnancy diagnosis, i.e. either "yes" (positive diagnosis) or "no" (negative). After the chance node, another decision node is encountered, "Replace" (RP), with "yes" and "no'" as actions. If we decide to keep the animal, the final chance node is encountered, "Farrowing" (FA) with possible outcomes, "yes" or "no". If we initially decide not to make pregnancy diagnosis, we have to wait and see what the outcome is at farrowing time.

At each branching in the nodes, we can assign a utility and a probability. If the pregnancy diagnosis is made, the cost is the additional work. If the animal is replaced, the income from the slaughter value of the animal plus the expected value of a new animal is received. If we decide to let the animals farrow, we add the feeding cost until farrowing, and the future value of the present animal. If it farrows, we obtain in addition the income from the litter produced. The probability of each branching in the chance nodes can be found similarly. If we make a pregnancy diagnosis the probability of the two outcomes depends on the Pregnancy State (PS) of the animal and the precision of the measurement method, but by applying Bayes formula, the probabilities can be found. (Note that the state of nature, PS, is not represented in the diagram). Similarly the probability of farrowing given PS and PD can be found and assigned to each branch leading from FA. To solve the decision tree diagram, one starts from the outmost branches (the leaves). In a chance node we calculate the expected utility. I.e. the expected utility in the FA node is the probability of farrowing times the income from a litter plus the probability of not farrowing times the income from not producing a litter (0). At a decision node we use maximizing instead of taking the expectation. I.e. we make the decision that have the highest expected utility. Gradually we move from the leaves to the stem/root of the decision tree.

As a real tree, the decision tree very soon becomes confusing to look at, i.e. the idea behind the graphic representation is soon lost. Furthermore, we had to do some calculations outside the tree, because some of the chance nodes were not represented (i.e. the pregnancy state). In addition some chance nodes are represented at several places in the diagram (e.g. the FA node)

The Influence diagram solved these problems to a large degree, when they were introduced by Howard and Matheson (1981). The use of the circle as a symbol for a chance node, and the square for a decision node was maintained, but in addition the diamond was added to represent the utility in the diagram. The branches

that represented the outcomes of decisions/chances, were removed and replaced by arrows that showed what influenced the decisions and chance nodes. The utility representation of the pregnancy diagnosis problem is also shown in Figure 9.1. The chance node PS (Pregnancy State) is added. The MPD node influences the node UD (Utility Diagnosis), and the outcome of the pregnancy diagnosis, and the next decision RP. The outcome of the pregnancy diagnosis is in addition influenced by the PS node. Note that PS influences FA directly disregarding the outcome of PD. The utility from replacement (UR) and from farrowing (UF) finishes the picture. (Strictly speaking, in the original approach only one mutual utility node was present).

Originally the influence diagrams were automatically translated to decision trees and solved using the same algorithm as mentioned above. Shachter (1986) showed how the influence diagram could be solved directly by removing nodes subsequently, based on 4 different transactions, the arc reversal (application of Bayes Theorem), node removal by summing out, expectation of a value node with respect to a chance node, and finally removal of a decision node into a value node by maximization. The first attempts were based on discrete valued state variables, but Shachter and Kenley (1989) implemented influence diagrams for Gaussian (normal) distributed variables.

12.3 From Bayesian network to influence diagram

If we look at the graphic representation of the decision graphs and compare with the Bayesian network, we find that they have the same nodes, i.e. the chance nodes, represented as circles. The two additional elements in decision graphs, the decision variables (squares) and utility functions (diamonds), can in fact also be handled within the framework of Bayesian networks by putting some restrictions on the solution method. This was first shown in Jensen et al. (1994). In fact, if the subsequent decisions are not sequential (i.e. the result of a previous decision does not influence other decisions) they can be directly represented in the Bayesian network by using the calculation trick, shown in Jensen (1996, Chapter 6). In realistic decision problems this is seldom the case.

The exact procedure for solving influence diagrams as it has been implemented in HUGIN, can be found in Jensen et al. (1994). Here we will summarize the method based on the diagram in Figure 9.1. An important difference between Bayesian networks and influence diagrams is the dynamic aspect, i.e. the sequential decisions impose an ordering on the elements. First of all, the decisions are ordered according to the sequence in which they are performed, ie. $U_D = \{MPD, RP\}$ in this case. The ordering is secured by putting directed edges between the decision nodes on the graph. Second, the chance nodes are put into sequential groups as well, i.e $U_R = \{I_0, I_1, I_2\}$, where I_0 is the set of observations before the first decision (in this case it is empty), I_1 is the set of observations between the first and the second decision, i.e. $I_1 = \{PD\}$, and finally the nodes that

are observed after the second decision or never observed, i.e. $I_2 = \{PS, FA\}$. We thus obtain a partial ordering,

$$\emptyset \prec \{MPD\} \prec \{PD\} \prec \{RP\} \prec \{PS, FA\}.$$

In Bayesian networks, the approach, as described in Jensen (1996, Section 4.5), would be to make a junction tree from the graph, by moralizing and triangulating the graph. In this case, however, a so-called strong junction tree is made. This means that the ordering, we have made, should be preserved in the junction tree, implying that the variables at the right should be eliminated before the variables on the left.

This ordering, which is a consequence of the "no forgetting" assumption discussed in Section 9.6, means that the computational complexity of influence diagrams with decision nodes in general is much larger than in Bayesian networks without.

As described in Jensen et al. (1994) we can then use the junction tree for propagation of evidence by simply interchanging maximization for calculation of expectation.

One note of caution is that the ordering should be preserved in the subsequent calculations made by in the network. In standard Hugin, evidence can be entered anywhere in the tree, and the evidence is distributed properly. This is not possible with influence diagrams. For example, it does not make sense to study the effect of a known pregnancy state, because evidence of pregnancy state is defined only to be available after all decisions have been made. Similarly the effect of known outcome of pregnancy diagnosis can only be studied for given level of the decision MPD. Unfortunately, the program does not have a facility that checks that evidence is input in the correct sequence. However, if the decisions are assigned fixed values, the evidence can be entered.

The rather new concept of *LImited Memory Influence Diagrams* described by Lauritzen and Nilsson (2001) relaxes the "no forgetting" assumption as mentioned in Section 9.6. The solution algorithm for this class of models is called *Single Policy Updating*, and all calculations are done in an ordinary junction tree just like those used with Bayesian networks (i.e. a *strong* junction tree is not needed). From a computational point of view, LIMIDs are therefore very appealing, but on the cost of exactness. In many cases, Single Policy Updating only leads to near-optimal solutions, even though they in most cases are very close to the optimum.

12.4 From Dynamic Programming to influence diagrams

The central paper in this respect is Tatman and Shachter (1990). Until the publication of their paper, influence diagrams were constructed with only one utility node. They showed that if the utility function was separable, i.e. either additive or multiplicative, the influence diagram technique corresponded to the Dynamic Pro-

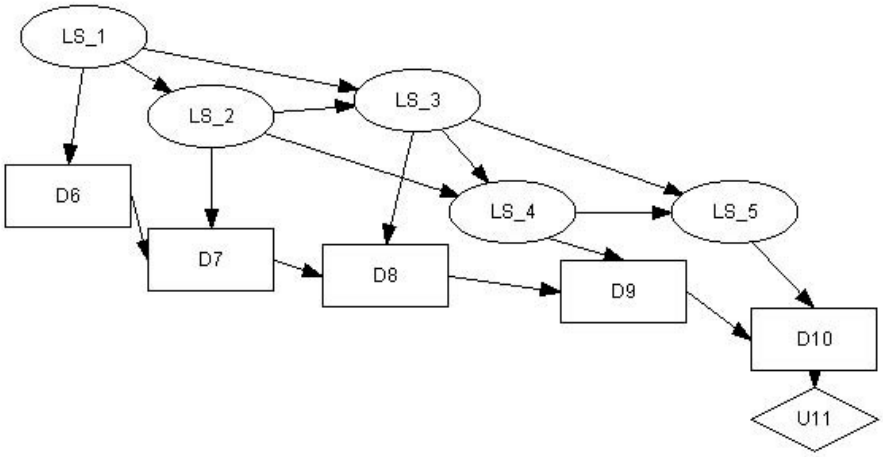


Figure 12.1: Influence Diagram representing part of the replacement problem for sows. LS_i is litter size in parity i, Dx represents decisions and Ux utility.

gramming method, i.e. Dynamic Programming is a special case of the influence diagram method.

Because of the more flexible approach towards representing the causal relations, the use of influence diagrams could often result in computational savings, by reducing the complexity of the problems.

To illustrate the potential of their techniques, Tatman and Shachter (1990) showed some examples. Their example (B) has close correspondence to the techniques used in solving the replacement problem in animal production, as exemplified by Huirne et al. (1991). Therefore, their example (B) is adapted to the replacement problem. The core in the replacement model by Huirne et al. (1991) is a model for the relation between litter size in subsequent parities. This can be illustrated as in Figure 12.1. (Note that the decision and utility part is not adequately represented in the figure).

The litter size is observed at subsequent parities and the decision based on the observation. The litter size is influenced by the litter size in the two previous parities as indicated by the arrows. Each litter size node has some 20 different levels $(0, \ldots, 19)$.

To solve this in standard dynamic programming, Huirne et al. (1991) used state augmentation, i.e. they made a state variable consisting of $(\text{LS\_i}, \text{LS\_i} + 1)$, as illustrated in the next figure, i.e. the state variable had $20^2 = 400$ states. This corresponds to the influence diagram in Figure 12.2.

The resulting transition matrix consisted of $400 \times 400 = 160,000$ elements. If the influence diagram shown if Figure 12.1 was used directly, the problem would only consist of matrices of $20 \times 400 = 8000$ elements. If information from further previous litters were included, the difference would be even more pronounced. It

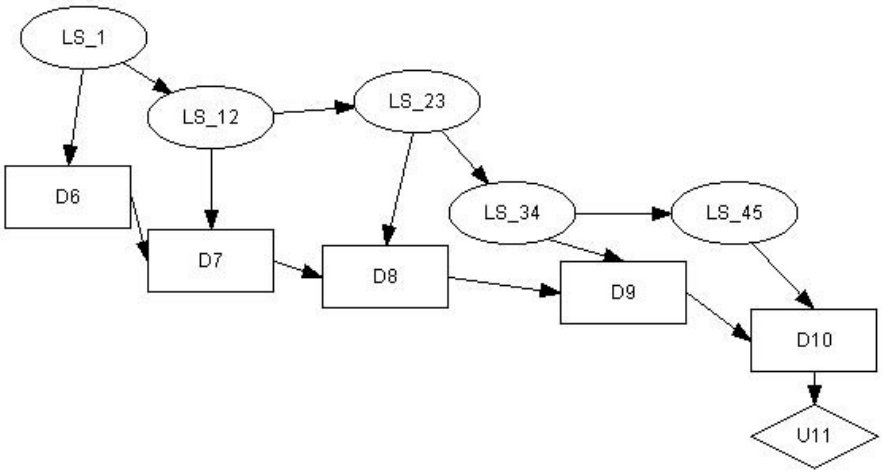


Figure 12.2: Representation of the state augmentation, LS_ij is the combined state variable of litter size in parity i and j.

is worth noting that this reduction in dimensionality is unrelated to the reductions due to hierarchic Markov processes (cf. Section 13.3.2). Similar savings in state space is expected in representing other independent components of the state space, e.g. state for pregnancy ability (number of re-matings).

However, in the current framework of decision graphs, only problems with finite planning horizon can be solved, as every time step has to be represented in the diagram. Furthermore, the use of semi-Markov decision processes rather than Markov decision processes, with the resulting flexibility in using different time scales has not been implemented either. Finally, the advantages of the hierarchic Markov process, where in fact a decision can select a process with a different causal structure is not yet possible.

12.5 Examples

The following is a presentation of examples of the possibility for using influence diagrams. The examples have not been constructed in detail but the overall structure of the decision problem should be clear, and quantification of the elements in the example should be possible. The intention is to stimulate ideas for application of the technique within animal production.

12.5.1 The two-sow problem

We shall once again return to the "two-sow problem" introduced in Example 6.1 of Chapter 6. As the reader will probably remember, a pig producer has bought two sows with the same expected farrowing date. One of the sows has to be culled because there is only room for one sow in the farrowing department. The litter size

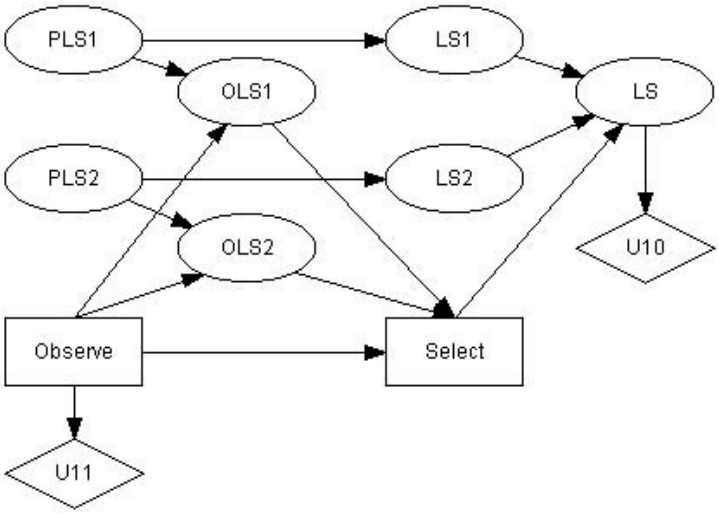


Figure 12.3: Influence diagram representation of the two sow problem.

of a sow is correlated with the litter size in her previous litter. Information about the litter size in the previous litter can be bought from the farmer, who sold the two sows. How much should he be willing to pay for the information, and which sow should he keep?

The example can be formulated as an influence diagram as shown in Figure 12.3. The litter size LSi of sow i depends on the previous litter size PLSi. We can decide to observe i.e. buy the information concerning the previous litter size, and obtain Observed Litter Size (OLSi). Based on the observed value we can decide which of the sows we should keep, and the resulting litter size is kept in the node LS.

12.5.2 The registration problem for the whole cycle

In fact the registration problem does not only concern the previous litter size, but also the use of heat-detection, pregnancy diagnosis etc. We can easily add nodes for Heat detection, Pregnancy diagnosis and the corresponding decision nodes and utility nodes to the diagram as shown in Figure 12.4 for the one sow case.

The decision nodes are UHD (use heat detection), UPD (use pregnancy diagnosis), and ULS (use litter size information). To each decision a utility node has been assigned. An additional chance node has been assigned, PS, representing pregnancy state. The network in 12.4 can be seen as a combination of Figure 9.1 and 12.3.

In Figure 12.5 the decision example is specified for the two sow problem. In addition nodes for herd level of pregnancy rate (HPR), heat detection rate (HDR) and efficiency of pregnancy diagnosis (Herdetect) has been added. These nodes will represent the uncertainty in the parameters used in the calculation. In addition

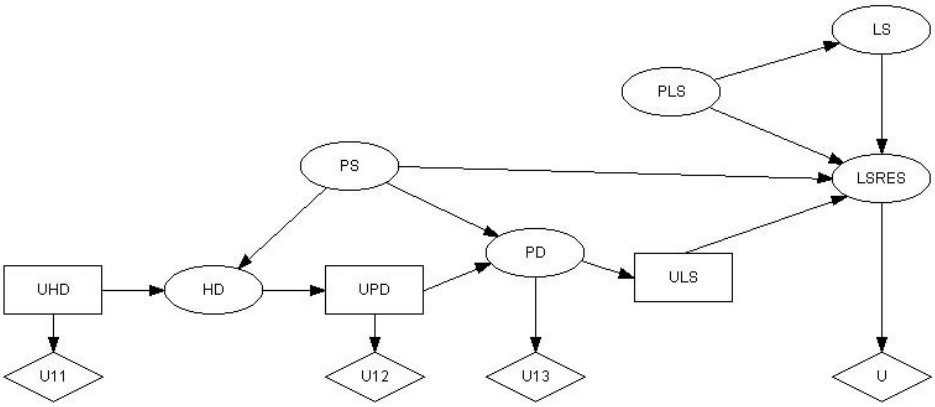


Figure 12.4: Influence diagram representing the decisions in a whole gestation period for a single sow.

the nodes will make the system "learn" from the observations made. A trick is used to ease the specification of the conditional probabilities. The nodes HDi, PDi and PLSi represent the outcome of the observations. Additional nodes XXOi represent the information that decisions are based upon, e.g. if a pregnancy diagnosis is performed the states of PDO will be identical to the state of PD. If observations are not made, the state "Not observed" of PDO will have probability 1, i.e. the variable contains no information about the true state.

The network in Figure 12.5 is an example of a network that should be used cautiously when entering evidence. The herd pregnancy rate is one of the nodes, where it would be tempting to ask: What if the pregnancy rate was 0.75, what would the optimal decision be. This is not allowed, because herd pregnancy rate is defined as an unobservable node, i.e. it does not influence any decision.

Note also that the complexity of Figure 12.5. Even with as few as two animals in the network, the decision graph is not readily understood. Of course the problem can be generalized to three sows, but now the diagram becomes almost impossible to understand. A method exists for easy representation of replicated processes, the so-called frames where a rectangular box around part of the network signify that this part of the network is to be replicated. This approach is eg. used in the specification in BUGS[3] (Gilks et al., 1993). This to some extent solves the problem of too complicated networks, and is especially suited for automatic learning within the network (Buntine, 1994). Such a representation for the network is shown in 12.6. Unfortunately this technique cannot yet be used within Hugin.

[3]The BUGS programme and manual can be found by visiting the web-adress with URL: http://www.mrc-bsu.cam.ac.uk/bugs/

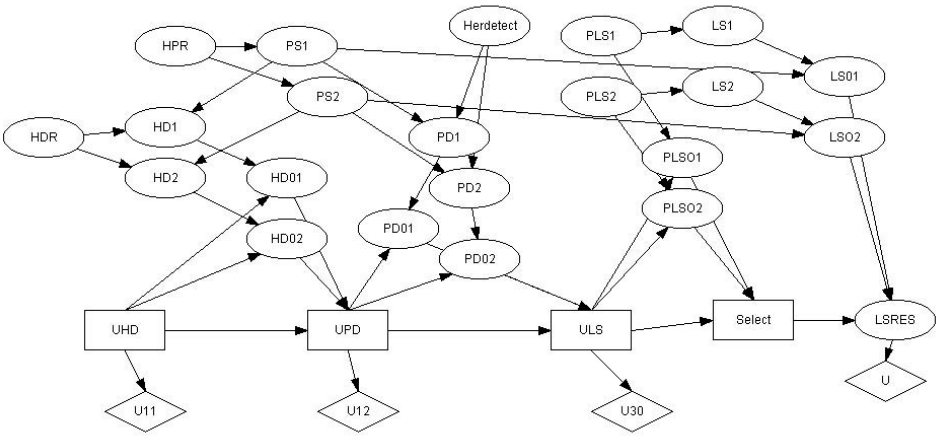


Figure 12.5: Influence diagram representing the decisions within gestation for two sows.

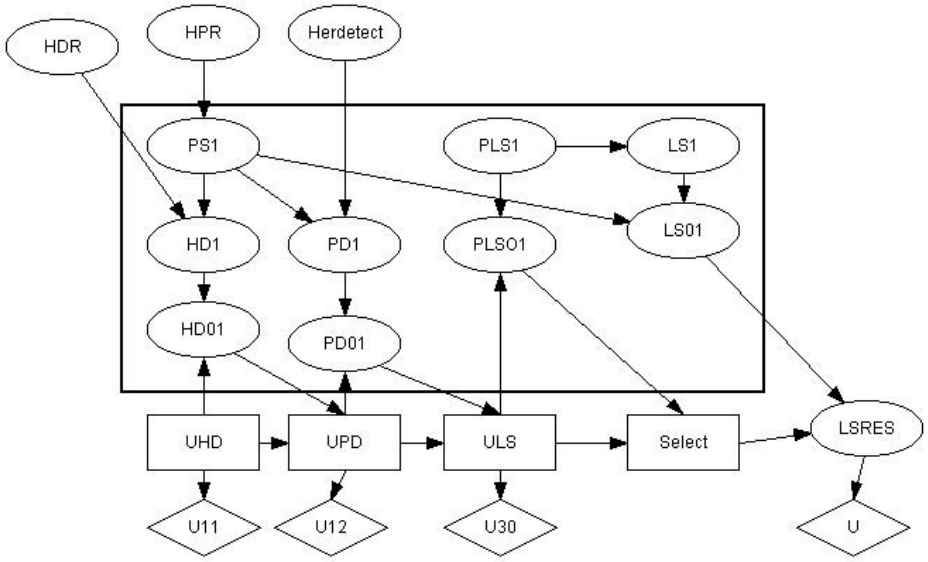


Figure 12.6: Frame representation of the general N-sow problem. Nodes within the frame is replicated automatically. (not yet possible within Hugin).

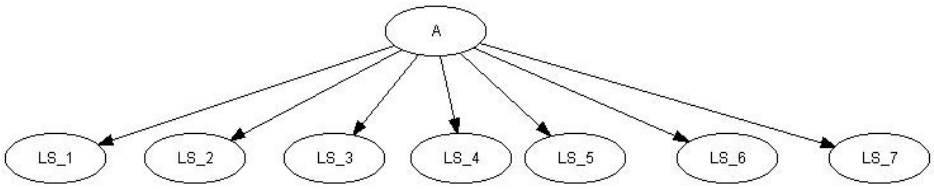


Figure 12.7: Representing model for litter size with additive effect of sow, A.

Another aspect is the computational complexity. Already with a few sows the problem becomes difficult to handle on a standard PC. As the complexity increases exponentially with the number of animals, clearly any computer will be overburdened if realistic problems are to be solved. Therefore, approximate methods need to be implemented within the diagram. In a similar framework, considering optimal number of sows mated, Greve (1995) implemented a heuristic algorithm and was able to solve problems of realistic size, i.e. herd sizes in excess of 500 sows.

12.5.3 The repeated measurement problem

The dynamic programming model by Huirne et al. (1991) assumed only effect of the two previous litters as described in Section 12.4. A more natural approach would be to assume an additive effect specific for each sow, i.e. a network like Figure 12.7.

The network corresponds to a statistical model represented as:

$$LS_{ij} = \mu_i + A_j + \varepsilon_{ij}$$

with μ_i as the mean of parity i, $A_j \sim \mathcal{N}(0, \sigma_A^2)$ as the effect of sow j and $\varepsilon_{ij} \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ as an independent and random residual.

Using ordinary Bayesian networks such a system can be handled very effectively, but not when influence diagrams are used. This is because of the strong junction tree mentioned earlier. The junction tree without decisions consists of cliques with $\{A, LS_p\}$, and is thus of a moderate size. However as A is an unobservable effect, it has to be included in the last set in the elimination ordering, when the influence diagram is compiled. That is, the triangulation should be made so that A can be eliminated before any of the other nodes. This can only be attained if links are added between litter size in every parity, and then we are back where we started. The problem shown in Figure 12.7 has a clique in the strong junction tree of size $O(N^7)$, where N is the number of state levels in each litter size node. The

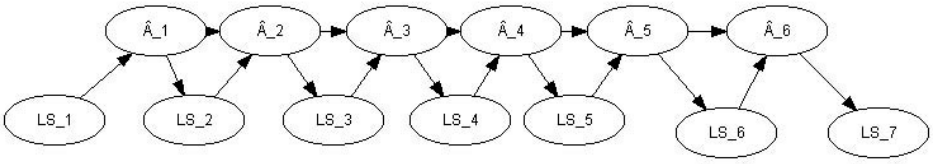


Figure 12.8: Efficient representation of the additive litter size model. A replaced by the observable estimate $\hat{A}_p$. Decions are based on $\hat{A}_p$.

same problem arises if any other of the current algorithms for solution in influence diagrams is used.

Fortunately, this problem can be circumvented. The trick is to use the underlying model to determine observable, $\hat{A}_{ij}$ values, i.e. estimate of the sow effect based on results from her previous litters, so that $LS_{i+1,j} = \mu_{i+1} + \beta(\hat{A}_{ij} - \mu_i) + \varepsilon$. In the case of the model specified, this can easily be found from the multivariate normal distribution. Then we obtain a network like Figure 12.8 instead. Note that the LS nodes are influenced by the estimate. From a causal point of view this of course nonsense, but it provides the correct answer and reduces the complexity drastically.

In 12.8 we have explicitly specified the learning or estimation process based on the observations, and based the decision on the learning. As described in Jensen (1996), Bayesian networks are an efficient method for a decision maker to learn posterior distributions from observed data. Therefore it seems a bit surprising that at the current state of development, when we specify the decision makers subsequent decisions within the network, we have to presume that he does not use this efficient learning method, but only bases his decisions on observed values. The alternative is to specify the learning algorithm directly, as in 12.8.

The figure corresponds to a decision maker that uses a Bayesian network like 12.7, as a DSS-tool and base his decisions on the mode of the probability distribution of A after evidence from previous litters has been added to the network.

Further illustration of the possibility for using this updating technique in connection with decision support can be found in Kristensen (1993) and Jørgensen (1992). For examples of applications, reference is made to Kristensen and Søllested (2004a,b); Lien et al. (2003).

12.5.4 Optimal timing of matings

After weaning most sows show oestrus/heat within 4 to 6 days. Based on the observation, the manager decides to mate the sow. In order to maximize the resulting conception rate and litter size the general advice is to repeat matings with 12 hours interval up to a maximum of three times, if the sow continues to show oestrus. The

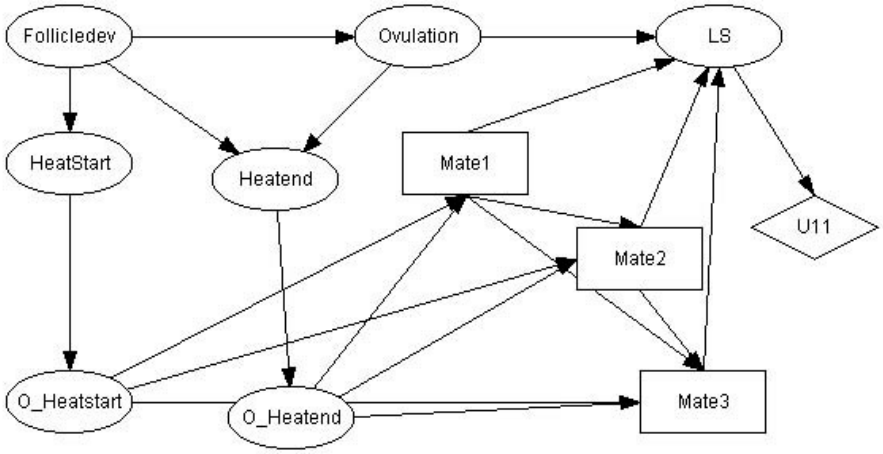


Figure 12.9: Influence diagram representing the problem of optimal timing of matings.

reason for this is that fresh semen (< 24h) should be available around the newly released ovum to increase fertilization success.

Ovulation can, in general, not be observed (see Soede, 1993, for methods for experimental methods for observing ovulation) and the pig producer has to rely on the observable signs of heat. Time of ovulation and start of heat are both depending on the hormonal development that can be summarized as follicular development (it is a complex interaction between the hormones LH, FSH and Oestrogen). In other words they are conditionally independent given follicle development. The end of oestrus is determined by follicular development and possible a feed back mechanism from ovulation. An influence diagram with the relevant mechanism is shown in Figure 12.9.

Start of heat (Heatstart) and end of heat (Heatend) is checked at regular intervals, e.g. three times a day. The strength of the oestrous signs develops almost following a gaussian curve. Experienced observers will detect weaker oestrous signs than inexperienced, that is, the heat will be observed (O_heatstart) earlier on average. Based on the observed oestrous sign, the pig producer shall decide the timing and frequency of matings (Mate1, Mate2 and Mate3). Depending on the timing of matings and the unobserved ovulation time (ovulation) the litter size (LS) will vary. In figure 12.9 the utility nodes are omitted from the mating decisions. Further refinements of the diagram would be to include decisions concerning oestrus detection schedule and connected costs.

12.5.5 The feed analysis problem

This example is modified from Pedersen (1996). In dairy production a major part of the feed mix consists of locally produced roughage, such as beets, silage and

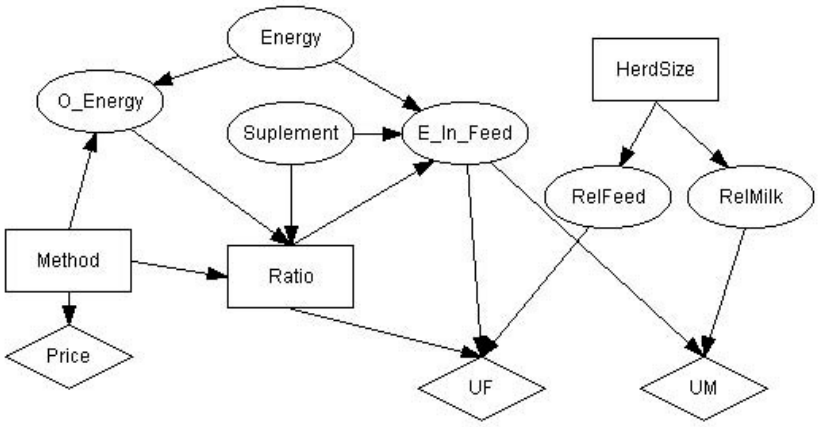


Figure 12.10: Influence diagram representing the ration formulation problem in a dairy herd under uncertainty.

straw. Knowledge of the energy contents of the feed is used in e.g. feed budgeting and feed planning. The energy content of roughage can be estimated using several methods. The simplest is a table look-up for standard figures for energy content in different categories of roughage. Another simple approach is to estimate dry matter content and digestibility subjectively, based on experience concerning the look and feel of the roughage. A more precise evaluation requires a laboratory assessment of the value. The digestibility can be measured using in vivo techniques that are the most precise. Alternatively, different in vitro techniques that simulate the in vivo techniques can be used to a much reduced price. Other aspects of the feed evaluation can be measured using different techniques, such as NIR.

The dairy farmer thus has to decide, how he should analyze the feed. Before this question can be answered, we have to consider the decisions that the feed analysis influences. In Figure 12.10 this is illustrated.

The roughage has a certain true and unobservable energy content (Energy). The farmer can decide to observe the energy level (O_Energy) using different methods (Method). These methods differ in price. Together with the observable energy level in the roughage and the energy level in the supplement feed (Suplement) the farmer decides the mixing ratio between these two feed stuffs. The resulting energy in the feed (E_In_feed) is found based on the mixing ratio and the true energy levels in the two feed stuffs. The energy level in the feed will in turn influence the feed intake and the feed costs (UF). In addition the milk production level will be influenced (UM). As the relative price of the feed analysis increases with decreasing herd size, because the information and the feed mixture are used on fewer animals, the utility should be corrected to take this into account. The two nodes Relfeed and Relmilk serve this function.

12.6 Outlook

When we judge the possible uses of the decision graphs, we should consider, that it is a very recent technique, and thus still have some shortcomings.

When we look upon the technique as a generalization of the Markov Chain process for Dynamic Programming, the more static approach used in influence diagrams is clearly a constraint. Easy specification of a dynamic system, such as used in dHugin (Kjærulff, 1995) would clearly be a step forward and should make it possible to utilize techniques comparable to policy iteration. To the authors' knowledge, this has not been developed yet. The flexible time scales, based on discounting the utilities, would represent an even further improvement.

The frame approach used in BUGS, see footnote on page 222, would ease the specification of multi-component systems. It may also be used to represent decision that chooses between qualitatively different subsystems, similar to Hierarchic Markov processes (HMP). The subsystem could be represented as different frames, and the decisions a matter of selecting the frame.

Such developments would make the influence diagram method able to solve similar problems as is currently solved by HMP. In addition, savings in complexity similar to those described in section 12.4 would be obtained.

The spinoff from the statistical research in Bayesian networks is expected to result in numerical methods for handling very large systems, as illustrated by the early attempts by Bielza et al. (1999) and Charnes and Shenoy (1996) to use Monte Carlo techniques to optimize the decision strategies. The savings in complexity by basing decisions on the learning process in Bayesian networks illustrated in section 12.5.3 is another obvious improvement.

Chapter 13

Dynamic programming and Markov decision processes

13.1 Introduction

13.1.1 Historical development

In the late fifties Bellman (1957) published a book entitled *Dynamic Programming*. In the book he presented the theory of a new numerical method for the solution of *sequential decision problems*. The basic elements of the method are the *Bellman principle of optimality* and *functional equations*. The idea may be illustrated as follows.

Consider a system being observed over a finite or infinite time horizon split up into periods or *stages*. At each stage, the *state* of the system is observed, and a *decision* (or an *action*) concerning the system has to be made. The decision influences (deterministically or stochastically) the state to be observed at the next stage, and depending on the state and the decision made, an immediate *reward* is gained. The expected total rewards from the present stage until the end of the planning horizon is expressed by a *value function*. The relation between the value function at the present stage and the one at the following stage is expressed by the *functional equation*. Optimal decisions depending on stage and state are determined backwards step by step as those maximizing the right hand side of the functional equation. This way of determining an optimal *policy* is based on the Bellman principle of optimality which says: *An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision* (Bellman, 1957, p. 83).

During the following years, Bellman published several books on the subject (Bellman, 1961; Bellman and Dreyfus, 1962; Bellman and Kalaba, 1965). The books were very enthusiastic, and the method was expected to be the solution to a very wide range of decision problems of the real world. The expectations were

so great, and they were adduced with such a conviction, that Johnston (1965) ironically compared dynamic programming to a new religion. Others regarded the method as a rather trivial computational device.

Similar stories might be told regarding other new numerical methods, as for instance linear programming. However, after some years, the applicational scopes of the methods are encircled. Most often the conclusion is that the method is neither an all-embracing technique nor a triviality. Between these extremities a rather narrow range of problems remains where it is a powerful tool. Other problems are, in some cases, not suitable to be solved by the method. In other cases alternative methods are better.

This also turned out to be the case in dynamic programming. One of the basic elements of dynamic programming is the sequential approach, which means that it fits sequential decision problems best. Obvious examples of sequential decisions in animal production include replacement of animals (it is relevant to consider at regular time intervals whether the present asset should be replaced or it should be kept for an additional period), insemination and medical treatment. Dynamic programming is a relevant tool, but if the traits of the animal are well defined and their precise behavior over time is known in advance, there are other methods that might be applied to determine the optimal decisions analytically. On the other hand, if the traits of the animal are affected by *random* variation over time and among animals, the decisions will depend on the present observations of the traits. In that case dynamic programming is an obvious technique to be used in the determination of optimal decisions and policies.

Having identified dynamic programming as a relevant method to be used with sequential decision problems in animal production, we shall continue on the historical development. Howard (1960) published a book on *Dynamic Programming and Markov Processes*. As will appear from the title, the idea of the book was to combine the dynamic programming technique with the mathematically well established notion of a *Markov chain*. A natural consequence of the combination was to use the term *Markov decision process* to describe the notion. Howard (1960) also contributed to the solution of infinite stage problems, where the *policy iteration method* was created as an alternative to the stepwise backward contraction method, which Howard called *value iteration*. The policy iteration was a result of the application of the Markov chain environment and it was an important contribution to the development of optimization techniques.

The policy iteration technique was developed for two criteria of optimality, namely maximization of total expected *discounted* rewards and maximization of expected *average* rewards per stage. Later on, Jewell (1963) presented a policy iteration technique for the maximization of average rewards over time for *semi-Markov decision processes*, which are Markov decision processes of which the stage length is a random variable. Howard (1971) presented a value iteration method for semi-Markov decision processes.

For the sake of completeness it should also be mentioned that *linear programming* was early identified as an optimization technique to be applied to Markov

decision processes as described by, for instance, Hadley (1964), but no animal production models known to the authors have applied that technique. This is in accordance with a conclusion of White and White (1989) that policy iteration (except in special cases) is more efficient than linear programming.

Since the publication of the first mentioned book by Howard (1960) an intensive research in Markov decision programming has been carried out. Many results have been achieved concerning the relations between the various optimization techniques and criteria of optimality. Reviews of these developments are given by van der Wal and Wessels (1985) as well as White and White (1989).

13.1.2 Applications in animal production

The dominant area of application in animal production has been for solving the animal replacement problem either alone or in connection with insemination and medical treatment. It is however expected that recent methodological developments will broaden the applicational scope.

Already three years after the book by Howard (1960), an application to the dairy cow replacement problem was published by Jenkins and Halter (1963). Their model only included the trait "lactation number" (at 12 levels), and the permanent value of the study was only to illustrate that Markov decision programming is a possible tool to be applied to the problem. A few years later, however, Giaever (1966) published a study which represents a turning-point in the application of the method to the animal (dairy cow) replacement problem. He considered all three optimization techniques (value iteration, policy iteration and linear programming), described how to ensure that all mathematical conditions were satisfied, and presented an eminent model to describe the production and feed intake of a dairy cow. The work by Giaever (1966) has not got the credit in literature that it deserves (maybe because it is only available on microfilm). In a review by van Arendonk (1984) it is not even mentioned.

During the following 20 years, several dairy cow replacement models using Markov decision programming were published, but from a *methodological* point of view none of them have contributed anything new compared to Giaever (1966). Several studies, however, have contributed in *other* ways. Smith (1971) showed that the rather small model of Giaever (1966) with 106 states did not represent the upper limit. His state space included more than 15 000 states. Kristensen and Østergaard (1982) as well as van Arendonk (1985, 1986) and van Arendonk and Dijkhuizen (1985) studied the influence of prices and other conditions on the optimal replacement policy. Other studies (Killen and Kearney, 1978; Reenberg, 1979) hardly reached the level of Jenkins and Halter (1963).

Even though the sow replacement problem is almost identical to that of dairy cows, very few *early* studies on sows have been published. The only exceptions known to the authors are Huirne et al. (1988, 1991, 1993) and Jørgensen (1992).

A study by Ben-Ari et al. (1983) deserves special attention. As regards methodology it is not remarkable, but in that study the main difficulties concerning appli-

cation to animal production models were identified and clearly formulated. Three features were mentioned:

Uniformity: The traits of an animal are difficult to define and measure. Furthermore the random variation of each trait is relatively large.

Reproductive cycle: The production of an animal is cyclic. It has to be decided *in which* cycle to replace as well as *when* to replace inside a cycle.

Availability: Only a limited supply of replacements (in that case heifers) is available.

The first feature in fact covers two different aspects, namely *uniformity* because the traits are difficult to define and measure, and *variability* because the traits vary at random among animals and over time. The third feature is an example of a *herd restraint*, i.e. a restriction that applies to the herd as a whole and not to the individual animal. Other examples of herd restraints are a production quota or a limited housing capacity. We shall therefore consider the more general problem of herd restraints.

We may conclude that until the mid-eighties, the methodological level concerning the application of Markov decision programming to animal production models was represented by Giaever (1966). The main difficulties that the method should overcome had been identified by Ben-Ari et al. (1983). If we compare the approach of Giaever (1966) to the difficulties that it ought to solve, we may conclude that the problems related to *variability* are directly solved, and as it has been shown by Kristensen and Østergaard (1982) as well as van Arendonk (1985); van Arendonk and Dijkhuizen (1985); van Arendonk (1986), the problems concerning the *cyclic production* may readily be solved without any methodological considerations. The only problem concerning variability and cyclic production is that in order to cover the variability, the state variables (traits) have to be represented by many levels, and to deal with the cyclic production a state variable representing the stage of the cycle has to be included. Both aspects contributes significantly to an explosive growth of the state space. We therefore face a *dimensionality* problem. Even though all necessary conditions of a Markov decision process are met, the solution in practice is prohibitive even on modern computers. The problems concerning uniformity and herd restraints are *not* solved by the approach of Giaever (1966).

13.2 Variability and cyclic production: Markov decision programming

As mentioned in the introduction, Markov decision programming is directly able to take the variability in traits and the cyclic production into account without any adaptations. In order to have a frame of reference, we shall briefly present the theory of traditional Markov decision programming originally described by Howard (1960).

Figure 13.1: Stages of a Markov decision process.

13.2.1 Notation and terminology

Consider a discrete time Markov decision process with a finite *state* space $U = \{1, 2, ..., u\}$ and a finite *action* set D. A *policy*, s is a map assigning to each state $i \in U$ an action $s(i) \in D$. Let p_{ij}^d be the *transition probability* from state i to state j if the action $d \in D$ is taken. The *reward* to be gained when the state i is observed, and the action d is taken, is denoted as r_i^d. The time interval between two transitions is called a *stage*.

We have now defined the elements of a traditional Markov decision process, but in some models we further assume that if state i is observed, and action d is taken, a physical quantity of m_i^d is involved (e.g. Kristensen, 1989, 1991). In this study we shall refer to m_i^d as the *physical output*. If $s(i) = d$, the symbols r_i^d, m_i^d and p_{ij}^d are also written as r_i^s, m_i^s and p_{ij}^s, respectively.

An *optimal* policy is defined as a policy that maximizes (or minimizes) some predefined objective function. The optimization technique (i.e. the method to identify an optimal policy) depends on the form of the objective function or - in other words - on the criterion of optimality. The over-all objective to maximize net revenue of the entire herd may (depending on the circumstances) result in different criteria of optimality formulated as alternative objective functions. The choice of criterion depends on whether the planning horizon is finite or infinite.

13.2.2 A simple dairy cow replacement model

For any dairy cow it is relevant to consider at regular time intervals whether it should be kept for an additional period or it should be replaced by a heifer. If the line of Figure 13.1 represents time, the markers indicate where we consider to replace. The time interval between two markers is called a stage and in this example we assume the stage length to be one year which for convenience is assumed always to be equal to a lactation period. At the beginning of each stage, we observe the state of the animal in production. The state space must be defined in such way that all relevant information is given by the state. In this very simple example we assume, that the only relevant information is whether the cow is low, average or high yielding. Thus we have got one state variable (milk yield) and three states.

Table 13.1: Rewards (gross margins), r_i^d, depending on state, i, and action, d.

State	$d = 1$ (Keep)	$d = 2$ (Replace)
$i = 1$ (low yielding)	10,000 DKK	9,000 DKK
$i = 2$ (average yielding)	12,000 DKK	11,000 DKK
$i = 3$ (high yielding)	14,000 DKK	13,000 DKK

Table 13.2: Physical outputs (expected milk yields), m_i^d, depending on state, i, and action, d.

State	$d = 1$ (Keep)	$d = 2$ (Replace)
$i = 1$ (low yielding)	5,000 kg	5,000 kg
$i = 2$ (average yielding)	6,000 kg	6,000 kg
$i = 3$ (high yielding)	7,000 kg	7,000 kg

Having observed the state, we have to take an action concerning the cow. We assume that the action is either to keep the cow for at least an additional stage or to replace it by a heifer at the end of the stage.

The economic net returns (gross margin) from the cow will of course depend on whether it is low yielding or high yielding **and** whether it is kept or replaced. In the model this is represented by a reward depending on state and action as appearing in Table 13.1. Those amounts are simply the annual net returns from a low, average and high yielding cow respectively. If the action replace is taken, we assume that the replacement takes place at the *end* of the stage at a cost of 1,000 DKK.

In this example, we shall define the milk yield during a stage as the physical output. In Table 13.2, the expected milk yield depending on state and action is shown. Since replacement is assumed to take place at the end of a stage, the milk yield will not depend on the action.

If a cow has been low yielding during a stage, there is large risk, that it will also be low yielding during the following stage if it is kept. This is illustrated by the transition probabilities from state i at a stage to state j at the following stage. We assume that the probability to remain at the same level of milk yield is 0.6. The probability of transition to an other level is assumed to be 0.3 from low (or high) to average and 0.1 from low to high or vice versa, if the cow is kept. On the other hand, if it is replaced, we assume that there are equal probabilities of the new heifer to be low, average or high yielding. All transition probabilities are shown in Table 13.3.

All parameters of a traditional Markov decision process are now defined, and we may consider what policy to follow. A policy is defined as a map (rule) that tells us which action to take if a certain state is observed. An example of a logical policy in this very simple case would be to replace if the cow is low yielding and keep if it is average or high yielding.

Our problem is now to determine an *optimal* policy, which in some sense maximizes the net returns of the dairy farmer. In the following section the phrase

Table 13.3: Transition probabilities from state i at a stage to state j at the following stage.

State at present stage	$d = 1$ (Keep)			$d = 2$ (Replace)		
	$j = 1$ (L)	$j = 2$ (A)	$j = 3$ (H)	$j = 1$ (L)	$j = 2$ (A)	$j = 3$ (H)
$i = 1$ (L)	0.6	0.3	0.1	0.333	0.333	0.333
$i = 2$ (A)	0.2	0.6	0.2	0.333	0.333	0.333
$i = 3$ (H)	0.1	0.3	0.6	0.333	0.333	0.333

"in some sense maximizes" is clarified. Afterwards, a survey of optimization techniques is given. Throughout the discussion we shall refer to this numerical example as *the simple dairy model*.

13.2.3 Criteria of optimality

Finite planning horizon

A farmer, who knows that he is going to terminate his production after N stages, may use the maximization of total expected rewards as his criterion of optimality. The corresponding objective function h is

$$h(s^1, ..., s^N) = \mathrm{E}\left(\sum_{n=1}^{N} r_{I(n)}^{s^n} \right), \tag{13.1}$$

where E denotes the expected value, s^n is the policy at stage n, and $I(n)$ is the (unknown) state at stage n. Applying this criterion to the simple dairy model means that the the total expected net returns over a fixed number (N) of years are maximized.

If the farmer has a time preference, so that he prefers an immediate reward to an identical reward later on, a better criterion is the maximization of total expected discounted rewards. If all stages are of equal length, this is equal to applying the objective function

$$h(s^1, ..., s^N) = \mathrm{E}\left(\sum_{n=1}^{N} \beta^{n-1} r_{I(n)}^{s^n} \right), \tag{13.2}$$

where $0 < \beta < 1$ is the discount factor defined by the interest rate and the stage length.

Infinite planning horizon

A situation where the stage of termination is unknown (but at least far ahead) is usually modeled by an infinite planning horizon (i.e. $N = \infty$). In that case the optimal policy is constant over stages. The function 13.1 cannot be applied in this

situation, but since $\beta < 1$, the function 13.2 will converge towards a fixed value for N becoming very large. Thus the objective function is given by

$$h(s) = \mathrm{E}\left(\sum_{n=1}^{\infty} \beta^{n-1} r^s_{I(n)}\right). \tag{13.3}$$

Since, usually, each animal and its future successors are represented by a separate Markov decision process, this criterion together with the criterion 13.2, are equal to the maximization of total discounted net revenues *per animal*. Such a criterion is relevant in a situation where a limiting housing capacity is the only (or at least the *most* limiting) herd restraint.

An alternative criterion under infinite planning horizon is the maximization of expected average reward per unit of time. If all stages are of equal length, the objective function in this situation is

$$h(s) = g^s = \sum_{i=1}^{u} \pi^s_i r^s_i, \tag{13.4}$$

where π^s_i is the limiting state probability under the policy s (i.e. when the policy is kept constant over an infinite number of stages). This criterion maximizes the average net revenues per stage, i.e. *over time*. In the simple dairy model, application of this criterion indicates that average annual net returns are maximized. It may be relevant under the same conditions as criterion 13.3 if an animal and its future successors are represented by a separate Markov decision process. Practical experience shows that the optimal replacement policies determined under criteria (13.3) and (13.4) are almost identical.

If a herd restraint (e.g. a milk quota) is imposed on the physical output, a relevant criterion may be the maximization of expected average reward per unit of physical output using the objective function

$$h(s) = g^s = \frac{g^s_r}{g^s_m} = \frac{\sum_{i=1}^{u} \pi^s_i r^s_i}{\sum_{i=1}^{u} \pi^s_i m^s_i}. \tag{13.5}$$

In case of a milk quota, the physical output m^s_i is the milk produced by a cow in state i under policy s, and accordingly, average net returns per kg milk is maximized in the simple dairy model. The function 13.5 is also relevant if the criterion is the maximization of the expected average reward over time in a model where the stage length varies. In that case the physical output represents the stage length. It should be noticed that if $m^d_i = 1$ for all i and d, the function 13.5 is identical to 13.4. The symbol g^s_r is the average reward over stages (equal to g^s of Eq. (13.4)) and g^s_m is the average physical output over stages.

13.2.4 Optimization techniques in general Markov decision programming

Value iteration

Under finite planning horizon the *value iteration method* is excellent. The optimal policies are determined sequentially using the functional equations

$$f_i(n) = \max_d \left\{ r_i^d + \beta \sum_{j=1}^{u} p_{ij}^d f_j(n-1) \right\}, \quad i = 1, ..., u, \tag{13.6}$$

where the action d maximizing the right hand side is optimal for state i at the stage in question. The function $f_i(n)$ is the total expected discounted rewards from the process when it starts from state i and will operate for n stages before termination. Thus $f_i(0)$ is the salvage value of the system when it is in state i. At each stage an optimal policy is chosen using Eqs. (13.6). If the objective function (13.1) is used, $\beta = 1$ in Eq. (13.6). Otherwise β is the discount factor.

Under infinite planning horizon, the value iteration method may be used to approximate an optimal policy. Under the objective function 13.3 it is possible to show that (Howard, 1960)

$$\lim_{n\to\infty} f_i(n) = f_i, \quad i = 1, ..., u, \tag{13.7}$$

where f_i for fixed i is a constant. By using Eqs. (13.6) over a large number of stages, we will sooner or later observe that $f_i(n+1)$ is almost equal to $f_i(n)$ for all i. Further, we will observe that the same policy is chosen during several stages. We can feel rather sure that such a policy is close to be optimal, but there is no guarantee that it is identical to an optimal policy. For practical purposes, however, the approximation usually suffices.

Since the objective function (13.4) is just a special case of function (13.5), where $m_i^s = 1$ for all i and d, we shall only consider the criterion given by (13.5). In this case $f_i(n)$ is the total expected rewards when the process starts from the beginning of a stage in state i and will operate *until n units of physical output have been produced.* Under the criterion given by the objective function (13.4), the production of n units of output is just the operation of the process over n stages. It is assumed that the physical output only takes integer values (for practical purpose this is just a question of selecting an appropriate unit). According to Howard (1971) an optimal policy for producing n units of output (i.e. a policy that maximizes the expected reward of producing n units) is determined recursively by the relations ($i = 1, \ldots, u$):

$$f_i(n) = \max_d \left\{ a\left(\frac{nr_i^d}{m_i^d} + f_i(0)\right) + (1-a)\left(r_i^d + \sum_{j=1}^{u} p_{ij}^d f_j(n-m_i^d)\right)\right\}, \quad n = 1, \ldots$$

where

$$a = \begin{cases} 1, & m_i^d \geq n \\ 0, & m_i^d < n \end{cases} \tag{13.8}$$

This is under the assumption that the reward/output rate has the constant value of r_i^d/m_i^d during the entire stage. However, since the physical output is bounded, it is easily seen that for n sufficiently large, $a = 0$. Hence we get for $i = 1, \ldots, u$

$$f_i(n) = \max_d \left\{ r_i^d + \sum_{j=1}^{u} p_{ij}^d f_j(n-m_i^d)\right\}, \quad large\ n. \tag{13.9}$$

Thus in the long run, the assumption concerning constant reward/output rate in all states will have no effect. The equivalence of Eq. (13.7) is in this case

$$\lim_{n\to\infty}(f_i(n) - f_i(n-1)) = g, \tag{13.10}$$

and sooner or later the policy will not differ from step to step of Eqs. (13.9).

Further details on the value iteration method are given by Howard (1960, 1971). It should particularly be noticed that m_i^d, which in this study is interpreted as a physical output (e.g. milk yield), in the study by Howard (1971) is interpreted as the expected *stage length* when state i is observed under the action d. Thus, in his model the criterion (13.5) is the expected average reward over time. Compared to Eq. (13.9), Howard (1971) described a more general case where the stage length is a random variable of which the distribution is given by the action and the present state as well as the state to be observed at the next stage. Furthermore, the reward depends on the state combination, the action and the stage length. The interpretation as physical output has been discussed by Kristensen (1991).

The value iteration method is identical to what is usually referred to as *dynamic programming*, *successive iteration* or *successive approximation*.

The simple dairy model optimized by value iteration

If we assume the discount factor, β, to be 0.85 and the salvage value of the system to be zero (independently of terminal state and action), we may directly apply the value iteration method as described by Eqs. (13.6). In Table 13.4, the results are shown stage by stage backwards from $n = 1$ to $n = 64$ (some stages omitted).

For any time horizon n, we may read the optimal policy directly from the table as the decisions d_1 for low yielding cows, d_2 for cows of average milk yield and

Table 13.4: The value iteration method of Eqs. (13.6) applied to the simple dairy model.

n	d_1	$f_1(n)$	$\Delta f_1(n)^*$	d_2	$f_2(n)$	$\Delta f_2(n)^*$	d_3	$f_3(n)$	$\Delta f_3(n)^*$
1	1	10000	10000	1	12000	12000	1	14000	14000
2	1	19350	9350	1	22200	10200	1	25050	11050
3	2	27870	8520	1	30870	8670	1	34081	9031
4	2	35299	7429	1	38275	7405	1	41622	7541
5	2	41639	6340	1	44597	6322	1	47988	6366
6	2	47030	5391	1	49981	5384	1	53385	5397
7	2	51612	4582	1	54561	4580	1	57969	4584
8	2	55507	3895	1	58455	3894	1	61864	3895
9	2	58817	3310	1	61765	3310	1	65175	3311
10	2	61631	2814	1	64579	2814	1	67989	2814
20	2	74437	554	1	77385	554	1	80795	554
30	2	76958	109	1	79906	109	1	83316	109
40	2	77455	21	1	80402	21	1	83812	21
50	2	77552	4	1	80500	4	1	83910	4
60	2	77572	1	1	80519	1	1	83929	1
61	2	77572	1	1	80520	1	1	83930	1
62	2	77573	1	1	80520	1	1	83930	1
63	2	77573	1	1	80521	1	1	83931	1
64	2	77574	0	1	80521	0	1	83931	0

$^*\Delta f_i(n) = f_i(n) - f_i(n-1)$

Table 13.5: The value iteration method of Eqs. (13.9) applied to the simple dairy model.

n	d_1	$f_1(n)$	$\Delta f_1(n)^*$	d_2	$f_2(n)$	$\Delta f_2(n)^*$	d_3	$f_3(n)$	$\Delta f_3(n)^*$
1	1	10000	10000	1	12000	12000	1	14000	14000
2	1	21000	11000	1	24000	12000	1	27000	13000
3	2	33000	12000	1	36000	12000	1	39500	12500
4	2	45167	12167	1	48100	12100	1	51800	12300
5	2	57355	12189	1	60253	12153	1	64027	12227
6	2	69545	12190	1	72428	12175	1	76228	12201
7	2	81734	12189	1	84612	12183	1	88420	12192
8	2	93922	12188	1	96798	12186	1	100609	12189
9	2	106109	12188	1	108985	12187	1	112797	12188
10	2	118297	12188	1	121172	12187	1	124984	12188
11	2	130484	12187	1	133359	12187	1	137172	12188
12	2	142672	12187	1	145547	12187	1	149359	12188
13	2	154859	12187	1	157734	12187	1	161547	12187
14	2	167047	12187	1	169922	12187	1	173734	12187
15	2	179234	12187	1	182109	12187	1	185922	12187

$^*\Delta f_i(n) = f_i(n) - f_i(n-1)$

d_3 for high yielding cows. If, for instance, we assume a time horizon of $n = 10$ stages, we see that low yielding cows should be replaced whereas average and high yielding cows should be kept. We also observe, that for values of n higher than 3, the optimal policy does not vary over stages.

The columns $f_i(n)$ show the expected present value of the chain (i.e. the present cow and its future successors), when the present cow is in state i and n stages remain in the time horizon. From the table we observe, that the present values of the three states converges towards fixed values in accordance with Eq. (13.7). Those values appear to be (close to) 77,574 DKK, 80,521 DKK and 83,931 DKK for low, average and high yielding cows respectively. The differences between these figures represent the relative values of the three states. For instance, $80,521 - 77,574 = 2,947$ is the economic advantage of having a cow of average milk yield instead of a low yielding cow.

In Table 13.5, the corresponding results for the value iteration method under the criterion (13.4) are shown. The optimal policies appear to be exactly identical to those of Table 13.4. Under this criterion, however, the value functions $f_i(n)$ does *not* converge towards fixed values, because no discounting is involved. Instead, we see from Table 13.5 that the *increments*, $\Delta f_i(n) = f_i(n) - f_i(n-1)$, in accordance with Eq. (13.10), converge towards a fixed value of 12,187 DKK independently of state. Thus, 12,187 is the numerical value of g, the annual net returns being maximized under this criterion.

It is also possible to calculate the economic benefit of, for instance, state 2 (average milk yield) over state 1 (low milk yield) from Table 13.5. If we use the

Table 13.6: Equations and expressions to be used in the policy iteration cycle with different objective functions.

Obj.	Linear equations of Step 2			Expression
func.	Equations, $i = 1, \ldots, u$	Unknowns	Add. eq.	Step 3
(13.3)	$f_i^s = r_i^s + \beta \sum_j p_{ij}^s f_j^s$	$f_1^s, \ldots, f_u^s$	-	$r_i^d + \beta \sum_j p_{ij}^d f_j^s$
(13.4)	$g^s + f_i^s = r_i^s + \sum_j p_{ij}^s f_j^s$	$g^s, f_1^s, \ldots, f_u^s$	$f_u^s = 0$	$r_i^d + \sum_j p_{ij}^d f_j^s$
(13.5)	$g^s m_i^s + f_i^s = r_i^s + \sum_j p_{ij}^s f_j^s$	$g^s, f_1^s, \ldots, f_u^s$	$f_u^s = 0$	$r_i^d - g^s m_i^d + \sum_j p_{ij}^d f_j^s$

figures relating to $n = 15$, the benefit mentioned is $(182,109 - 15 \times 12,187) - (179,234 - 15 \times 12,187) = -696 - (-3,571) = 2,875$. This figure is very close to the corresponding value (2,947) calculated from Table 13.4. This observation confirms the remark in relation to Criterion (13.4), that in practice results from this criterion only slightly differ from those under Criterion (13.3).

Policy iteration

Under infinite planning horizon, the *policy iteration method* may be applied. Unlike the value iteration method it always provides an optimal policy. It covers all three objective functions (13.3), (13.4) and (13.5). The iteration cycle used for optimization has the following steps:

1. Choose an arbitrary policy s. Go to 2.

2. Solve the set of linear simultaneous equations appearing in Table 13.6. Go to 3.

3. For each state, i, find the action d' that maximizes the expression given in Table 13.6, and put $s'(i) = d'$. If $s' = s$, then stop, since an optimal policy is found. Otherwise redefine s according to the new policy (i.e. put $s = s'$ and go back to 2.

From the equations and expressions of Table 13.6, we see that also with the policy iteration method, the objective function (13.4) is just a special case of (13.5), where $m_i^s = 1$ for all i and s. For the objective functions (13.3) and (13.4) the policy iteration method was developed by Howard (1960), and for the function (13.5) a policy iteration method was presented by Jewell (1963). Like Howard (1971), Jewell (1963) interpreted m_i^d as the expected stage length.

Under Criterion (13.3), f_i^s is the total present value of the expected future rewards of a process starting in state i and running over an infinite number of stages following the constant policy s. Under Criterions (13.4) and (13.5), f_i^s is the *relative value* of state i under the policy s. The difference in relative values between two states equals the amount of money a rational person is just willing to pay in order to start in the highest ranking of the two states instead of the lowest ranking.

The absolute value of f_i^s is determined arbitrarily by the additional equation of Table 13.6, where the relative value of state u is defined to be zero. The interpretation of relative values is discussed in details by Kristensen (1991).

Linear programming

Under an infinite planning horizon, linear programming is a possible optimization technique. When the criterion (13.3) is applied, the linear programming problem becomes (Ross, 1970)

$$\begin{aligned} \sum_{i=1}^{u} x_i \quad &= \quad Max! \\ subject\ to & \\ x_i - \beta \sum_{j=1}^{u} p_{ij}^d x_j \quad &\geq \quad r_i^d, \quad all\ d \in D, \quad i = 1, ..., u\ . \end{aligned} \tag{13.11}$$

It appears from 13.11 that each combination of state and action is represented by exactly one restriction. An action d is optimal in state i if, and only if, the corresponding restriction is satisfied as an equation when the values of $x_1, \ldots, x_u$ arises from an optimal solution to the linear programming problem. The *optimal* values of $x_1, \ldots, x_u$ are equal to the present values $f_1^s, \ldots, f_u^s$ under an optimal policy.

If the objective function (13.4) is applied, the linear programming problem becomes

$$\begin{aligned} \sum_{i=1}^{u} \sum_{d \in D} r_i^d x_i^d \quad &= \quad Max! \\ subject\ to & \\ \sum_{d \in D} x_i^d - \sum_{j=1}^{u} \sum_{d \in D} p_{ij}^d x_j^d \quad &= \quad 0, \quad i = 1, ..., u \\ \sum_{i=1}^{u} \sum_{d \in D} x_i^d \quad &= \quad 1 \\ x_i^d \quad &\geq \quad 0, \quad d \in D, \quad i = 1, ..., u. \end{aligned} \tag{13.12}$$

In this case an action d is optimal in state i if, and only if, x_i^d from the optimal solution is strictly positive. The optimal value of the objective function is equal to the average rewards per stage under an optimal policy. The optimal value of $\sum_{d \in D} x_i^d$ is equal to the limiting state probability π_i under an optimal policy.

Using Criterion (13.5), we may solve the following linear programming problem (Kennedy, 1986):

$$
\begin{aligned}
x_u &= Max! \\
& subject\ to \qquad (13.13)\\
-x_i + \sum_{j=1}^{u-1} p_{ij}^d x_j - m_i^d x_u &\leq -r_i^d - p_{iu}^d a, \quad d \in D, \quad i = 1, ..., u-1 \\
\sum_{j=1}^{u-1} p_{uj}^d x_j - m_u^d x_u &\leq -r_u^d - p_{uu}^d a + a, \quad d \in D \\
x_i &\geq 0, \quad i = 1, ..., u,
\end{aligned}
$$

where a is a pre-determined relative value of state u chosen sufficiently large to ensure that all other relative values are positive. The optimal value of the objective function of the linear programming problem is equal to the expected average reward per unit of output as defined in Eq. (13.5) under an optimal policy. The optimal values of the variables $x_1, \ldots, x_{u-1}$ are equal to the relative values of the states $1, \ldots, u-1$, provided that the relative value of state u is equal to a . As it appears, each combination of state and action is represented by one, and only one, restriction. An action is optimal in a state if, and only if, the corresponding restriction is satisfied as an equation in the optimal solution.

Since Criterion (13.4) is just a special case of (13.5) with all physical outputs set to the value 1, the linear programming problem (13.13) may also be used in the determination of an optimal policy under Criterion (13.4).

13.2.5 Discussion and applications

Under finite planning horizon, the value iteration method is perfect, but in herd management models the planning horizon is rarely well defined. Most often the process is assumed to operate over an unknown period of time with no pre-determined stage of termination. In such cases the abstraction of an infinite planning horizon seems more relevant. Therefore we shall pay specific attention to the optimization problem under the criteria (13.3), (13.4) and (13.5) where all three techniques described in the previous sections are available.

The value iteration method is not exact, and the convergence is rather slow. On the other hand, the mathematical formulation is very simple, and the method makes it possible to handle very large models with thousands of states. Further it is possible to let the reward and/or the physical output depend on the stage number in some pre-defined way. This has been mentioned by van Arendonk (1984) as an advantage in modeling genetic improvement over time. The method has been used in a lot of dairy cow replacement models as an approximation to the infinite stage optimum. Thus it has been used by Jenkins and Halter (1963); Giaever (1966); Smith (1971); McArthur (1973); Stewart et al. (1977, 1978); Killen and Kearney

(1978); Ben-Ari et al. (1983); van Arendonk (1985, 1986); van Arendonk and Dijkhuizen (1985); DeLorenzo et al. (1992); McCullough and DeLorenzo (1996a,b). Some of the models mentioned have been very large. For instance, the model by van Arendonk and Dijkhuizen (1985) contained 174 000 states (reported by van Arendonk, 1988). In sows, the method has been used by Huirne et al. (1988).

The policy iteration method has almost exactly the opposite characteristics of the value iteration method. Because of the more complicated mathematical formulation involving solution of large systems of simultaneous linear equations, the method can only handle rather small models with, say, a few hundred states. The solution of the linear equations implies the inversion of a matrix of the size $u \times u$, which is rather complicated. On the other hand, the method is exact and very efficient in the sense of fast convergence. The rewards are not allowed to depend on the stage except for a fixed rate of annual increase (e.g. inflation) or decrease. However, a seasonal variation in rewards or physical outputs is easily modeled by including a state variable describing season (each state is usually defined by the value of a number of state variables describing the system).

An advantage of the policy iteration method is that the equations in Table 13.6 are *general*. Under any policy, s, we are able to calculate directly the economic consequences of following the policy by solution of the equations. This makes it possible to compare the economic consequences of various non-optimal policies to those of the optimal. Further we may use the equations belonging to the criterion (13.5) to calculate the long run technical results under a given policy by redefining r_i^s and m_i^s. If for instance $r_i^s = 1$ if a calving takes place and zero otherwise, and m_i^s is the stage length when state i is observed under policy s, then g^s, which is the average number of calvings per cow per year, may be determined from the equations. Further examples are discussed by Kristensen (1991). For an example where the equations are used for calculation of the economic value of culling information, reference is made to Kristensen and Thysen (1991).

The policy iteration method has been used by Reenberg (1979) and Kristensen and Østergaard (1982). The models were very small, containing only 9 and 177 states, respectively.

13.3 The curse of dimensionality: Hierarchical Markov processes

In order to combine the computational advantages of the value iteration method with the exactness and efficiency of the policy iteration method Kristensen (1988, 1991) introduced a new notion of a hierarchical Markov process. It is a contribution to the solution of the problem referred to as the "curse of dimensionality" since it makes it possible to give exact solutions to models with even very large state spaces.

Table 13.7: Rewards (annual net returns) and outputs (annual milk yield) of extended model.

			Rewards, r_i^d		Outputs, m_i^d	
i	Lactation	Yield	$d = 1$ (K)	$d = 2$ (R)	$d = 1$ (K)	$d = 2$ (R)
1	1	Low	8,000	7,000	4,000	4,000
2	1	Ave.	10,000	9,000	5,000	5,000
3	1	High	12,000	11,000	6,000	6,000
4	2	Low	10,000	9,000	5,000	5,000
5	2	Ave.	12,000	11,000	6,000	6,000
6	2	High	14,000	13,000	7,000	7,000
7	3	Low	10,000	9,000	5,000	5,000
8	3	Ave.	12,000	11,000	6,000	6,000
9	3	High	14,000	13,000	7,000	7,000
10	4	Low	8,000	8,000	4,500	4,500
11	4	Ave.	10,000	10,000	5,500	5,500
12	4	High	12,000	12,000	6,500	6,500

13.3.1 The curse of dimensionality

In order to illustrate how the curse of dimensionality arises, we shall reexamine the simple dairy model used in the previous sections. First we shall realize that the *age* of the animal is not represented. Thus, if a cow remains high yielding it will never be replaced according to the optimal policies shown in Tables 13.4 and 13.5. This is certainly not realistic, and furthermore, the milk yield also depends on the lactation number. In order to account for age we shall introduce an additional state variable representing the lactation number of the cow. For convenience, we shall assume that the new variable may take the values 1, 2, 3 or 4 indicating that the maximum age of a cow in this (still very simple) model is assumed to be 4 lactations.

The milk yields (i.e. outputs m_i^d) and economic net returns (i.e. rewards r_i^d) assumed for this slightly extended model appear in Table 13.7. Because a cow is always replaced after 4 lactations, the rewards are identical under both actions for those states representing 4th lactation.

The transition matrices of the extended model are shown in Tables 13.8 and 13.9. It should be emphasizes that the state variable concerning milk yield should be interpreted relatively for a cow of the parity in question. As long as a cow is kept, it is assumed to change relative level of milk yield with the same probabilities as in the simple model, but when a replacement takes place, the new heifer is assumed to be low, average or high yielding with equal probabilities.

Inspection of the new matrices clearly illustrate that this very modest extension of the model causes a rather dramatic increase of the dimensions.

Now, suppose that we in addition to lactation and milk yield also want to take the genetic merit into account. We shall assume that the genetic merit of the cow

Table 13.8: Transition matrix of the extended model under action 1 (Keep).

		j	1	2	3	4	5	6	7	8	9	10	11	12
		l_j	1	1	1	2	2	2	3	3	3	4	4	4
i	l_i	Y	L	A	H	L	A	H	L	A	H	L	A	H
1	1	L	0	0	0	0.6	0.3	0.1	0	0	0	0	0	0
2	1	A	0	0	0	0.2	0.6	0.2	0	0	0	0	0	0
3	1	H	0	0	0	0.1	0.3	0.6	0	0	0	0	0	0
4	2	L	0	0	0	0	0	0	0.6	0.3	0.1	0	0	0
5	2	A	0	0	0	0	0	0	0.2	0.6	0.2	0	0	0
6	2	H	0	0	0	0	0	0	0.1	0.3	0.6	0	0	0
7	3	L	0	0	0	0	0	0	0	0	0	0.6	0.3	0.1
8	3	A	0	0	0	0	0	0	0	0	0	0.2	0.6	0.2
9	3	H	0	0	0	0	0	0	0	0	0	0.1	0.3	0.6
10	4	L	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
11	4	A	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
12	4	H	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0

Legends: l_i = lactation number of state i. Y = Milk yield indicated as L=Low, A=Average or H=High.

Table 13.9: Transition matrix of the extended model under action 2 (Replace).

		j	1	2	3	4	5	6	7	8	9	10	11	12
		l_j	1	1	1	2	2	2	3	3	3	4	4	4
i	l_i	Y	L	A	H	L	A	H	L	A	H	L	A	H
1	1	L	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
2	1	A	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
3	1	H	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
4	2	L	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
5	2	A	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
6	2	H	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
7	3	L	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
8	3	A	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
9	3	H	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
10	4	L	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
11	4	A	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0
12	4	H	1/3	1/3	1/3	0	0	0	0	0	0	0	0	0

Legends: l_i = lactation number of state i. Y = Milk yield indicated as L=Low, A=Average or H=High.

is either "Bad", "Average" or "Good". When a cow is replaced we assume that the probability of the new heifer to be either genetically "Bad", "Average" or "Good" is 1/3 each. The total size of the state space then becomes $3 \times 4 \times 3 = 36$. The milk yields m_i^d and rewards r_i^d appear from Table 13.10. The transition matrices of this 36-state model are now very large. They are shown in Appendix E as Tables E.1 - E.6 for actions "Keep" and "Replace", respectively.

This stepwise extension of the model clearly illustrates that each time a new state variable at n levels is added to the model, the size of the state space is increased by a factor of n. When, in a real model, several traits are represented by state variables at a realistic number of levels, the size of the state space very soon reaches prohibitive dimensions (millions of states). As an example, consider the dairy cow replacement model presented by Houben et al. (1994). The traits considered, when a decision was made, were

- The age of the cow (204 levels).
- Milk yield in present lactation (15 levels).
- Milk yield in previous lactation (15 levels).
- Time interval between 2 successive calvings (8 levels)
- Clinical mastitis - an infectious disease in the udder (2 levels - yes/no).
- Accumulated number of mastitis cases in present lactation (4 levels).

In principle the size of the state space is formed as the product of the number of levels of all traits, i.e. $204 \times 15 \times \ldots \times 4 = 11,750,400$ states. In practise it is smaller because some combinations are impossible and because the traits related to previous lactation are not considered during first lactation. Exclusion of such not feasible states resulted in a model with 6,821,724 different states. The model described only considers traits that vary over time for the same animal. If furthermore, we wish to include permanent traits of the present animal (like the genetic merit of the 36-state model) being considered for replacement, the state space would become even larger. In order to circumvent this curse of dimensionality, Kristensen (1988, 1991) introduced a new notion of a hierarchical Markov process, which is described in the following sections.

13.3.2 Notation and terminology

A hierarchical Markov process is only relevant under infinite planning horizon, and there is no relevance of the criterion (13.4) because the special situation where the physical output equals 1 in all stages has no computational advantages over other values. Therefore we shall only consider the criteria (13.3) and (13.5).

A hierarchical Markov process is a series of Markov decision processes called *child processes* built together in one Markov decision process called the *founder*

Table 13.10: Rewards and outputs of the 36 state model.

	Gen.			Rewards, r_i^d		Outputs, m_i^d	
i	merit	Lact.	Yield	$d = 1$ (K)	$d = 2$ (R)	$d = 1$ (K)	$d = 2$ (R)
1	Bad	1	Low	6,000	5,000	3,000	3,000
2	Bad	1	Ave.	8,000	7,000	4,000	4,000
3	Bad	1	High	10,000	9,000	5,000	5,000
4	Bad	2	Low	8,000	7,000	4,000	4,000
5	Bad	2	Ave.	10,000	9,000	5,000	5,000
6	Bad	2	High	12,000	11,000	6,000	6,000
7	Bad	3	Low	8,000	7,000	4,000	4,000
8	Bad	3	Ave.	10,000	9,000	5,000	5,000
9	Bad	3	High	12,000	11,000	6,000	6,000
10	Bad	4	Low	6,000	6,000	3,500	3,500
11	Bad	4	Ave.	8,000	8,000	4,500	4,500
12	Bad	4	High	10,000	10,000	5,500	5,500
13	Ave.	1	Low	8,000	7,000	4,000	4,000
14	Ave.	1	Ave.	10,000	9,000	5,000	5,000
15	Ave.	1	High	12,000	11,000	6,000	6,000
16	Ave.	2	Low	10,000	9,000	5,000	5,000
17	Ave.	2	Ave.	12,000	11,000	6,000	6,000
18	Ave.	2	High	14,000	13,000	7,000	7,000
19	Ave.	3	Low	10,000	9,000	5,000	5,000
20	Ave.	3	Ave.	12,000	11,000	6,000	6,000
21	Ave.	3	High	14,000	13,000	7,000	7,000
22	Ave.	4	Low	8,000	8,000	4,500	4,500
23	Ave.	4	Ave.	10,000	10,000	5,500	5,500
24	Ave.	4	High	12,000	12,000	6,500	6,500
25	Good	1	Low	10,000	9,000	5,000	5,000
26	Good	1	Ave.	12,000	11,000	6,000	6,000
27	Good	1	High	14,000	13,000	7,000	7,000
28	Good	2	Low	12,000	11,000	6,000	6,000
29	Good	2	Ave.	14,000	13,000	7,000	7,000
30	Good	2	High	16,000	15,000	8,000	8,000
31	Good	3	Low	12,000	11,000	6,000	6,000
32	Good	3	Ave.	14,000	13,000	7,000	7,000
33	Good	3	High	16,000	15,000	8,000	8,000
34	Good	4	Low	10,000	10,000	5,500	5,500
35	Good	4	Ave.	12,000	12,000	6,500	6,500
36	Good	4	High	14,000	14,000	7,500	7,500

process[1]. A child process is a finite time Markov decision process with N stages and a finite state space $\Omega_n = \{1, \ldots, u_n\}$ for stage n, $1 \leq n \leq N$. The action set D_n of the nth stage is assumed to be finite, too. A policy s of a child process is a map assigning to each stage n and state $i \in \Omega_n$ an action $s(n, i) \in D_n$. The set of all possible policies of a child process is denoted Γ. When the state i is observed and the action d is taken, a reward $r_i^d(n)$ is gained. The corresponding physical output is denoted as $m_i^d(n)$. Let $p_{ij}^s(n)$ be the transition probability from state i to state j where i is the state at the nth stage, j is the state at the following stage and d is the action taken at stage n. Under the Criterion (13.3) we shall denote the discount factor in state i under the action d as $\beta_i^d(n)$ assuming that the stage length is given by stage, state and action.

Assume that we have a set of v possible child processes each having its own individual set of parameters. The founder process is then a Markov decision process running over an infinite number of stages and having the finite state space $\{1, \ldots, v\}$. Each stage in this process represents a particular child process. The action sets of the main process are the sets Γ_ι, $\iota = 1, \ldots, v$, of all possible policies of the individual child processes (to avoid ambiguity the states of the founder process will be denoted by Greek letters ι, κ etc.). A policy σ is a map assigning to each state, ι, of the founder process an action $\sigma(\iota) \in \Gamma_\iota$. The transition matrix of the founder process has the dimension $v \times v$, and it is denoted $\Phi = \{\phi_{\iota\kappa}\}$. The transition probabilities are assumed to be independent of the action taken. The reward f_ι^σ and the physical output h_ι^σ in state ι of the founder process are determined from the total rewards and output functions of the corresponding subprocess as

$$
\begin{aligned}
f_i^s(n) &= r_i^s(n), \quad n = N \\
f_i^s(n) &= r_i^s(n) + \beta_i^s(n) \sum_{j=1}^{u_{n+1}} p_{ij}^s(n) f_j^s(n+1) \ , \quad n = 1, ..., N-1, \\
&\textit{and}, \\
f_\iota^\sigma &= \sum_{i=1}^{u_1} p_i(0) f_i^s(1), \quad s = \sigma(\iota) \ ,
\end{aligned}
\tag{13.14}
$$

and analogously for h_ι^σ (except for the discount factor). The symbol $p_i(0)$ is the probability of observing state i at the first stage of the child process. Finally, the expected discount factor in state ι under the action s is denoted as B_ι^s and calculated as follows

[1] In some texts, a child process is called a *sub process*, and the founder process is called the *main process*.

$$
\begin{aligned}
b_i^s(n) &= \beta_i^s(n), \quad n = N, \\
b_i^s(n) &= \beta_i^s(n) \sum_{j=1}^{u_{n+1}} p_{ij}^s(n) b_j^s(n+1), \quad n = 1, ..., N-1, \\
&and, \\
B_\iota^s &= \sum_{i=1}^{u_1} p_i(0) b_i^s(1). \qquad (13.15)
\end{aligned}
$$

13.3.3 Optimization

Since the founder process is just an ordinary Markov decision process, the policy iteration cycle described previously might be used directly for optimization. In practice Steps 1 and 2 are easily carried out, but Step 3 is prohibitive because of the extremely large number of alternative actions $s \in \Gamma_\iota$ (as mentioned above s is an entire policy of the ιth child process). To circumvent this problem Kristensen (1988, 1991) constructed an iterative method, where a value iteration method is applied in the child processes and the results are used in Step 3 of the policy iteration method of the founder process. The different versions of the method cover the criteria of optimality under infinite planning horizon defined as (13.3) and (13.5). Since criterion (13.4) is a special case of (13.5) it is also indirectly covered.

The general form of the iteration cycle of a hierarchical Markov process has the following steps:

1. Choose an arbitrary policy, σ. Go to 2.

2. Solve the following set of linear simultaneous equations for $F_1^\sigma, \ldots, F_v^\sigma$ and in case of Criterion (13.5) for g^σ:

$$
g^\sigma h_\iota^\sigma + F_\iota^\sigma = f_\iota^\sigma + B_\iota^\sigma \sum_{\kappa=1}^{v} \phi_{\iota\kappa} F_\kappa^\sigma, \quad \iota = 1, ..., v\ . \qquad (13.16)
$$

   In case of Criterion (13.5) the additional equation $F_v^\sigma = 0$ is necessary in order to determine a unique solution. Go to 3.

3. Define

$$
T_\iota = \sum_{\kappa=1}^{v} \phi_{\iota\kappa} F_\kappa^\sigma \qquad (13.17)
$$

under Criterion (13.3) and $T_\iota = 0$ under Criterion (13.5). For each child process, ι, find by means of the recurrence equations

$$\begin{aligned}
\tau_{\iota,i}(n) &= \max_d \left\{ r_i^d(n) - m_i^d(n) g^\sigma + \beta_i^d(N) T_\iota \right\}, \quad n = N \\
\tau_{\iota,i}(n) &= \max_d \left\{ r_i^d(n) - m_i^d(n) g^\sigma + \beta_i^d(n) \sum_{j=1}^{u_{n+1}} p_{ij}^d(n) \tau_{\iota,j}(n+1) \right\}, \\
& \qquad n = 1, ..., N-1
\end{aligned} \tag{13.18}$$

a policy s' of the child process. The action $s'(n)$ is equal to the d' that maximizes the right hand side of the recurrence equation of state i at stage n. Put $\sigma'(\iota) = s'$ for $\iota = 1, \ldots, v$. If $\sigma' = \sigma$, then stop since an optimal policy is found. Otherwise, redefine σ according to the new policy (i.e. put $\sigma = \sigma'$ and go back to 2.

When the iteration cycle is used under Criterion (13.3) all physical outputs ($m_i^d(n)$ and accordingly also h_ι^σ) are put equal to zero. The iteration cycle covering this situation was developed by Kristensen (1988).

Under Criterion (13.4) all physical outputs, $m_i^d(n)$, and all discount factors $\beta_i^d(n)$ and B_ι^σ are put equal to 1, but under Criterion (13.5) only the discount factors are put equal to 1. The iteration cycle covering these situations was described by Kristensen (1991).

13.3.4 Discussion and applications

The hierarchical Markov process is specially designed to fit the structure of animal decision problems where the successive stages of the child processes correspond to the age of the animal in question. By appropriate selection of state spaces in the child processes and the founder process it is possible to find optimal solutions to even very large models. The idea is to let the number of states in the child processes (where a value iteration technique is applied) be very large and only include very few states in the main process (where the technique is directly based on the policy iteration method). Thus we have got a method which is at the same time fast, exact and able to handle very large models.

Kristensen (1987) used the technique in a dairy cow replacement model which in a traditional formulation as an ordinary Markov decision process would have contained approximately 60,000 states, and later Kristensen (1989) in a model with approximately 180 000 states. In both cases the number of states in the founder process was only 5, reducing Step 2 to the solution of only 5 simultaneous linear equations (versus 180,000 in a traditional formulation). Haran (1997) also built a dairy cow replacement model based on a hierarchical Markov process. Also Houben et al. (1994) used the method in a dairy cow replacement model. The reduction of the system of equations was in their case from 6,821,724 to just 1,

because no permanent traits were considered in the model. An interesting aspect of their model is that for the first time a disease is taken into account.

In a different application area, Mourits et al. (1999) used a hierarchical Markov process in the determination of optimal feeding level and insemination age for dairy heifers. Similar ideas for use in fattening of bull calves have been presented by Makulska and Kristensen (1999).

In sows, Huirne et al. (1993) seem to have applied a technique which in many aspects is similar to a hierarchical Markov process, but they have not explained their method in all details. Also Jørgensen (1992) has applied a technique which is inspired of a hierarchical Markov process in a sow replacement model, and later (Jørgensen, 1993), he used the hierarchical method in the determination of optimal delivery policies in slaughter pigs. Also Broekmans (1992) used the method in the determination of optimal delivery policies in slaugther pigs taking random variation in prices into account. More recent applications to slaughter pigs include Kure (1997) and, in particular, Toft et al. (2005) who studied the influence of epidemic diseases on the slaughter policies. Verstegen et al. (1998) used the technique in an experimental economics study investigating the utility value of management information systems. They used a formulation involving Bayesian updating of traits as described by Kristensen (1993).

Naturally the hierarchical model just described may also be formulated as an ordinary Markov decision process. In that case each combination of child process (founder state), stage and state should be interpreted as a state. We shall denote a state in the transformed process as (ιni), and the parameters are

$$\begin{aligned} r^d_{\iota ni} &= r^d_i(n), \\ m^d_{\iota ni} &= m^d_i(n)\ , \\ \beta^d_{\iota ni} &= \beta^d_i(n), \\ p^d_{(\iota ni)(\kappa mj)} &= \begin{cases} p^d_{ij}(n), & \iota = \kappa \wedge m = n-1 \\ \phi_{\iota\kappa} p_i(0), & n = N \wedge m = 1 \\ 0, & otherwise \end{cases}\ , \end{aligned} \tag{13.19}$$

where the parameters mentioned on the right hand side of the equations are those belonging to the ιth child process except for $p_i(0)$ which belongs to child process κ. This formulation of course has the same optimal policies as the hierarchical formulation, so it is only computational advantages that make the hierarchical model relevant. A comparison to traditional methods may therefore be relevant.

Since the policy iteration method involves the solution of a set of u equations (where u is the number of states) it is only relevant for small models. The value iteration method, however, has been used with even very large models and may handle problems of the same size as the hierarchical formulation, but the time spend on optimization is much lower under the hierarchical formulation. To recognize this, we shall compare the calculations involved.

Step 3 of the hierarchical optimization involves exactly the same number of operations as one iteration of the value iteration method (Eq. 13.6). The further needs of the hierarchical method are the calculation of the rewards and *either* the physical output *or* the expected discount factor of a stage in the founder process according to Eqs. (13.14) and (13.15). Since the calculations at each stage is only carried out for one action, the calculation of both founder state parameters involves approximately the same number of operations as one iteration under the value iteration method if the number of alternative actions is 2. If the number of actions is higher, the calculations relatively involves a lower number of operations than an iteration under the value iteration method. These considerations are based on the assumption that the value iteration method is programmed in an efficient way, so that the sum of Eq. (13.6) is not calculated as a sum of all u elements, but only as a sum of those elements where p_{ij}^{d} is not zero according to Eq. (13.19). Otherwise the hierarchical technique will be even more superior. Finally the system of linear equations of Step 2 of the hierarchical cycle must be solved, but in large models with only a few states in the founder process the time spent on this is negligible.

If we use the considerations above in a practical example, the advantages of the hierarchical technique becomes obvious. As reported by Kristensen (1991) a model with 180,000 state combinations was optimized by the hierarchical technique under 100 different price conditions. The number of iterations needed ranged from 3 to 6 corresponding to between 6 and 12 iterations of the value iteration method. If the latter method was used instead, a planning horizon of 20 years would be realistic (cf. van Arendonk, 1985). Since each state in the model equals 4 weeks, this horizon represents 260 iterations, which should be compared to the equivalence of from 6 to 12 when the hierarchical technique was applied.

13.3.5 The numerical example formulated and solved as a hierarchical Markov process

In order to illustrate the hierarchical technique, we shall formulate the numerical example (the 36-state model) as a hierarchical Markov process. A Java based software tool for hierarchical Markov processes has been developed by Kristensen (2003a)[2]. The following example is pre-defined and comes with the program.

The three classes of the genetic merit are defined as states in the founder process of a hierarchical Markov process. Thus the number of child processes is also 3, and each child process represents a dairy cow of a certain genetic merit. When a new heifer is purchased, we assume, like before, that the probability distribution over main states is uniform, so that the probability of entering either one is 1/3. The maximum age of a cow was assumed to be 4 lactations, and the states of the child process are defined from the relative level of milk yield. Further a dummy state

[2]The program is available on World Wide Web. A Windows installer file may be downloaded from URL: http://www.prodstyr.ihh.kvl.dk/software/mlhmp.html

Table 13.11: Parameters of the hierarchical Markov process: Transition probabilities of founder process and initial state probabilities of child processes.

Transition probabilities, founder				Initial state probabilities, children			
State	$\phi_{\iota\kappa}$			$p_i(0)$			
ι	κ=1	κ=2	κ=3	$i=1$	$i=2$	$i=3$	$i=4$
1	1/3	1/3	1/3	1/3	1/3	1/3	0
2	1/3	1/3	1/3	1/3	1/3	1/3	0
3	1/3	1/3	1/3	1/3	1/3	1/3	0

of length, reward and output equal to 0 is included at each stage of the child processes. If the cow is replaced at the end of a stage, the process enters the dummy state with probability 1 at the next stage, and for the rest of the duration of the subprocess it will stay in the dummy states. Stage numbers in the child processes directly correspond to lactation numbers.

Thus, for all child processes, the probability of staying at the same relative level of milk yield (state in the subprocess) is 0.6, and if the present state is "Average", the probability of transition to either "Low" or "High" is 0.2 each. The probability of transition (if kept) from "Low" or "High" to "Average" is in both cases 0.3, and from "Low" to "High" and vice versa the probability is 0.1. The initial state probabilities of the child processes represent the probability of a new heifer being either "Low", "Average" or "High" yielding. Thus, all initial state probabilities are 1/3.

Like before, the physical outputs are interpreted as milk yields and the rewards are defined as the economic net returns. All parameters of the hierarchical model are shown in Tables 13.12 and 13.11.

We shall determine an optimal solution under the following 3 criteria of optimality:

1. Maximization of total expected discounted rewards, i.e., the objective function (13.3). In this case the physical outputs of Table 13.12 are ignored, and a discount factor $\beta_i^d(n) = \exp(-r)$, where r is the interest rate, is applied (for states where the stage length is not zero).

2. Maximization of average rewards over time. In this situation we use the objective function (13.5) letting the output represent stage length. No discounting is performed in this case.

3. Maximization of average rewards over output defined as in Table 13.12. Thus the objective function (13.5) is applied, and no discounting is performed.

In Table 13.13, optimal policies under the three criteria are shown. It appears that the policies under the first two criteria are quite similar, but under the third

Table 13.12: Parameters of the hierarchical Markov process, child processes.

			$p_{ij}^1(n)$						$p_{ij}^2(n)$					
ι	n	i	j=1	j=2	j=3	j=4	$r_i^1(n)$	$m_i^1(n)$	j=1	j=2	j=3	j=4	$r_i^2(n)$	$m_i^2(n)$
1	1	1	0.6	0.3	0.1	0.0	6,000	3,000	0.0	0.0	0.0	1.0	5,000	3,000
1	1	2	0.2	0.6	0.2	0.0	8,000	4,000	0.0	0.0	0.0	1.0	7,000	4,000
1	1	3	0.1	0.3	0.6	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
1	1	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
1	2	1	0.6	0.3	0.1	0.0	8,000	4,000	0.0	0.0	0.0	1.0	7,000	4,000
1	2	2	0.2	0.6	0.2	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
1	2	3	0.1	0.3	0.6	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
1	2	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
1	3	1	0.6	0.3	0.1	0.0	8,000	4,000	0.0	0.0	0.0	1.0	7,000	4,000
1	3	2	0.2	0.6	0.2	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
1	3	3	0.1	0.3	0.6	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
1	3	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
1	4	1	-	-	-	-	6,000	3,500	-	-	-	-	6,000	3,500
1	4	2	-	-	-	-	8,000	4,500	-	-	-	-	8,000	4,500
1	4	3	-	-	-	-	10,000	5,500	-	-	-	-	10,000	5,500
1	4	4	-	-	-	-	0	0	-	-	-	-	0	0
2	1	1	0.6	0.3	0.1	0.0	8,000	4,000	0.0	0.0	0.0	1,0	7,000	4,000
2	1	2	0.2	0.6	0.2	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
2	1	3	0.1	0.3	0.6	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
2	1	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
2	2	1	0.6	0.3	0.1	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
2	2	2	0.2	0.6	0.2	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
2	2	3	0.1	0.3	0.6	0.0	14,000	7,000	0.0	0.0	0.0	1.0	13,000	7,000
2	2	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
2	3	1	0.6	0.3	0.1	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
2	3	2	0.2	0.6	0.2	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
2	3	3	0.1	0.3	0.6	0.0	14,000	7,000	0.0	0.0	0.0	1.0	13,000	7,000
2	3	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
2	4	1	-	-	-	-	8,000	4,500	-	-	-	-	8,000	4,500
2	4	2	-	-	-	-	10,000	5,500	-	-	-	-	10,000	5,500
2	4	3	-	-	-	-	12,000	6,500	-	-	-	-	12,000	6,500
2	4	4	-	-	-	-	0	0	-	-	-	-	0	0
3	1	1	0.6	0.3	0.1	0.0	10,000	5,000	0.0	0.0	0.0	1.0	9,000	5,000
3	1	2	0.2	0.6	0.2	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
3	1	3	0.1	0.3	0.6	0.0	14,000	7,000	0.0	0.0	0.0	1.0	13,000	7,000
3	1	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
3	2	1	0.6	0.3	0.1	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
3	2	2	0.2	0.6	0.2	0.0	14,000	7,000	0.0	0.0	0.0	1.0	13,000	7,000
3	2	3	0.1	0.3	0.6	0.0	16,000	8,000	0.0	0.0	0.0	1.0	15,000	8,000
3	2	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
3	3	1	0.6	0.3	0.1	0.0	12,000	6,000	0.0	0.0	0.0	1.0	11,000	6,000
3	3	2	0.2	0.6	0.2	0.0	14,000	7,000	0.0	0.0	0.0	1.0	13,000	7,000
3	3	3	0.1	0.3	0.6	0.0	16,000	8,000	0.0	0.0	0.0	1.0	15,000	8,000
3	3	4	0.0	0.0	0.0	1.0	0	0	0.0	0.0	0.0	1.0	0	0
3	4	1	-	-	-	-	10,000	5,500	-	-	-	-	10,000	5,500
3	4	2	-	-	-	-	12,000	6,500	-	-	-	-	12,000	6,500
3	4	3	-	-	-	-	14,000	7,500	-	-	-	-	14,000	7,500
3	4	4	-	-	-	-	0	0	-	-	-	-	0	0

Legends: ι = Child process (Founder State), n = Stage number, i = State at Stage n, j = State at Stage $n+1$.

Table 13.13: Optimal policies under the three criteria (c1, c2, c3) defined in the text (actions: 1 = "Keep", 2 = "Replace").

		State 1			State 2			State 3		
Subprocess	Stage	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
1	1	2	2	1	2	2	1	2	1	1
1	2	2	2	1	2	2	1	2	1	1
1	3	2	2	1	2	2	1	2	2	1
2	1	1	1	1	1	1	1	1	1	1
2	2	2	1	1	1	1	1	1	1	1
2	3	2	2	1	2	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1
3	2	1	1	1	1	1	1	1	1	1
3	3	1	1	1	1	1	1	1	1	1

Table 13.14: The performance of the hierarchical technique compared to the policy and value iteration methods under the three criteria (c1, c2, c3) defined in the text.

	Hierarchical model			Policy iteration			Value iteration		
	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
Number of iterations	4	3	2	4	4	2	100	100	100
Computer time, rel.	1	0.86	0.43	268	267	139	48	46	11

criterion the optimal policy differs significantly. A more detailed example of the effect of criterion of optimality was discussed by Kristensen (1991).

In order to compare the efficiency of the hierarchical technique to the traditional policy and value iteration methods, the problem of the example was transformed to an ordinary Markov decision process and optimized by those methods. The transformed model has $3 \times 4 \times 4 = 48$ states, which is not larger than the policy iteration method may be applied without problems. In Table 13.14 some performance data of the three optimization techniques are compared.

The superiority of the hierarchical technique over the policy iteration method is due mainly to the time spent on solving the linear simultaneous equations of Step 2. In the hierarchical case a system of 3 equations is solved, whereas 48 equations are solved under the ordinary policy iteration method.

In this numerical example the performance of the hierarchical technique is even more superior to the value iteration method than expected from the theoretical considerations of Section 13.3.4. In the present case an iteration of the hierarchical model is performed even faster than one of the value iteration method applied to the same (transformed) model. The reason is that the value iteration algorithm has not been programmed in the most efficient way as defined in Section 13.3.4. On the contrary, the sum of Eq. (13.6) has been calculated over all 48 states of the trans-

formed model. Since only 4 transition probabilities from each state are positive, the sum could be calculated only over these 4 states.

13.4 Uniformity: Bayesian updating

13.4.1 Principles of learning from observations

As discussed earlier, it is obvious that the traits of an animal varies no matter whether we are considering the milk yield of a dairy cow, the litter size of a sow or almost any other trait. On the other hand, it is *not* obvious to what extent the *observed* trait Y_n at stage n is, for instant, the result of a permanent property of the animal X_1, a permanent damage caused by a previous disease X_2 or a temporary random fluctuation e_n. Most often the observed value is the result of several permanent and random effects. With Y_n, X_1, X_2 and e_n defined as above the relation might for instance be

$$Y_n = m + X_1 + aX_2 + e_n, \tag{13.20}$$

where m is the expected value for an arbitrarily selected animal under the circumstances in question, and $a = -1$ if the animal has been suffering from the disease, and $a = 0$ otherwise. In this example, X_1 only varies *among* animals, whereas e_n also varies *over time* for the same animal. The effect of the damage caused by the disease X_2 is in this example assumed to be constant over time when it has been "switched on". The value of X_2 is a property of the individual disease case (the "severity" of the case).

In a replacement decision it is of course important to know whether the observed value is mainly a result of a permanent effect or it is just the result of a temporary fluctuation. The problem, however, is that only the resulting value Y_n is observed, whereas the values of X_1, X_2 and e_n are unknown. On the other hand, as observations of $Y_1, Y_2, \ldots$ are done, we are learning something about the value of the permanent effects. Furthermore, we have got a *prior* distribution of X_1 and X_2, and each time an observation is done, we are able to calculate the *posterior* distribution of X_1 and X_2 by means of the Kalman-filter theory in connection with Dynamic Linear Models (described for instance by West and Harrison, 1997) if we assume all effects to be normally distributed.

A model as described by Eq. (13.20) fits very well into the structure of a hierarchical Markov process. Thus we may regard Y_n as a state variable in a child process, and the permanent effects X_1 and X_2 as state variables of the founder process. We then face a hierarchical Markov process with *unobservable founder state*. Kristensen (1993) discusses this notion in details, and it is shown that under the assumption of normally distributed effects, we only have to keep the present expected values of X_1 and X_2, the currently observed value of Y_n and (in this example) the number of stages since the animal was suffering from the disease (if it has been suffering from the disease at all). The expectations of X_1 and X_2 are

sufficient to determine the current posterior distribution of the variables, because the variance is known in advance. Even though the posterior variance decreases as observations are done, the decrease does *not* depend on the *values* of $Y_1, Y_2, \ldots$, but only on the number of observations done.

In the study of Kristensen (1993), a more general case involving several traits each being influenced by several unobservable effects is sketched, and a numerical example involving only a single trait is given. An example concerning replacement of sows has been given by Jørgensen (1992). It was demonstrated in both studies that the Bayesian approach in some cases may result in state space reduction without loss of information.

In multi-trait updating models a Kalman filter technique based on state space models may be relevant as described by Kristensen (1994). Recently, Nielsen et al. (2011) provided general guidelines for embedding state space models into a Markov decision process.

13.4.2 Applications

The described principles for Bayesian updating have in particular been used in combination with Dynamic Linear Models (see West and Harrison, 1997, for a detailed description of the concept). In addition to the model by Jørgensen (1992), such models with built-in learning capabilities have been developed by Kristensen and Søllested (2004a,b) for sow replacement taking all litter size results into account. Nielsen et al. (2010) presented a dairy cow replacement model where the milk yield capacity of a cow was updated daily based on data from automatic milking systems. Lien et al. (2003) developed a model for determination of optimal length of leys in an area with winter damage problems, where the productivity of the ley is learned over time as observations are done. Finally, the principles are used in a slaughter pig marketing model with emphasis on information from online weighing equipment (Kristensen et al., 2012).

An example of embedding a Dynamic Generalized Linear Model into a Markov decision process is given by Ge et al. (2010b,a) who used the technique for managing foot-and-mouth disease epidemics. The number of new cases were described by a Poisson process, and the properties of the epidemic were updated based on the observations.

13.5 Herd restraints: Parameter iteration

One of the major difficulties identified in the introduction was *herd restraints*. All the replacement models mentioned in the previous sections have been single-component models, i.e., only one animal is considered at the same time, assuming an unlimited supply of all resources (heifers or gilts for replacement, feed, labor etc) and no production quota. In a multi-component model all animals of a herd are simultaneously considered for replacement. If all animals (components) com-

pete for the same limited resource or quota, the replacement decision concerning an animal does not only depend on the state of that particular animal, but also on the states of the other animals (components) of the herd.

If the *only* (or at least the *most limiting*) herd restraint is a limited housing capacity, the number of animals in production is the scarce resource, and accordingly the relevant criterion of optimality is the maximization of net revenues per animal as it is expressed in the criteria (13.1), (13.2), (13.3) and (13.4). Thus the optimal replacement policy of the single component model is optimal for the multi-component model too.

If the only (or most limiting) herd restraint is a milk quota, the situation is much more complicated. Since the most limiting restriction is a fixed amount of milk to produce, the relevant criterion of optimality is now the maximization of average net revenues per kg milk yield as expressed in criterion (13.5), because a policy that maximizes net revenues per kg milk will also maximize total net revenues from the herd which was assumed to be the objective of the farmer.

By following a policy which is optimal according to criterion (13.5) we assure at any time that the cows which produce milk in the cheapest way are kept. Thus the problem of selecting which cows to keep in the long run (and the mutual ranking of cows) is solved, but the problem of determining the optimal number of cows in production at any time is *not* solved. If for instance, it is recognized 2 months before the end of the quota year that the quota is expected to be exceeded by 10 percent, we have to choose whether to reduce the herd size or to keep the cows and pay the penalty. The problem is that both decisions will influence the possibilities of meeting the quota of the next year in an optimal way. To solve this short run quota adjustment problem we need a true multi-component model.

An other example of a herd restraint is a limited supply of heifers. If the dairy farmer only uses home-grown heifers for replacement, the actions concerning individual cows become inter-dependent, and again a multi-component model is needed in order to solve the replacement problem. Ben-Ari and Gal (1986) and later Kristensen (1992) demonstrated that the replacement problem in a dairy herd with cows and a limited supply of home grown heifers may be formulated as a Markov decision process involving millions of states. This multi-component model is based on a usual single-component Markov decision process representing one cow and its future successors. Even though the hierarchical technique has made the solution of even very large models possible, such a model is far too large for optimization in practice. Therefore, the need for an approximate method emerged, and a method called *parameter iteration* was introduced by Ben-Ari and Gal (1986).

The basic idea of the method is to approximate either the present value function $f_i(n)$ (objective function (13.3)) or the relative values f_i^s (objective functions (13.4) and (13.5)) by a function G involving a set of parameters $a_1, \ldots, a_m$ to be determined in such a way that $G(i, a_1, \ldots, a_m) \approx f_i(n)$ or $G(i, a_1, \ldots, a_m) \approx f_i^s$.

In the implementation of Ben-Ari and Gal (1986) the parameters were determined by an iterative technique involving the solution of sets of simultaneous linear equations generated by simulation.

In a later implementation Kristensen (1992) determined the parameters by ordinary least squares regression on a simulated data set. The basic idea of the implementation is to take advantage from the fact that we are able to determine an optimal solution to the underlying (unrestricted) single-component model. If no herd restraint was present, the present value of the multi-component model would equal the sum of the present values of the individual animals determined from the underlying single-component model. Then it is argued in what way the restraint will logically reduce the (multi-component) present value, and a functional expression having the desired properties is chosen. The parameters of the function are estimated from a simulated data set, and the optimal action for a given (multi-component) state is determined as the one that maximizes the corrected present value. (A state in the multi-component model is defined from the states of the individual animals in the single-component model, and an action defines the replacement decision of each individual animal).

Ben-Ari and Gal (1986) compared the economic consequences of the resulting optimal multi-component policy to a policy defined by dairy farmers, and they showed that the policy from the parameter iteration method was better. Kristensen (1992) compared the optimal multi-component policies to policies from usual single-component models in extensive stochastic simulations and showed that the multi-component policies were superior in situations with shortage of heifers.

The parameter iteration method has been applied under a limited supply of heifers. It seems to be realistic to expect, that the method and the basic principles of Kristensen (1992) may be used under other kinds of herd restraints as for instance the short time adjustment to a milk quota as mentioned above.

13.6 General discussion

In Section 13.1, the main difficulties concerning animal production models were identified as *variability* in traits, *cyclic production*, *uniformity* (the traits are difficult to define and measure) and *herd restraints*. We are now able to conclude that the difficulties of variability and the cyclic production are directly solved by the application of Markov decision programming, but when the variability of several traits are included we face a problem of *dimensionality*. The formulation of the notion of a *hierarchical Markov process* contributed to the solution of the dimensionality problem, but did not solve it. The upper limit of number of states to be included has been raised considerably, but not eliminated.

This is for instance clearly illustrated when we formulate *multi-component* herd models in order to deal with herd restraints. In that case we still have to use approximate methods to determine an "optimal" replacement policy. On the other hand it

has been demonstrated by Kristensen (1992) that the *parameter iteration* method applied to a multi-component herd model (even though it is only approximate) is able to improve the total net revenue compared to the application of a usual single-component (animal) model in a situation with shortage of heifers. The parameter iteration method is an important contribution to the problem of determining optimal replacement policies under herd restraints.

In other situations with a limiting herd restraint it may be relevant to use an alternative criterion of optimality maximizing average net revenue per unit of the limiting factor. This method has been successfully applied in a situation with milk production under a limiting quota.

Recent results have also contributed to the solution of the uniformity problem. The Bayesian updating technique described in Section 13.4 has turned out to be a promising approach, which has been applied to several problems in farm management. It might also be a solution to the problem of including animal health as a trait to be considered. As concerns other traits such as litter size or milk yield the Bayesian approach may in some cases result in a reduction of the state space without loss of information (Jørgensen, 1992; Kristensen, 1993). Thus it contributes indirectly to the solution of the dimensionality problem.

New developments in the area of Markov decision programming for herd management support includes the notion of multi-level hierarchical models, where actions are defined at different levels with different time horizons. Thus the method is able to optimize decisions with different time horizons simultaneously as described by Kristensen and Jørgensen (2000). A good example of this technique is a model developed by Nielsen et al. (2004) for optimization of decisions concerning winter feeding level, summer grazing strategy, start of fattening and slaughtering of organic steers. An other example is the dairy cow replacement model by Bar et al. (2008b,a) who studied the effects of diseases using multi-level hierarchical models.

Also the development of a standard tool (the Java based MLHMP software Kristensen, 2003a) for (multi-level hierarchical) Markov decision processes is considered to be promising for construction of decision support models in the future.

Bibliography

Abramowitz, M., Stegun, I., 1964. Handbook of Mathematical Functions. Applied Mathematics Series, vol. 55. National Bureau of Standards, Washington.

Anderson, D., Weeks, D., 1989. Cattle liveweight sampled on a continuous versus intermittent basis. Livestock Production Science 23, 117–135.

Anonymous, 1993. Modern Heuristic techniques for combinatorial problems. Halsted Press.

Arnold, R. N., Bennet, G. L., 1991. Evaluation of four simulation models of cattle growth and body composition: Part i - comparison and characterization of the models. Agricultural Systems 35, 401–432.

Bangsø, O., 2004. Object oriented bayesian networks. Ph.D. thesis, Department of Computer Science, Aalborg University, Aalborg.

Bar, D., Tauer, L., Bennett, G., González, R., Hertl, J., Schulte, H., Schukken, Y., Welcome, F., Gröhn, Y., 2008a. Use of a dynamic programming model to estimate the value of clinical mastitis treatment and prevention options utilized by dairy producers. Agricultural Systems 99 (1), 6 – 12. DOI: 10.1016/j.agsy.2008.09.001

Bar, D., Tauer, L. W., Bennett, G., Gonzalez, R. N., Hertl, J. A., Schukken, Y. H., Schulte, H. F., Welcome, F. L., Grohn, Y. T., 2008b. The cost of generic clinical mastitis in dairy cows as estimated by using dynamic programming. J. Dairy Sci. 91 (6), 2205–2214. DOI: 10.3168/jds.2007-0573

Barnard, G., 1959. Control charts and stochastic processes. Journal of the Royal Statistical Society 21.

Bellman, R. E., 1957. Dynamic Programming. Princeton University Press, Princeton.

Bellman, R. E., 1961. Adaptive control processes: A guided tour. Princeton University Press, Princeton.

Bellman, R. E., Dreyfus, S. E., 1962. Applied dynamic programming. Princeton University Press, Princeton.

Bellman, R. E., Kalaba, R., 1965. Dynamic programming and modern control theory. Academic Press, New York.

Ben-Ari, Y., Amir, I., Sharar, S., 1983. Operational replacement decision model for dairy herds. Journal of Dairy Science 66, 1747–1759.

Ben-Ari, Y., Gal, S., 1986. Optimal replacement policy for multicomponent systems: An application to a dairy herd. European Journal of Operational Research 23, 213–221.

Besag, J., Green, P., Higdon, D., Mengersen, K., 1995. Bayesian computation and stochastic systems. Statistical Science 10, 3–66.

Bielza, C., Müller, P., Ríos Insua, D., 1999. Decision analysis by augmented probability simulation. Management Science 45, 995–1007.

Black, J. L., 1995. The evolution of animal growth models modelling energy metabolism in the pig - critical evaluation of a simple reference model. In: Moughan, P. J., Verstegen, M. W. A., Visserreyneveld, M. I. (Eds.), Modelling Growth in the Pig. Vol. 78 of Eaap Publication. Wageningen Peers, Wageningen.

Brandl, N., Jørgensen, E., 1996. Determination of live weight in pigs from dimensions by image analysis. Computers and Electronics in Agriculture 15, 57–72.

Bressers, H., te Brake, J., Jansen, M., Nijenhuis, P., Noordhuizen, J., 1994. Monitoring individual sows - radiotelemetrically recorded ear base temperature changes around farrowing. Livestock Production Science 37, 353–361.

Bressers, H., te Brake, J., Noordhuizen, J., 1991. Oestrus detection in group-housed sows by analysis of data on visits to the boar. Applied Animal Behaviour Science 31, 183–189.

Broekmans, J. E., 1992. Influence of price fluctuations on delivery strategies for slaughter pigs. Dina Notat 7, 1–28.

Buntine, W. L., 1994. Operations for learning with graphical models. Journal of Artificial Intelligence Research 2, 159–225.

Chang, W., Streeter, D., Jones, L., 1994. An object-oriented model for simulating milking parlor operations. Journal of Dairy Science 77, 84–93.

Charnes, J. M., Shenoy, P. P., 1996. A forward monte carlo method for solving influence diagrams using local computation (abstract). In: Proceedings of the First European Conference on Highly Structured Stochastic Systems. 19.-24. may 1996. Rebild, Denmark.

Chib, S., Greenberg, E., 1995. Understanding the metropolis-hastings algorithm. American Statistician 49, 327–335.

Clausen, S., 1969. Kybernetik - Systemer og Modeller. Institute of Mathematical Statistics and Operations Research. The Technical University of Denmark.

Cooper, W. W., Seiford, L. M., Tone, K., 2007. Data Envelopment Analysis. A comprehensive text with models, applications, references and DEA-solver software, 2nd Edition. Kluwer Academic Publishers.

Cornou, C., 2006. Automated oestrus detection methods in group housed sows: Review of the current methods and perspectives for development. Livestock Science 105, 1–11.

Cornou, C., Heiskanen, T., 2007. Automated oestrus detection method for group housed sows using acceleration measurements. In: ECPLF 2007. Proceedings of the 3rd European Conference on Precision Livestock Farming. Skiathos, Greece, pp. 211–217.

Cornou, C., Lundbye-Christensen, S., 2008. Classifying sows' activity types from acceleration patterns. an applicaqtion of the multi-process kalman filter. Applied Animal Behaviour Science 111, 262–273.

Cornou, C., Lundbye-Christensen, S., 2010. Classification of sows' activity types from acceleration patterns using univariate and multivariate models. Computers and Electronics in Agriculture 72 (2), 53 – 60. DOI: 10.1016/j.compag.2010.01.006

Cornou, C., Vinther, J., Kristensen, A. R., 2008. Automatic detection of oestrus and health disorders using data from electronic sow feeders. Livestock Science 118, 262–271.

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., Spiegelhalter, D. J., 1999. Probabilistic Networks and Expert Systems. Statistics for Engineering and Information Science. Springer-verlag, New York, Berlin, Heidelberg.

Danfær, A., 1990. A dynamic model of nutrient digestion and metabolism in lactating dairy cows. Beretning fra Statens Husdyrbrugsforsøgs 671, National Institute of Animal Science.

de Mol, R., Keen, A., Kroeze, G., 1999. Description of a detection model for oestrus and diseases in dairy cattle based on time series analysis with a kalman filter. Computers and Electronics in Agriculture 22, 171–185.

de Roo, G., 1987. A stochastic model to study breeding schemes in a small pig population. Agricultural Systems 25, 1–25.

de Vries, A., 2001. Statistical process control charts applied to dairy herd reproduction. Ph.D. thesis, University of Minnesota.

de Vries, A., Conlin, B. J., 2003. Design and performance of statistical process control charts applied to estrus detection efficiency. Journal of Dairy Science 86, 1970–1984.

de Vries, A., Conlin, B. J., 2005. A comparison of the performance of statistical quality control charts in a dairy production system through stochastic simulation. Agricultural Systems 82, 317–341.

de Vries, A., Reneau, J. K., 2010. Application of statistical process control charts to monitor changes in animal production systems. Journal of Animal Science 88, E11–24. DOI: 10.2527/jas.2009-2622

de Vries, A., van Leeuwen, J., Thatcher, W. W., 2004. Economic importance of improved reproductive performance. In: Proceedings 2004 Florida Dairy Reproduction Road Show. Okeechobee, Ocala and Chipley, Florida, pp. 33–43.

DeGroot, M. H., 1970. Optimal statistical decisions. McGraw-Hill, Inc,, New York.

DeLorenzo, M. A., Spreen, T. H., Bryan, G. R., Beede, D. K., van Arendonk, J. A. M., 1992. Optimizing model: insemination, replacement, seasonal variation, and cash flow. Journal of Dairy Science 75, 885–896.

Dempster, A., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the em algorithm (with discussion). Journal of The Royal Statistical Society: Series B (Statistical Methodology) 39, 1–38.

Dent, J., Anderson, J., 1971. System analysis in agricultural management. Wiley.

Dent, J. B., Harrison, S. R., Woodford, K. B., 1986. Farm planning with linear programming: Concept and Practice. Butterworths, Sydney.

Dijkhuizen, A., Krabbenborg, R., Huirne, R., 1989. Sow replacement: A comparison of farmer's actual decisions and model recommendations. Livestock Production Science 23, 207–218.

Dindorph, U., 1992. Agricultural applications of knowledge based system concepts - exemplified by a prototype on weed control in organic farming. WEEDOF. Ph.D. thesis, Tidsskrift for planteavls Sprecialserie Beretning nr S 2201.

Déry, R., Landry, M., Banville, C., 1993. Revisiting the issue of model validation in or: An epistemological view. European Journal of Operations Research 66, 168–183.

Durbin, J., Koopman, S. J., 2001. Time series analysis by state space methods. Oxford Statistical Science Series. Oxford University Press, Oxford.

Enevoldsen, C., Sorensen, J. T., Thysen, I., Guard, C., Grohn, Y. T., 1995. A diagnostic and prognostic tool for epidemiologic and economic analyses of dairy herd health management. Journal of Dairy Science 78 (4), 947–961.

Finlayson, J., Cacho, O., Bywater, A. C., 1995. A simulation model of grazing sheep: I. animal growth and intake. Agricultural Systems 48, 1–25.

Firk, R., Stamer, E., Junge, W., Krieter, J., 2002. Automation of oestrus detection in dairy cows: a review. Livestock Production Science 75 (219-232).

Ge, L., Kristensen, A. R., Mourits, M. C., Huirne, R. B., 2010a. A new decision support framework for managing foot-and-mouth disease epidemics. Annals of Operations Research Online First, ??–?? DOI: 10.1007/s10479-010-0774-2

Ge, L., Mourits, M. C., Kristensen, A. R., Huirne, R. B., 2010b. A modelling approach to support dynamic decision-making in the control of fmd epidemics. Preventive Veterinary Medicine 95 (3-4), 167 – 174. DOI: 10.1016/j.prevetmed.2010.04.003

Giaever, H. B., 1966. Optimal dairy cow replacement policies. Ph.D. thesis, University of California, Berkeley, University Microfilms, Ann Arbor, Michigan.

Gilks, W., 1992. Derivative-free adaptive rejection sampling for gibbs sampling. In: Bernardo, J., Berger, J., Dawid, A., Smith, A. (Eds.), Bayesian Statistics 4. Oxford University Press, pp. 641–649.

Gilks, W., Thomas, A., Spiegelhalter, D. J., 1993. A language and program for complex bayesian modelling. The Statistician 43, 169–178.

Gilks, W., Wild, P., 1992. Adaptive rejection sampling for gibbs sampling. Applied Statistics 41(2), 337–348.

Gonzales-Alcorta, M. J., Dorfman, J. H., Pesti, G. M., 1994. Maximizing profit in broiler production as price change: A simple approximation with practical value. Agribusiness 10 (5), 389–399.

Goodall, E. A., Sprevak, D., 1985. A bayesian estimation of the lactation curve of a dairy cow. Animal Production 40, 189–193.

Greve, J., 1995. The dynamic ranking approach and its application to piglet production. Dina research report, 35, Faculty of Technology and science. Aalborg University., Aalborg.

Hadley, G., 1964. Nonlinear and dynamic programming. Vol. Reading, Massachusetts. Addison-Wesley.

Hansen, A. E., 1992. Styret vækst hos slagtesvin ved ad libitum fodring. Master thesis, Department of Animal Science and Animal Health, The Royal Veterinary and Agricultural University, København.

Hansen, M. S., 2006. Optimization of delivery strategies for slaughter pigs. Master's thesis, Department of Large Animal Sciences, The Royal Veterinary and Agricultural University.

Haran, P., 1997. Markov decision processes in the optimisation of culling decisions for irish dairy herds. Master's thesis, School of Computer Applications, Dublin City University.

Hardie, A., 1996. Strategic modelling in dairy production. Ph.D. thesis, Department of Mathematical Statistics and Operational Research, University of Exeter.

Hayes-Roth, F., Waterman, D. A., Lenat, D. B., 1996. Building Expert Systems. Addison-wesley.

Henry, G. M., DeLorenzo, M. A., Beede, D. K., van Horn, H. H., Moss, C. B., Boggess, W. G., 1995. Determining the optimal nutrient management strategies for dairy farms. Journal of Dairy Science 78, 693–703.

Hillier, F. S., Lieberman, G. J., 1996. Introduction to Operations Research, 3rd Edition. Holden-Day Inc., San Francisco.

Hoel, P., Port, S., Stone, C., 1972. Introduction to Stochastic Processes. Waveland Press, Inc., Illinois.

Houben, E., Huirne, R. B. M., Dijkhuizen, A. A., Schreinemakers, J. F., 1995. Genetic algorithm to link decisions at animal and herd level on dairy farms. In: Houben, E. H. P. (Ed.), Economic optimization of decisions with respect to dairy cow health management. Department of Farm Management, Wageningen Agricultural University, pp. 101–121.

Houben, E. H. P., Huirne, R. B. M., Dijkhuizen, A. A., Kristensen, A. R., 1994. Optimal replacement of mastitis cows determined by a hierarchic markov process. Journal of Dairy Science 77, 2975–2993.

Howard, R., 1960. Dynamic Programming and Markov Processes. The M.I.T. Press, Cambridge, Massachusetts.

Howard, R. A., 1971. Dynamic probabilistic systems. Volume II: Semi-Markov and decision processes. John Wiley & Sons, Inc., New York.

Howard, R. A., Matheson, J. E., 1981. Influence diagrams. In: Howard, R. A., Matheson, J. E. (Eds.), Readings on the Principles and Applications of Decision Analysis. Strategic Decisions Group, Menlo Park, California, pp. 719–762.

Huirne, R., 1990. Computerized Management Support for Swine Breeding Farms. Ph.D. thesis. Department of Farm Management, Agriculturel University , P.O.box 338,6700 AH Wageningen. The Netherlands, Wageningen.

Huirne, R., Dijkhuizen, A., Renkema, J. A., 1991. Economic optimization of sow replacement decisions on the personal computer by method of stochastic dynamic programming. Livestock Production Science 28, 331–347.

Huirne, R., Hendriks, T., Dijkhuizen, A., Giesen, G., 1988. The economic optimisation of sow replacement decisions by stochastic dynamic programming. Journal of Agricultural Economics 39, 426–438.

Huirne, R., van Beek, P., Hendriks, T., Dijkhuizen, A., 1993. An application of stochastic dynamic programming to support sow replacement decisions. European Journal of Operational Research 67, 161–171.

Ingber, A., 1993. Simulated annealing: Practice versus theory. J. Mathl. Comput. Modelling. 18 (11), 29–57.

Jalvingh, A. W., Dijkhuizen, A. A., van Arendonk, J. A. M., Brascamp, E. W., 1992a. Dynamic probabilistic modelling of reproductive and replacement in sow herds. general aspects and model description. Agricultural Systems 39, 133–152.

Jalvingh, A. W., Dijkhuizen, A. A., van Arendonk, J. A. M., Brascamp, E. W., 1992b. An economic comparison of management strategies on reproduction and replacement in sow herds using a dynamic probabilistic model. Livestock Production Science 32, 331–350.

Jenkins, K. B., Halter, A. N., 1963. A multistage stochastic decision model: Application to replacement of dairy cows. Technical Bulletin 67, Agricultural Experimental Station, Oregon State University.

Jensen, A. L., 1995. A probabilistic model based decision support system for mildew management in winter wheat. Dina research report 39, University of Aalborg.

Jensen, F., Jensen, F. V., Dittmer, S. L., 1994. From influence diagrams to junction trees. In: de Mantaras, R. L., Poole, D. (Eds.), Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers, San Francisco, CA, pp. 367–373.

Jensen, F. V., 1996. An Introduction to Bayesian Networks. UCL Press, London.

Jensen, F. V., 2001. Bayesian Networks and Decision Graphs. Statistics for Engineering and Information Science. Springer-Verlag, New York.

Jewell, W., 1963. Markov renewal programming i and ii. Operations Research 11, 938–971.

Johnson, N., Leone, F., 1974. Statistics and Experimental Design, Second Edition, Volume 1. John Wiley & Sons, Inc.

Johnston, J., 1965. Dynamic programming and the theory of the firm. Journal of Agricultural Economics 16, 532–543.

Jørgensen, E., 1992. Sow replacement: Reduction of state space in dynamic programming models and evaluation of benefit from using the model. Dina Research Report 6.

Jørgensen, E., August 1993. The influence of weighing precision on delivery decisions in slaughter pig production. Acta Agriculturæ Scandinavica, Section A 43 (3), 181–189.

Jørgensen, E., 1985. Principper bag produktionskontrol i sohold. Phd thesis, Department of Animal Science, The Royal Veterinary and Agricultural University, Copenhagen, in Danish.

Jørgensen, E., 1991. Probabilistic growth model for pigs. paper presented at dina-workshop 1991. ebeltoft 3-4 december. dupl.

Jørgensen, E., 1992. Sow replacement: Reduction of state space in dynamic programming model and evaluation of benefit from using the model. Dina Research Report 6, –21.

Jørgensen, E., 1993. The influence of weighing precision on delivery decisions in slaughter pig production. Acta Agriculturae Scandinavica Section A - Animal Science 43, 181–189.

Jørgensen, E., Kristensen, A. R., 1995. An object oriented simulation model of a pig herd with emphasis on information flow. In: FACTS 95 March 7, 8, 9. Farm Animal Computer Technologies Conference. Orlando, Florida, pp. 206–215.

Kaelbling, L. P., Littman, M. L., Cassandra, A. R., 1998. Planning and acting in partially observable stochastic domains. Artificial Intelligence 101 (1–2), 99–134.

Kalman, R., 1960. A new approach to linear filtering and prediction problems. Journal of Basic Engineering 82.

Kass, R., Raftery, A., 1995. Bayes factors. J. Amer. Statist. Assn. 90 (430), 773–795.

Kennedy, J. O. S., 1986. Dynamic Programming Applications to Agriculture and Natural Resources. Elsevier Applied Science Publishers, London and New York.

Killen, L., Kearney, B., 1978. Optimal dairy cow replacement policy. Irish Journal of Agricultural Economics and Rural Sociology 7, 33–40.

Kjærulff, U., 1995. dhugin: A computational system for dynamic time-sliced bayesian networks. International Journal of Forecasting 11, 89–111.

Korsgaard, I. R., Løvendahl, P., Sloth, K. H., 2005. Monitoring daily measurements of somatic cell count: Automatic and on-line detection of mastitis. In:

Hogeveen, H. (Ed.), Mastitis in Dairy Production: Current knowledge and future solutions. Wageningen Academic Publishers, pp. 611–616.

Kristensen, A. R., 1987. Optimal replacement and ranking of dairy cows determined by a hierarchic markov process. Livestock Production Science 16, 131–144.

Kristensen, A. R., 1988. Hierarchic markov processes and their applications in replacement models. European Journal of Operational Research 35, 207–215.

Kristensen, A. R., 1989. Optimal replacement and ranking of dairy cows under milk quotas. Acta Agriculturæ Scandinavica 39, 311–318.

Kristensen, A. R., 1991. Maximization of net revenue per unit of physical output in markov decision processes. European Review of Agricultural Economics 18, 231–244.

Kristensen, A. R., 1992. Optimal replacement in the dairy herd: A multicomponent system. Agricultural Systems 39, 1–24.

Kristensen, A. R., 1993. Bayesian updating in hierarchic markov processes applied to the animal replacement problem. European Review of Agricultural Economics 20, 223–239.

Kristensen, A. R., 1994. A survey of markov decision programming techniques applied to the animal replacement problem. European Review of Agricultural Economics 21, 73–93.

Kristensen, A. R., 2003a. A general software system for markov decision processes in herd management applications. Computers and Electronics in Agriculture 38, 199–215.

Kristensen, A. R., 2003b. Information from on-line live weight assessment for optimal selection of slaughter pigs for market. In: Harnos, H., Herdon, M., Wiwczaroski, T. B. (Eds.), Proceedings EFITA 2003 conference. European Federation for Information Technology in Agriculture, Food and Environment, Debrecen, Hungary.

Kristensen, A. R., Jørgensen, E., 2000. Multi-level hierarchic markov processes as a framework for herd management support. Annals of Operations Research. 94, 69–89.

Kristensen, A. R., Nielsen, L., Nielsen, M. S., 2012. Optimal slaughter pig marketing with emphasis on information from on-line live weight assessment. Livestock Science 145 (1-3), 95 – 108. 10.1016/j.livsci.2012.01.003

Kristensen, A. R., Pedersen, C. V., 2003. Representation of uncertainty in a monte carlo simulation model of a scanvenging chicken production system. In: EFITA

2003. Fourth European Conference of the European Federation for Information Technology in Agriculture, Food and the Environment. Debrecen, Hungary, pp. 451–459.

Kristensen, A. R., Søllested, T. A., 2004a. A sow replacement model using bayesian updating in a 3-level hierarchic markov process. i. biological model. Livestock Production Science 87, 13–24.

Kristensen, A. R., Søllested, T. A., 2004b. A sow replacement model using bayesian updating in a 3-level hierarchic markov process. ii. optimization model. Livestock Production Science 87, 25–36.

Kristensen, A. R., Østergaard, V., 1982. Optimalt udskiftningstidspunkt for malkekoen fastlagt ved en stokastisk model. Beretning fra Statens Husdyrbrugsforsøg 533, National Institute of Animal Science.

Kristensen, A. R., Thysen, I., 1991. Economic value of culling information in the presence and absence of a milk quota. Acta Agriculturæ Scandinavica 41, 129–135.

Kure, H., 1997. Marketing management support in slaughter pig production. Ph.D. thesis, The Royal Veterinary and Agricultural University.

Lauritzen, S. L., 1995. Probabilistic expert systems. Tech. Rep. R-95-2013, Aalborg University. Insititute for Electronic Systems. Department of Mathematics and Computer Science., Aalborg.

Lauritzen, S. L., Nilsson, D., 2001. Limids of decision problems. Management Science 47, 1235–1251.

Lauritzen, S. L., Spiegelhalter, D. J., 1988. Local computations with probabilities on graphical structures and their application to expert system (with discussion). Journal of the Royal Statististical Society B 50, 157–224.

Lien, G., Kristensen, A. R., Hegrenes, A., Hardaker, J. B., 2003. Optimal length of leys in an area with winter damage problems. Grass and Forage Science 58, 168–177.

Liu, J., 1996. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. Statistics and Computing 6, 113–119.

Lovejoy, W. S., 1991. A survey of algorithmic methods for partially observed markov decision processes. Annals of Operations Research 28 (1-4), 47–66.

Lucas, J., 1976. The design and use of cumulative sumquality control schemes. Journal of Quality Technology 8.

Madsen, T. N., 2001. Tools for monitoring growing pigs. Ph.D. thesis, The Royal Veterinary and Agricultural University.

Madsen, T. N., Andersen, S., Kristensen, A. R., 2005. Modelling the drinking patterns of young pigs using a state space model. Computers and Electronics in Agriculture 48, 39–62.

Madsen, T. N., Kristensen, A. R., 2005. A model for monitoring the condition of young pigs by their drinking behaviour. Computers and Electronics in Agriculture 48, 138–154.

Madsen, T. N., Ruby, V., 2000. An application for early detection of growth rate changes in the slaughter pig production unit. Computers and Electronics in Agriculture 25, 261–270.

Makulska, J., Kristensen, A. R., 1999. Economic optimization of bull fattening. In: Perspectives of Modern Information and Communication Systems in Agriculture, Food Production and Environmental Control. European Federation for Information Technology in Agriculture, Food and the Environment, pp. 443–449.

Marchant, J. A., Schofield, C. P., Jun. 1993. Extending the snake image processing algorithm for outlining pigs in scenes. Computers and Electronics in Agriculture 8 (4), 261–275.

Mayer, D., Belward, J., Burrage, K., 1996. Use of advanced techniques to optimize a multi-dimensional dairy model. Agricultural Systems 50, 239–253.

McAinsh, C. V., Kristensen, A. R., 2004. Dynamic modelling of a traditional african chicken production system. Tropical Animal Health and Production 36, 609–626.

McArthur, A. T. G., 1973. Application of dynamic programming to the culling decision in dairy cattle. Proceedings of the New Zealand Society of Animal Production 33, 141–147.

McCullough, D. A., DeLorenzo, M. A., 1996a. Effects of price and management level on optimal replacement and insemination decisions. Journal of Dairy Science 79, 242–253.

McCullough, D. A., DeLorenzo, M. A., 1996b. Evaluation of a stochastic dynamic replacement and insemination model for dairy cattle. Journal of Dairy Science 79, 50–61.

McKendrick, I. J., Gettinby, G., Gu, Y., Reid, S. W. J., Revie, C. W., 2000. Using a bayesian belief network to aid differential diagnosis of tropical bovine diseases. Preventive Veterinary Medicine 47, 141–156.

Montgomery, D. C., 2005. Introduction to statistical quality control, 5th Edition. John Wiley & Sons, Inc.

Morris, C., Normand, S., 1992. Hierarchical models for combining information and for meta-analysis. In: Bernardo, J., Berger, J., Dawid, A., Smith, A. (Eds.), Bayesian Statistics 4. Oxford University Press, pp. 321–344.

Mourits, M. C. M., Huirne, R. B. M., Dijkhuizen, A. A., Kristensen, A. R., Galligan, D. T., 1999. Economic optimization of dairy heifer management decisions. Agricultural Systems 61, 17–31.

Nash, D. L., Rogers, G. W., 1995. Herd sire portfolio selection: A comparison of rounded linear and integer programming. Journal of Dairy Science 78, 2486–2495.

Nielsen, B. K., Kristensen, A. R., Thamsborg, S. M., 2004. Optimal decisions in organic steer production - a model including winter feeding level, grazing strategy and slaughtering policy. Livestock Production Science 88, 239–250.

Nielsen, L. R., Jørgensen, E., Højsgaard, S., 2011. Embedding a state space model into a markov decision process. Annals of Operations Research 190, 289–309. DOI: 10.1007/s10479-010-0688-z

Nielsen, L. R., Jørgensen, E., Kristensen, A. R., Østergaard, S., 2010. Optimal replacement policies for dairy cows based on daily yield measurements. Journal of Dairy Science 93 (1), 75 – 92. DOI: 10.3168/jds.2009-2209

Norberg, E., Korsgaard, I. R., Sloth, K. H. M. N., Løvendahl, P., 2008. Time-series models on somatic cell score improve detection of mastitis. Acta Agriculturae Scandinavica, Section A - Animal Sciences 58, 165–169.

Østergaard, S., Chagunda, M., Friggens, N., Bennedsgaard, T., Klaas, I., 2005. A stochastic model simulating pathogen-specific mastitis control in a dairy herd. Journal of Dairy Science 88, 4243–4257.

Østergaard, S., Sørensen, J. T., Hindhede, J., Houe, H., 2004. Control strategies against milk fever in dairy herds evaluated by stochastic simulation. Livestock Production Science 86, 209–223.

Østergaard, S., Sørensen, J. T., Kristensen, A. R., 2000. A stochastic model simulating the feeding-health-production complex in a dairy herd. Journal of Dairy Science 83, 721–733.

Ostersen, T., Cornou, C., Kristensen, A., 2010. Detecting oestrus by monitoring sows' visits to a boar. Computers and Electronics in Agriculture 74, 51 – 58. DOI: 10.1016/j.compag.2010.06.003

Otto, L., Kristensen, C. S., 2004. A biological network describing infection with mycoplasma hyopneumoniae in swine herds. Preventive Veterinary Medicine 66, 141–161.

Page, E., 1954. Continuous inspection schemes. Biometrika 41, 100–114.

Pearl, J., 1988. Probabilistic reasoning in intelligent systems. Morgan Kaufman Publishers, San Mateo.

Pedersen, B., Ruby, V., Jørgensen, E., 1995. Organisation and application of research and development in commercial pig herds: The Danish approach. In: Hennesy, D., Cranwell, P. (Eds.), Manipulating Pig Production V. Proceedings of the Fifth Biennial Conference of the Australasien Pig Science Association (APSA). Canberra, November 26 to 29, 1995.

Pedersen, C., 1996. Kan det betale sig at analysere grovfoder. [is there a pay-off from roughage analysis], student report in the course "Advanced Herd Management".

Pettigrew, J., Cornelius, S., Eidman, S., Moser, R., 1986. Integration of factors affecting sow efficiency: A modelling approach. Journal of Animal Science 63, 1314–1321.

Press, W. H., Flannery, B. R., Teukolsky, S. A., Vetterling, W. T., 1989. Numerical Recipes in Pascal. The art of scientific computing. Cambridge University Press, Cambridge.

Puterman, M. L., 1994. Markov Decision Processes: discrete stochastic dynamic programming. Wiley - Interscience.

Rae, A. N., 1994. Agricultural Management Economics - Activity Analysis and Decision Making. CAB International.

Raftery, A., Givens, G., Zeh, J., 1995. Inference from a deterministic population dynamics model for bowhead whales. Journal of the American Statistical Association. 90 (430), 402–430.

Ramaekers, P., Huiskes, J., Verstegen, M., den Hartog, L., Vesseur, P., Swinkels, J., 1995. Estimating individual body weights of group-housed growing-finishing pigs using a forelegs weighing system. Computers and Electronics in Agriculture 13, 1–12.

Reenberg, H., 1979. Udskiftning af malkekøer. En stokastisk udskiftningsmodel. Memorandum 6, Jordbrugsøkonomisk Institut.

Reeves, C. (Ed.), 1995. Modern heuristic techniques for combinatorial problems. McGraw-Hill, London.

Ross, S. M., 1970. Applied probability models with optimization applications. Holden-Day, San Francisco, California.

Rubinstein, R., Shapiro, A., 1993. Discrete Event Systems. Sensitivity analysis and stochastic optimisation by the score function method. John Wiley & Sons, Chichester.

Sargent, R. G., 1998. Verification and validation of simulation models. In: Medeiros, D. J., Watson, J. S., Carson, J. S., Manivannan, M. S. (Eds.), Proceedings of the 1998 winter simulation conference. Washington DC, pp. 121–130.

Scholten, H., Udink ten Cate, A., Eradus, W., Geurts, P., Glas, C., Miedema, R., 1995. Optimization of a fuzzy inference system for estrus detection in cattle. In: Udink ten Cate, A., Martin-Clouaire, R., Dijkhuizen, A., Lokhorst, C. (Eds.), 2nd IFAC/IFIP/EurAgEng Workshop on AI in Agriculture, Wageningen, May 29-31 1995. Wageningen, pp. 103–108.

Settimi, R., Smith, J. Q., 2000. A comparison of approximate bayesian forecasting methods for non-gaussian time series. Journal of Forecasting 19, 135–148.

Shachter, R. D., 1986. Evaluating influence diagrams. Operations Research 34, 871–882.

Shachter, R. D., Kenley, C., 1989. Gaussian influence diagrams. Management Science 35, 527–550.

Shenoy, P. P., 1992. Valuation-based systems for bayesian decision analysis. Operations Research 40, 463–484.

Singh, D., 1986. Simulation of swine herd population dynamics. Agricultural Systems 22, 157–183.

Skidmore, A. L., Beverly, R. W., 1995. Utilizing object-oriented analysis, design, and programming techniques to develop a dairy herd simulation program. In: FACTS 95 March 7, 8, 9. Farm Animal Computer Technologies Conference. Orlando, Florida, pp. 190–195.

Slader, R., Gregory, A., 1988. An automatic feeding and weighing system for ad libitum fed pigs. Computers and Electronics in Agriculture 3, 157–170.

Søllested, T. A., 2001. Automatic oestrus detection by modelling eating behaviour of group-housed sows in electronic sow feeding systems. Master's thesis, Royal Veterinary and Agricultural University.

Smith, B. J., 1971. The dairy cow replacement problem. An application of dynamic programming. Bulletin 745, Florida Agricultural Experiment Station, Gainesville, Florida.

Spiegelhalter, D., Thomas, A., Best, N., Gilks, W., 1996. BUGS. Bayesian Inference Using Gibbs Sampling. Version 0.5 (version ii). MRC Biostatistics Unit.

Sørensen, J., 1990. Validation of livestock herd simulation models: a review. Livestock Production Science 26, 79–90.

Sørensen, J. T., Kristensen, E. S., Thysen, I., 1992. A stochastic model simulating the dairy herd on a pc. Agricultural Systems 39, 177–200.

Stewart, H. M., Burnside, E. B., Pfeiffer, W. C., 1978. Optimal culling strategies for dairy cows of different breeds. Journal of Dairy Science 61, 1605–1615.

Stewart, H. M., Burnside, E. B., Wilton, J. W., Pfeiffer, W. C., 1977. A dynamic programming approach to culling decisions in commercial dairy herds. Journal of Dairy Science 60, 602–617.

Strang, G., King, J., 1970. Litter productivity in large white pigs. Animal Production 12, 235–243.

Sundgren, P.-E., van Male, J., Aumaitre, A., Kalm, E., Nielsen, H., 1979. Sow and litter recording procedures. report of a working party of the eaap commision on pig production. Livestock Production Science 7, 393–401.

Tatman, J. A., Shachter, R. D., 1990. Dynamic programming and influence diagrams. IEEE Transactions on Systems, Man, and Cybernetics 20, 365–379.

Tess, M. W., Bennett, G. L., Dickerson, G. E., 1983. Simulation of genetic changes in life cycle efficiency of pork production. i. a bioeconomic model. Journal of Animal Science 56, 336–353.

Thompson, P., Pulvermacher, R., Timms, L., 1995. Pedometer use for estrus detection. In: Proc. Animal Behavior and the Design of Livestock and Poultry Systems. International Conference. Indianapolis, Indiana, april 19-21. 1995. pp. 230–243.

Thysen, I., 1993a. Monitoring bulk tank somatic cell counts by a multi-process kalman filter. Acta Agriculturæ Scandinavica, Section A, Animal Science 43, 58–64.

Thysen, I., 1993b. Productivity control and decision support in dairy herds. In: Schiefer, G. (Ed.), Integrated systems in agricultural informatics. Bonn, pp. 149–158.

Thysen, I., Enevoldsen, C., 1994. Visual monitoring of reproduction in dairy herds. Preventive Veterinary Medicine 19, 189–202.

Thysen, I., Enevoldsen, C., Hindhede, J., Sørensen, J. T., 1988. Et system til styring af mælkeproduktion og reproduktion i kvægbesætningen. In: Østergaard, V., Hindhede, J. (Eds.), Studier i kvægproduktionssystemer. Vol. 649 of Beretning fra Statens Husdyrbrugsforsøg. København, pp. 15–57.

Toft, N., Jørgensen, E., 2002. Estimation of farm specific parameters in a longitudinal model for litter size with variance components and random dropout. Livestock Production Science 77, 175–185.

Toft, N., Kristensen, A. R., Jørgensen, E., 2005. A framework for decision support related to infectious diseases in slaughter pig fattening units. Agricultural Systems 85, 120–137.

Treacy, D., 1987. In: Proc. of th 6th Conf. Australian Ass. of Anim. Breed. & Gen. University of Western Australia, Perth, WA Australia, Dep. Agric. & Rural Affairs, box 125, Bendigo, Vic. 3550 Australia,, pp. 285–288.

Turner, M., Gurney, P., Belyavin, C., 1983. Automatic weighing of layer-replacement pullets housed on litter or in cages. British Poultry Science 24, 33–45.

van Arendonk, J. A. M., 1984. Studies on the replacement policies in dairy cattle. i. evaluation of techniques to determine the optimum time for replacement and to rank cows on future profitability. Zeitschrift für Tierzüchtung und Züchtungsbiologie 101, 330–340.

van Arendonk, J. A. M., 1985. Studies on the replacement policies in dairy cattle. ii. optimum policy and influence of changes in production and prices. Livestock Production Science 13, 101–121.

van Arendonk, J. A. M., 1986. Studies on the replacement policies in dairy cattle. iv. influence of seasonal variation in performance and prices. Livestock Production Science 14, 15–28.

van Arendonk, J. A. M., 1988. Management guides for insemination and replacement decisions. Journal of Dairy Science 71, 1050–1057.

van Arendonk, J. A. M., Dijkhuizen, A. A., 1985. Studies on the replacement policies in dairy cattle. iii. optimum policy and influence of changes in production and prices. Livestock Production Science 13, 333–349.

Van Bebber, J., Reinsch, N., Junge, W., Kalm, E., 1999. Monitoring daily milk yields with a recursive test day repeatability model (kalman filter). Journal of Dairy Science 82 (11), 2421–2429.

Van der Stuyft, E., Schofield, C., Randall, J., Wambacq, P., Goedseels, V., 1991. Development and application of computervision systems for uses in livestock production. Computers and Electronics in Agriculture 6, 243–265.

van der Wal, J., Wessels, J., 1985. Markov decision processes. Statistica Neerlandica 39, 219–233.

Verstegen, J. A. A. M., Sonnemans, J., Huirne, R. B. M., Dijkhuizen, A. A., Cox, J. C., 1998. Quantifying the effects of sow-herd management information systems on farmers' decision making using experimental economics. American Journal of Agricultural Economics 80, 821.

West, M., Harrison, J., 1997. Bayesian Forecasting and Dynamic Models, 2nd Edition. Springer Series in Statistics. Springer, New York.

West, M., Harrison, J. F., Migon, H. S., 1985. Dynamic generalized linear models and bayesian forecasting. Journal of the American Statistical Association 80, 73–83.

White, C. C. I., White, D. J., 1989. Markov decision processes. European Journal of Operational Research 39, 1–16.

Whittemore, C. T., Fawcett, R. H., 1976. Theoretical aspects of a flexible model to simulate protein and lipid growth in pigs. Animal Production 22, 87.

Williams, H. P., 1985. Model Building in Mathematical Programming. Wiley.

Wood, P. D. P., 1967. Algebraic model of the lactation curve in cattle. Nature 216, 164–165.

Appendix E

Transition matrices for Chapter 13

E.1 Probabilities for the actions “Keep” and “Replace”

Table E.1: Transition probabilities from state i to state j of the 36-state model under the action "Keep" ($j = 1, \ldots, 12$).

				1	2	3	4	5	6	7	8	9	10	11	12
				Bad genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	0	0	0	0.6	0.3	0.1	0	0	0	0	0	0
2	B	1	A	0	0	0	0.2	0.6	0.2	0	0	0	0	0	0
3	B	1	H	0	0	0	0.1	0.3	0.6	0	0	0	0	0	0
4	B	2	L	0	0	0	0	0	0	0.6	0.3	0.1	0	0	0
5	B	2	A	0	0	0	0	0	0	0.2	0.6	0.2	0	0	0
6	B	2	H	0	0	0	0	0	0	0.1	0.3	0.6	0	0	0
7	B	3	L	0	0	0	0	0	0	0	0	0	0.6	0.3	0.1
8	B	3	A	0	0	0	0	0	0	0	0	0	0.2	0.6	0.6
9	B	3	H	0	0	0	0	0	0	0	0	0	0.1	0.3	0.6
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	0	0	0	0	0	0	0	0	0	0	0	0
14	A	1	A	0	0	0	0	0	0	0	0	0	0	0	0
15	A	1	H	0	0	0	0	0	0	0	0	0	0	0	0
16	A	2	L	0	0	0	0	0	0	0	0	0	0	0	0
17	A	2	A	0	0	0	0	0	0	0	0	0	0	0	0
18	A	2	H	0	0	0	0	0	0	0	0	0	0	0	0
19	A	3	L	0	0	0	0	0	0	0	0	0	0	0	0
20	A	3	A	0	0	0	0	0	0	0	0	0	0	0	0
21	A	3	H	0	0	0	0	0	0	0	0	0	0	0	0
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	0	0	0	0	0	0	0	0	0	0	0	0
26	H	1	A	0	0	0	0	0	0	0	0	0	0	0	0
27	H	1	H	0	0	0	0	0	0	0	0	0	0	0	0
28	H	2	L	0	0	0	0	0	0	0	0	0	0	0	0
29	H	2	A	0	0	0	0	0	0	0	0	0	0	0	0
30	H	2	H	0	0	0	0	0	0	0	0	0	0	0	0
31	H	3	L	0	0	0	0	0	0	0	0	0	0	0	0
32	H	3	A	0	0	0	0	0	0	0	0	0	0	0	0
33	H	3	H	0	0	0	0	0	0	0	0	0	0	0	0
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, H = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, H = High.

Table E.2: Transition probabilities from state i to state j of the 36-state model under the action "Keep" ($j = 13, \ldots, 24$).

				13	14	15	16	17	18	19	20	21	22	23	24
				Average genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	0	0	0	0	0	0	0	0	0	0	0	0
2	B	1	A	0	0	0	0	0	0	0	0	0	0	0	0
3	B	1	H	0	0	0	0	0	0	0	0	0	0	0	0
4	B	2	L	0	0	0	0	0	0	0	0	0	0	0	0
5	B	2	A	0	0	0	0	0	0	0	0	0	0	0	0
6	B	2	H	0	0	0	0	0	0	0	0	0	0	0	0
7	B	3	L	0	0	0	0	0	0	0	0	0	0	0	0
8	B	3	A	0	0	0	0	0	0	0	0	0	0	0	0
9	B	3	H	0	0	0	0	0	0	0	0	0	0	0	0
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	0	0	0	0.6	0.3	0.1	0	0	0	0	0	0
14	A	1	A	0	0	0	0.2	0.6	0.2	0	0	0	0	0	0
15	A	1	H	0	0	0	0.1	0.3	0.6	0	0	0	0	0	0
16	A	2	L	0	0	0	0	0	0	0.6	0.3	0.1	0	0	0
17	A	2	A	0	0	0	0	0	0	0.2	0.6	0.2	0	0	0
18	A	2	H	0	0	0	0	0	0	0.1	0.3	0.6	0	0	0
19	A	3	L	0	0	0	0	0	0	0	0	0	0.6	0.3	0.1
20	A	3	A	0	0	0	0	0	0	0	0	0	0.2	0.6	0.6
21	A	3	H	0	0	0	0	0	0	0	0	0	0.1	0.3	0.6
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	0	0	0	0	0	0	0	0	0	0	0	0
26	H	1	A	0	0	0	0	0	0	0	0	0	0	0	0
27	H	1	H	0	0	0	0	0	0	0	0	0	0	0	0
28	H	2	L	0	0	0	0	0	0	0	0	0	0	0	0
29	H	2	A	0	0	0	0	0	0	0	0	0	0	0	0
30	H	2	H	0	0	0	0	0	0	0	0	0	0	0	0
31	H	3	L	0	0	0	0	0	0	0	0	0	0	0	0
32	H	3	A	0	0	0	0	0	0	0	0	0	0	0	0
33	H	3	H	0	0	0	0	0	0	0	0	0	0	0	0
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, H = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, H = High.

Table E.3: Transition probabilities from state i to state j of the 36-state model under the action "Keep" ($j = 25, \ldots, 36$).

				25	26	27	28	29	30	31	32	33	34	35	36
				High genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	0	0	0	0	0	0	0	0	0	0	0	0
2	B	1	A	0	0	0	0	0	0	0	0	0	0	0	0
3	B	1	H	0	0	0	0	0	0	0	0	0	0	0	0
4	B	2	L	0	0	0	0	0	0	0	0	0	0	0	0
5	B	2	A	0	0	0	0	0	0	0	0	0	0	0	0
6	B	2	H	0	0	0	0	0	0	0	0	0	0	0	0
7	B	3	L	0	0	0	0	0	0	0	0	0	0	0	0
8	B	3	A	0	0	0	0	0	0	0	0	0	0	0	0
9	B	3	H	0	0	0	0	0	0	0	0	0	0	0	0
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	0	0	0	0	0	0	0	0	0	0	0	0
14	A	1	A	0	0	0	0	0	0	0	0	0	0	0	0
15	A	1	H	0	0	0	0	0	0	0	0	0	0	0	0
16	A	2	L	0	0	0	0	0	0	0	0	0	0	0	0
17	A	2	A	0	0	0	0	0	0	0	0	0	0	0	0
18	A	2	H	0	0	0	0	0	0	0	0	0	0	0	0
19	A	3	L	0	0	0	0	0	0	0	0	0	0	0	0
20	A	3	A	0	0	0	0	0	0	0	0	0	0	0	0
21	A	3	H	0	0	0	0	0	0	0	0	0	0	0	0
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	0	0	0	0.6	0.3	0.1	0	0	0	0	0	0
26	H	1	A	0	0	0	0.2	0.6	0.2	0	0	0	0	0	0
27	H	1	H	0	0	0	0.1	0.3	0.6	0	0	0	0	0	0
28	H	2	L	0	0	0	0	0	0	0.6	0.3	0.1	0	0	0
29	H	2	A	0	0	0	0	0	0	0.2	0.6	0.2	0	0	0
39	H	2	H	0	0	0	0	0	0	0.1	0.3	0.6	0	0	0
31	H	3	L	0	0	0	0	0	0	0	0	0	0.6	0.3	0.1
32	H	3	A	0	0	0	0	0	0	0	0	0	0.2	0.6	0.6
33	H	3	H	0	0	0	0	0	0	0	0	0	0.1	0.3	0.6
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, A = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, A = High.

Table E.4: Transition probabilities from state i to state j of the 36-state model under the action "Replace" ($j = 1, \ldots, 12$).

				1	2	3	4	5	6	7	8	9	10	11	12
				Bad genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
2	B	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
3	B	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
4	B	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
5	B	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
6	B	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
7	B	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
8	B	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
9	B	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
14	A	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
15	A	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
16	A	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
17	A	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
18	A	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
19	A	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
20	A	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
21	A	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
26	H	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
27	H	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
28	H	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
29	H	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
30	H	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
31	H	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
32	H	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
33	H	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, H = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, H = High.

Table E.5: Transition probabilities from state i to state j of the 36-state model under the action "Replace" ($j = 13, \ldots, 24$).

				13	14	15	16	17	18	19	20	21	22	23	24
				Average genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
2	B	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
3	B	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
4	B	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
5	B	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
6	B	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
7	B	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
8	B	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
9	B	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
14	A	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
15	A	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
16	A	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
17	A	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
18	A	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
19	A	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
20	A	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
21	A	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
26	H	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
27	H	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
28	H	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
29	H	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
30	H	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
31	H	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
32	H	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
33	H	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, H = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, H = High.

Table E.6: Transition probabilities from state i to state j of the 36-state model under the action "Replace" ($j = 25, \ldots, 36$).

				25	26	27	28	29	30	31	32	33	34	35	36
				High genetic merit											
				1st lactation			2nd lactation			3rd lactation			4th lactation		
i	g_i	l_i	y_i	L	A	H	L	A	H	L	A	H	L	A	H
1	B	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
2	B	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
3	B	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
4	B	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
5	B	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
6	B	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
7	B	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
8	B	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
9	B	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
10	B	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
11	B	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
12	B	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
13	A	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
14	A	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
15	A	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
16	A	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
17	A	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
18	A	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
19	A	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
20	A	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
21	A	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
22	A	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
23	A	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
24	A	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
25	H	1	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
26	H	1	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
27	H	1	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
28	H	2	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
29	H	2	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
39	H	2	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
31	H	3	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
32	H	3	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
33	H	3	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
34	H	4	L	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
35	H	4	A	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0
36	H	4	H	1/9	1/9	1/9	0	0	0	0	0	0	0	0	0

Legends: i = State; g_i = genetic merit of state i indicated as B = Bad, A = Average, A = High; l_i = lactation number of state i; y_i = milk yield of state i indicated as L = Low, A = Average, A = High.