Algorithms for optimal risk-based planning of inspections using influence diagrams

Jesus Luque & Daniel Straub

Engineering Risk Analysis Group, Technische Universität München, Germany

jesus.luque@tum.de, straub@tum.de

Abstract: Risk-based optimization of inspection using influence diagrams is investigated. To this end, a fatigue deterioration model using a Dynamic Bayesian Network (DBN) approach is presented. The DBN incorporates information from previous inspection campaigns. Decision and utility nodes are defined inside the network to represent inspection and repair activities. The optimal inspection strategy (subject to safety or utility constraints) is approximated using the Limited Memory Influence Diagram (LIMID) approach, and is solved using the single policy updating, a local optimization strategy. In a numerical investigation, this method is found to give solutions that are slightly better than those obtained with simple heuristics that were previously applied, such the reliability threshold or periodic inspection heuristic. Finally, the numerical example demonstrates the superiority of adaptive inspection strategies, whereby inspections are planned based on the results of previous inspections.

Keywords: optimal inspections, Bayesian network, decision models, fatigue, influence diagrams.

1. Introduction

Deterioration processes, in particular fatigue and corrosion, lead to a reduction of the reliability of structural systems. Because deterioration processes are commonly associated with significant uncertainty, inspection and monitoring are often an effective means to increase the reliability. Based on the results of inspections, repair and replacement actions can be planned. This is known as condition-based maintenance 0.

The uncertainty in deterioration processes is commonly represented through probabilistic models, comprising of deterministic deterioration models whose parameters are represented by random variables. In order to assess the effect of different inspection and/or monitoring strategies, their expected costs, including the risk associated with potential failures, can be computed and compared. This is commonly known as risk-based planning of inspection and monitoring [9] and is a special case of the pre-posterior analysis of the Bayesian decision theory (0, 0). The computation of the expected costs for a given inspection strategy requires integration over the entire outcome space of all random variables in the deterioration model as well as over all possible inspection outcomes. This is a computationally demanding problem. In addition, to compute the expected cost it is also necessary to include (and optimize) the maintenance and repair actions into the analysis. Since the number of potential alternative inspection and monitoring strategies is very large, solving the complete optimization problem is thus computationally intractable for realistic applications. For this reason, different heuristics (e.g. 0, 0) have been developed in order to approximate the optimal solution, including periodic inspections (PI) and reliability threshold (RT). More recently, the use of the limited memory influence diagram (LIMID) was suggested by NIELSEN & SØRENSEN [5].

In this paper, we present and compare different algorithms for the optimal planning of inspections in a structural element subject to fatigue deterioration. The fatigue crack growth process is represented through a dynamic Bayesian network (DBN). The optimization parameters are the times of inspections and times of repair actions. This decision problem is modeled as an influence diagram. Besides the classical PI and RT heuristics, we investigate two alternative formulations of the problem as a LIMID. We find that the LIMID outperforms the classical approaches, but also has increased computational demands. However, the complexity of the LIMID algorithm is shown to be of similar order than PI and RT. It is thus a viable alternative, which is particularly promising for planning inspections in systems, where the number of decision alternatives is much larger and simple heuristics such as PI and RT are not available.

2. Dynamic Bayesian networks and influence diagrams

2.1 Bayesian networks

A Bayesian network (BN) is a probabilistic model. It consists of a set of random variables (nodes) and directed links which form a directed acyclic graph (DAG), i.e. there is no directed path from any variable to itself. A discrete BN furthermore fulfils the following requisites 0:

- Each variable has a finite domain.
- To each variable X with parents Y_1, Y_2, …, Y_N is attached a conditional probability table $p(x|y_1, y_2, \dots, y_N) = \Pr(X = x|Y_1 = y_1 \cap \dots \cap Y_N = y_N)$. Y_i is called a parent of X if it has a link towards X. If a variable has no parents, the table corresponds to its unconditional probability mass function (PMF).

In Fig. 1, exemplarily a simple BN representing the condition of a structural element before and after applying a load is shown. The condition of the element is represented by a_0 and a_1, the damage size (crack depth) before and after the application of load L_1, respectively. Variable Z_1 represents a possible inspection outcome of the condition a_1 in case an inspection is carried out. Nodes a_0 and L_1 are described by unconditional PMFs. The probability table of a_1 contains the PMFs of the damage size conditional on the previous damage size a_0 and the load L_1. The probability table attached to Z_1 describes the likelihood of the inspection outcome, i.e. the probability of an observation (e.g. detect damage) given the condition a_1.

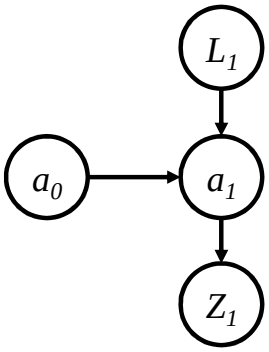

Fig. 1. Example of a Bayesian network

If the states of some variables are known (i.e. instantiated) in the BN, the PMFs of the remaining nodes can be updated to their posterior. For example, in the BN of Fig. 1, an inspection outcome Z_1 can be included by instantiating the corresponding node with the observed state z_1, e.g. no detection of a defect. The PMFs of the remaining nodes a_0, a_1,

and L_1 are then updated to their conditional PMF given z_1. This ability to efficiently perform Bayesian updating makes BNs suitable for modeling deterioration processes when partial observations from inspections and monitoring are to be included 0.

2.2Dynamic Bayesian networks

In some cases, BNs contain a repetitive sequence of nodes which are associated with multiple times or spatial locations. Such a BN is called dynamic Bayesian network (DBN) and is useful for modeling time-dependent processes, including structural deterioration 0. Extending the BN of Fig. 1 to multiple loads L_t, conditions a_t, and observations Z_t at times $t = 1, \dots, T$ the DBN shown in Fig. 2 is obtained.

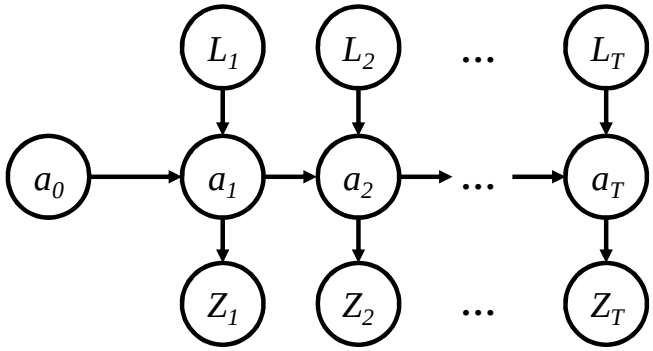

Fig. 2. Example of a dynamic Bayesian network

2.3Influence diagrams

BNs can be extended to influence diagrams (ID), which additionally include decisions and utility (cost). In the ID, decisions are shown as squared nodes and utilities as diamond-shaped nodes. In the latter, a utility value is assigned to each combination of states of the parents nodes, which can be either random variables or decision nodes, but not utility nodes. In case there are several utility nodes, the total utility is the sum of the individual utilities. In the ID, the optimal decision is the one that maximizes the total expected utility, in agreement with classical decision analysis 0.

The decision nodes describe different decision options, which influence the random variables that are children of the decision node. This influence is quantified through the conditional PMF of these child nodes. Links pointing towards the decision nodes represent the available information at the time of making the decision. All parents of the decision nodes are known when making the decision. However, there exist different versions of IDs, which differ in the way information is handled. Often, the ID is based on the no-forgetting assumption: When making a decision, all previous decisions as well as previous observations are known. This requires that there is a temporal ordering of the decisions. The no-forgetting assumption leads to significant computational demands. For this reason, the limited memory ID (LIMID) was introduced, which makes an explicit link between the nodes that are known before taking the decision and the decision node 0. In the LIMID, only the direct parents of a decision node are known at the time of making the decision. This reduces (or limits) the number of nodes that will be considered for the decision, decreases the size of the policy domain and facilitates the obtaining of the optimal strategy that gives the maximum expected utility. In this paper, we use LIMIDs to represent the inspection and repair decision processes.

Fig. 3 shows an example ID for the deterioration example presented earlier as a DBN in Fig. 2. Here, the decisions R_t are included on whether or not to repair the structural element at times $t = 1, \dots, T$. These decisions are made based on the result of the inspections Z_t, hence the links $Z_t \rightarrow R_t$. To differentiate the condition of the element before and after the repair, the nodes a_t' are introduced. The conditional PMF of these nodes are identical to that of a_t in case no repair is carried out, and they differ if a repair is carried out. The utilities $U_{R,t}$ are the (negative) cost of repairs and the utilities $U_{F,t}$ are the cost associated with failure at time t. The last slice does not include a repair decision, since such an action would be pointless at the end of the service life.

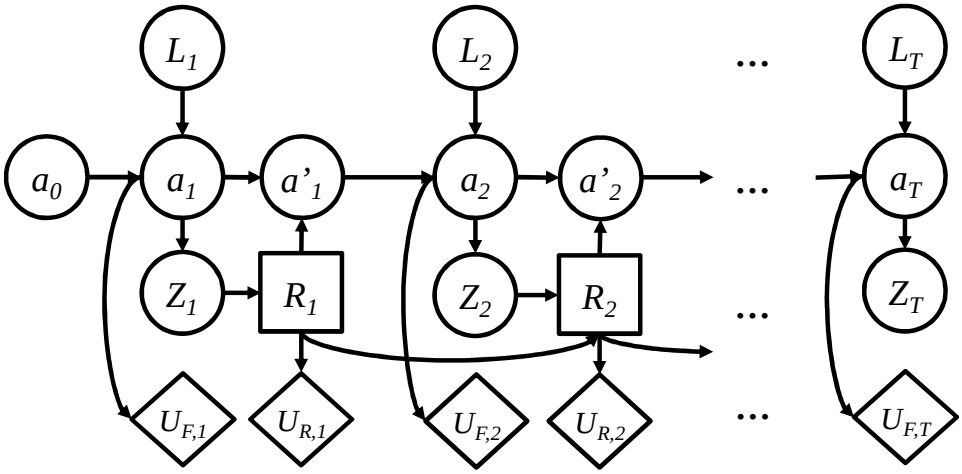

Fig. 3. ID of the multi-decision structure condition example

2.4Policies and strategies

In the ID, decisions are taken based on information available when making that decision. In the LIMID, these are the nodes with links pointing to the decision node. A policy consists of a set of rules defining which decision to take as a function of the available information. The more information is used for making a decision, the larger the policy domain and consequently the computational demand. A set of policies of all decision nodes in the ID is called a strategy.

3 Risk-based planning of inspections using influence diagrams for a structural element subject to fatigue

In condition-based maintenance of structures, it has to be decided when, where and how to inspect. Here we restrict ourselves to finding optimal decisions on when to inspect, and we present IDs to solve this problem. The optimal inspection strategy is defined as the one that minimizes the expected cost, defined as the sum of inspection, repair and failure cost. Note that the expected cost of failure is the risk.

For the numerical investigation, a structural element subject to fatigue deterioration is considered. Inspections are possible in each year of the service life, potentially followed by repair actions in case of adverse inspection outcomes.

3.1 Fatigue crack growth model

To model the fatigue crack growth, we consider a simplified case corresponding to crack growth in an infinite plate, described by Paris' law (e.g. 0):

$$\frac{da(n)}{dn} = C\left[\Delta S\sqrt{\pi a(n)}\right]^m \tag{1}$$

where a is the crack depth; n is the number of stress cycles; ΔS is the stress range per cycle with constant stress amplitudes; and C and m are empirically determined model parameters. Parameters ΔS, C, and m are modeled as time invariant random variables. Using the boundary condition $a(n=0) = a_0$, the previous equation leads to

$$a(n) = \left[\left(1-\frac{m}{2}\right)C\Delta S^m \pi^{m/2} n + {a_0}^{(1-m/2)}\right]^{(1-m/2)^{-1}} \tag{2}$$

In order to use a DBN for the fatigue model, the time is discretized in intervals of 1 year. If $n_t = n(t)$ is the number of cycles at time step t, then the crack depth at the end of each year can be expressed recursively as a function of the crack depth in the previous year as

$$a_t = \left[q\pi^{m/2} + {a_{t-1}}^{(1-m/2)}\right]^{(1-m/2)^{-1}} \tag{3}$$

where $q = \left(1-\frac{m}{2}\right)C\Delta S^m\,\Delta n$. Here, $\Delta n = n_t - n_{t-1}$ is the number of stress cycles per year. Variable q is defined in order to reduce the dimension of the variable space.

The failure event of the component is defined by the limit state function

$$g = a_c - a(n) \tag{4}$$

where a_c represents the critical crack depth. The condition of the component is indicated by the binary variable E_t, which takes value 1 when $g > 0$ (i.e. safe event) and 0 when $g \leq 0$ (i.e. failure event).

In STRAUB 0, a DBN model and algorithm was developed for this simple crack growth law, as presented in Fig. 4a. For the purpose of the present study, the model is simplified by eliminating the variables q_t and m_t, leading to a simple homogeneous Markovian deterioration model for a_t as shown in Fig. 4b. The resulting discrete Markov process for the crack depth a_t follows the recursive relation

$$\mathbf{p}_a(t) = \mathbf{A}\,\mathbf{p}_a(t-1) \tag{5}$$

where $t = 1,2,\ldots,T$, $\mathbf{A}$ is the transition probability matrix and $\mathbf{p}_a(t)$ is the vector describing the discretized probability distribution of a_t. $\mathbf{p}_a(0)$ is given by the probability distribution of a_0. The Markov model is homogenous if all random variables in the model are time-invariant. Note that the unconditional marginal distribution of the crack depth and the unconditional probability of failure of the two models in Fig. 4 are identical. However, as soon as observations are made, the conditional distributions computed with the two models will differ.

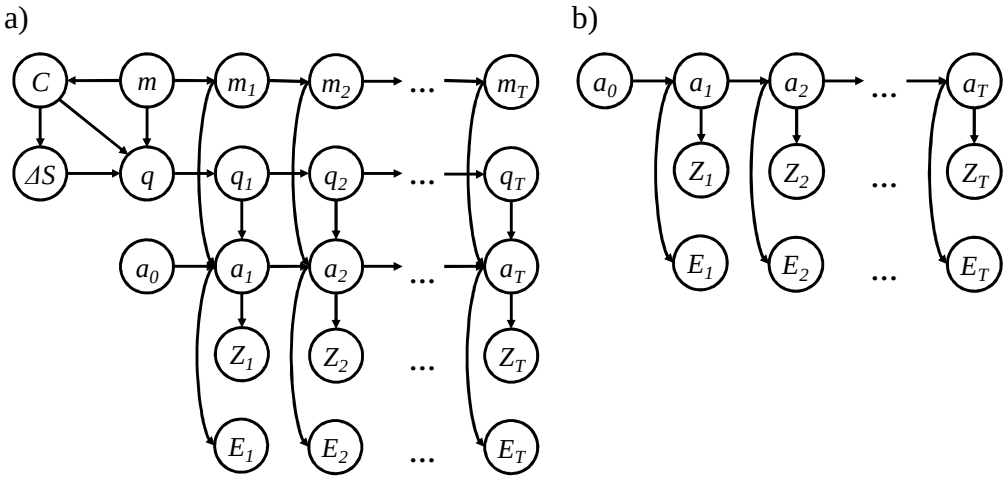

Fig. 4. DBN of the original (a) and simplified (b) fatigue crack growth model

3.2 Influence diagram for modeling inspections

To assess the effect of inspections and to determine optimal inspection times, the fatigue DBN of Fig. 4 is extended to an influence diagram (ID). The ID is presented in Fig. 5. The elements of this ID are introduced in the following.

Inspection decisions. At every time step t, a decision node D_t is included in the BN. Each decision has two possibilities: no inspection ($D_t = 0$) or inspection ($D_t = 1$). In the ID shown in Fig. 5, this decision node has no parents. Because we follow the LIMID convention described above, this implies that the inspection is planned without any previous knowledge. This assumption will later be relaxed.

Observations. In case an inspection is carried out at time t, the random variable Z_t will indicate if a crack was *detected* ($Z_t = 1$) or *not detected* ($Z_t = 2$). If no inspection was carried out (i.e. $D_t = 0$), then the corresponding state of the variable will be *no measurement* ($Z_t = 3$). In case an inspection is performed, the *probability of detection* (PoD) describes the probability of detecting the crack. It is a non-decreasing function of the crack depth (i.e. larger cracks have larger probabilities to be detected) and is here represented by the following relation 0:

$$\Pr(Z_t = 1 | a_t) = PoD(a_t) = 1 - \exp(-a_t / 10\, mm) \qquad (6)$$

Repairs. Repair actions are included in the model as a function of the observed conditions of the component and the system. Whether or not to repair at time t is in principle also a decision that may be optimized jointly with the inspection decision. However, it has been found that simple decision rules are sufficient for the repair action in the considered case, and no optimization is needed 0. The rule is that if the system fails or a crack is detected during an inspection, the component will be repaired. This is included in the ID through the variable a'_t. If no repair is carried out, the state of a'_t will be identical to a_t; if a repair is carried out, its conditional distribution is equal to the distribution of the initial crack depth a_0, assuming that the new condition is probabilistically identical to the original one.

Component and system condition. Structural systems are often redundant, so that failure of an element does not necessarily imply a system failure. Here, the redundancy of the system with respect to component failure is defined in a simplifying manner as the probability that the system does not fail when the component does. $E_{S,t}$ and $E_{C,t}$ denote the condition (i.e. failure or safe) of the system and the component, respectively, at time t. We define the redundancy r as:

$$\Pr\left(E_{S,t} = \text{safe} \middle| E_{C,t} = \text{failure}\right) = r \tag{7}$$

In the extreme case with no redundancy $r = 0$, element failure will directly lead to system failure. Similarly, if the system is fully redundant with respect to element failure, $r = 1$, then the system will not fail if only this element fails. This simple model does not account accurately for multiple element failures, which must addressed by an advanced model 0.

Utilities. The variables $E_{S,t}$ (system condition), D_t (inspection decision), and Z_t (observation) are associated with costs. These are modeled by the utility nodes $U_{S,t}$, $U_{I,t}$, and $U_{R,t}$. The utility of an inspection, repair and system failure events are $-C_I$, $-C_R$, and $-C_S$,.

3.3 Memory assumptions in the ID model

In the LIMID, only those nodes with links to the decision nodes are assumed to be known at the time of making the decision. This assumption can strongly reduce the computational effort when optimizing the decisions. With increasing memory, i.e. with increasing number of links to the decision nodes, the policy domains of the decision nodes increases, making the solution of the optimization problem intractable. On the other hand, reducing the number of information links toward the decision node leads to suboptimal solutions, in particular if compared to the no-forgetting assumption.

In the first ID presented in Fig. 5, it is assumed that no information is available when the inspection decision is made. We call this the *no-memory ID*. The advantage of the no-memory ID is that all inspections can be planned a-priori, since no observation during the service life will influence the inspection decisions.

Alternatively, we consider the ID presented in Fig. 6, where information from previous observations and decisions is taken into account when planning the inspections. In this ID, it is assumed that the observation made at the previous inspection is known when deciding upon inspection. Two additional variables, Z_t^* and τ_t, are included in the model and contain the observation from the last inspection and the time when it was performed. We call this ID the *last-inspection ID*.

3.4 Finding optimal inspection times with the ID

When solving decision problems, the size of the solution domain can quickly become intractable as the number of decision nodes increases. Depending on the type of application, some characteristics (e.g. symmetry) can be used to reduce the computational demands of the decision problem 0. Alternatively, approximate solutions can be obtained. In particular, s*ingle policy updating* (SPU) is an iterative algorithm for solving LIMIDs that runs over each decision node, obtaining its locally optimum policy that maximizes the expected utility of the decision problem by keeping the remaining policies fixed. An iteration is completed when all decision nodes are locally maximized and the algorithm stops when the next iteration does not further reduce the expected utility. Due to its local nature, the solution obtained with SPU is likely to be suboptimal.

For the inspection planning problem, simple heuristics were defined in the past to reduce the solution space (see e.g. 0). The two most common heuristics are summarized in the following.

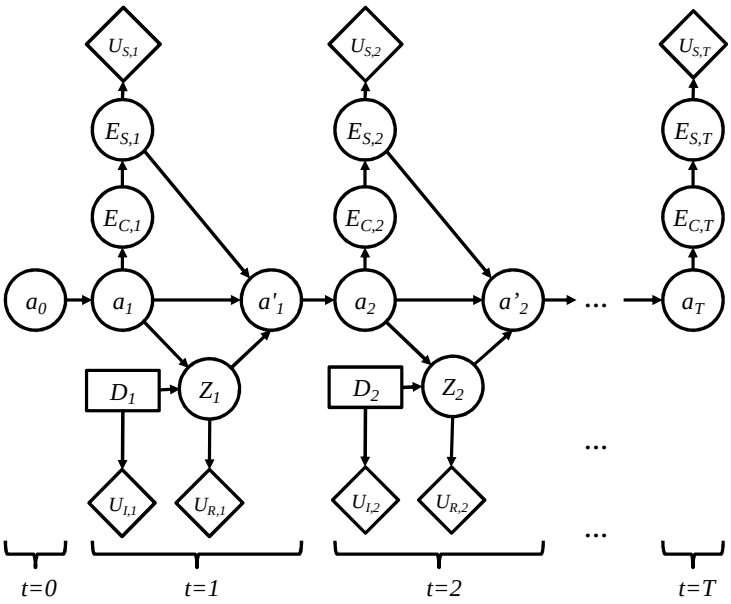

Fig. 5. ID modeling the fatigue inspection planning (no-memory ID).

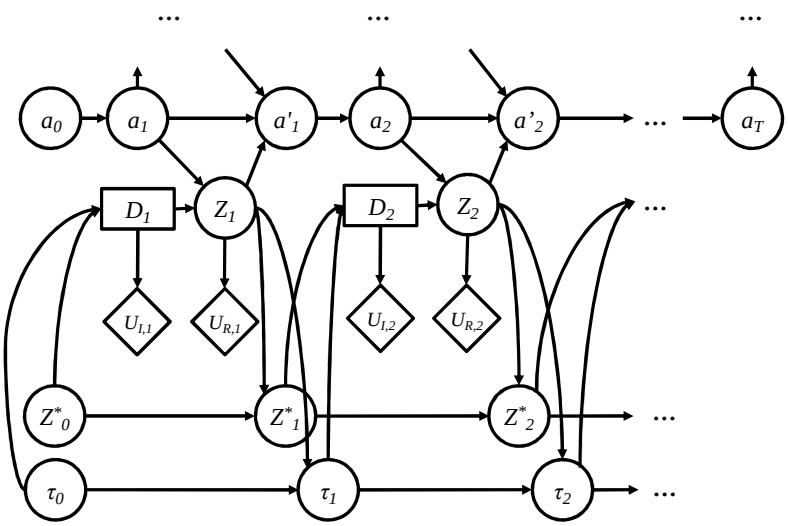

Fig. 6. ID modeling the fatigue inspection planning (last-inspection ID).

Periodic inspections (PI), also known as equidistant inspections: The number of possible inspection times of the decision problem increases exponentially with the number of time steps T (i.e. 2^T combinations). However, if it is required that the inspection intervals are fixed, then the number of possible combinations is reduced to T. The optimization problem can be formulated in terms of a single variable, the number of inspections n_p. If n_p periodic inspections are performed in T time steps, then the inspection times are

$$\left\lfloor \frac{T}{n_p+1} \right\rfloor, \left\lfloor \frac{2T}{n_p+1} \right\rfloor, \ldots, \left\lfloor \frac{n_p T}{n_p+1} \right\rfloor \tag{8}$$

where $\lfloor \cdot \rfloor$ is the smaller integer function. The goal is to find the optimal number of equally spaced inspections that gives the maximum expected utility.

Reliability threshold (RT). The reliability of the component at time step t is its probability of being in a safe condition $\Pr(\overline{F_t}) = \Pr(a_C - a_t > 0)$. Often, it is expressed through the reliability index $\beta_t = \Phi^{-1}[\Pr(\overline{F_t})]$, with Φ^{-1} being the standard normal cumulative

distribution function. A reliability threshold β_m defines a lower bound of the reliability index. For a given threshold, an inspection is planned at time t if it would hold that $\beta_{t+1} < \beta_m$ without this inspection. In this way, the inspection times follow directly from β_m. Note that the implementation of the RT heuristic with the ID differs slightly from the original version, because the computation of $\Pr(\overline{F}_t)$ and β_t is based on the averaged performance, i.e. it is not computed based on the actual repair history as in STRAUB & FABER 0.

The PI and RT heuristics approximate the solution of the no-memory decision problem. Their possible solutions domains are considerably smaller subsets of the complete solution domain, thus reducing the computational effort. Both PI and RT approaches define a single parameter optimization problem. Their algorithms have a linear complexity order with respect to the number of time steps whereas SPU complexity depends on the maximum size of the variable domains considered in the decision problem.

4 Numerical investigations

To investigate the different algorithms for optimizing the inspection planning with respect to the minimal life cycle cost, the simple DBN crack growth model presented in Sec. 3.1 is implemented. The parameters of the model are summarized in Tab. 1. These parameters as well as the discretization scheme are taken from STRAUB 0. In Tab. 2, the variables of the IDs are summarized. The cost of repair, C_R, and system failure, C_S, are expressed relative to the cost of inspection, C_I. The transition matrix **A** of Eq. (5) is estimated by Monte Carlo simulation with 10^6 samples of parameters C, ΔS and m according to the prior distributions of Tab. 1.

In this analysis, the following parameter values were assumed for the life cycle model: total time period $T = 15$ years; system failure cost $C_S = 5000$; inspection cost $C_I = 1$; repair cost $C_R = 0.1$; and redundancy $r = 0.2$. Discounting was neglected for the purpose of this example.

The inspection planning problem was solved for the no-memory ID (Fig. 5) with the PI and RT heuristics, with the SPU algorithm and – for comparison – with a full search (i.e. covering all 2^{15} possible inspection schedules). Furthermore, it was solved for the last-inspection ID (Fig. 6) with the SPU algorithm.

Tab. 1 Parameters of the decision problem

Variable	Distribution	Mean	Standard deviation and correlation
a_0 [mm]	Exponential	1	1
a_c [mm]	Deterministic	50	–
ΔS [N mm^{-2}]	Normal	60	10
$\ln(C), m$ [a]	Bi-Normal	(-33; 3.5)	(0.47; 0.3), $\rho = -0.9$
Δn [yr^{-1}]	Deterministic	10^5	–

[a] Dimensions corresponding to Newton and millimeter

Tab. 2 Domains and variables discretization

Variable	Number of states	Discretization / states	Conditional Probability Distribution
a_t (mm)	80	0,exp{ln(0.01): [ln(50) − ln(0.01)]/78 : ln(50) }, ∞	$Pr(a_t \in I_j \mid a_{t-1} \in I_k) = A_{j,k}$ where I_j is the j-th interval of the discretization of a_t or a'_{t-1}.
$E_{C,t}$	2	1: Safe 0: Fail	$\Pr(E_{C,t} = e \mid a_t) = \begin{cases} 0 & if\ e = 1, a_t \geq a_c \\ 1 & if\ e = 0, a_t \geq a_c \\ 1 & if\ e = 1, a_t < a_c \\ 0 & if\ e = 0, a_t < a_c \end{cases}$
$E_{S,t}$	2	1: Safe 0: Fail	$\Pr(E_{S,t} = e \mid E_{C,t}) = \begin{cases} r_i & if\ e = 1, E_{C,t} = 0 \\ 1 - r_i & if\ e = 0, E_{C,t} = 0 \\ 1 & if\ e = 1, E_{C,t} = 1 \\ 0 & if\ e = 0, E_{C,t} = 1 \end{cases}$
$U_{S,t}$	1	–	$U_{S,t} = \begin{cases} 0 & if\ E_t^S = 0 \\ -C_S & if\ E_t^S = 1 \end{cases}$
D_t	2	0: No inspection 1: Inspection	
$U_{I,t}$	1	–	$U_{I,t} = \begin{cases} 0 & if\ D_t = 0 \\ -C_I & if\ D_t = 1 \end{cases}$
Z_t	3	1: Insp. with detection 2: Insp. with no detection 3: No measurement	$\Pr(Z_t = z \mid a_t, D_t) = \begin{cases} 1 & if\ D_t = 0, z = 3 \\ PoD(a_t) & if\ D_t = 1, z = 1 \\ 1 - PoD(a_t) & if\ D_t = 1, z = 2 \\ 0 & otherwise \end{cases}$
$U_{R,t}$	1	–	$U_{R,t} = \begin{cases} 0 & if\ Z_t = 2,3 \\ -C_R & if\ Z_t = 1 \end{cases}$
a'_t (mm)	80	0,exp{ln(0.01): [ln(50) − ln(0.01)]/78 : ln(50) }, ∞	$\Pr(a'_t \in I_j \mid a_t, Z_t, E_{S,t}) = \begin{cases} \Pr(a_0 \in I_j) & if\ E_{S,t} = 0 \\ \Pr(a_0 \in I_j) & if\ E_{S,t} = 1, Z_t = 1 \\ 1 & if\ E_{S,t} = 1, a_t \in I_j, Z_t = 2 \\ 0 & otherwise \end{cases}$ where I_j is the j-th interval of the discretization of a_t or a'_{t-1}.

4.1 Results

The total expected cost of the inspection planning solutions obtained with the PI and the RT heuristics are shown in Fig. 7. For the PI approach, the optimal number of inspections is found to be 6, with the RT approach the optimal reliability threshold is found to be $\beta_m = 3.34$.

a) b)

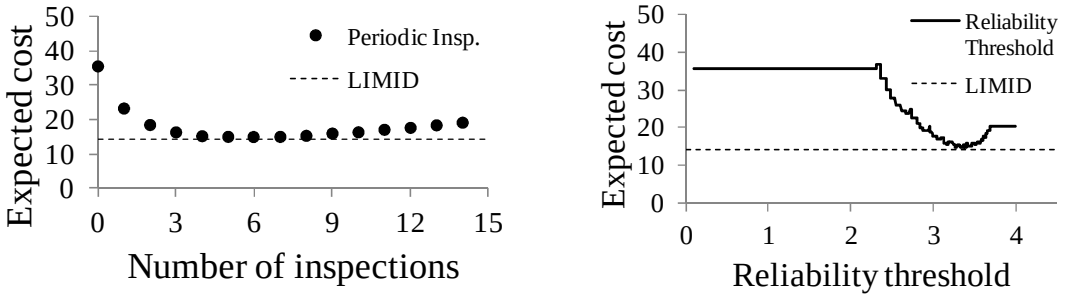

Fig. 7 Expected cost of inspection schedules with (a) periodic inspections and (b) reliability thresholds.

The total expected costs for all five solution strategies are summarized in Tab. 3. The difference in the total expected cost among the optimal solutions is relatively small despite the inspection times and disaggregated costs (i.e. failure, inspection and repair costs) being quite different among the solutions. As an example, Fig. 8 shows the disaggregated expected costs for the PI and the SPU (no-memory ID) solution. The reason for the small differences in expected costs is likely the fact that all optimal solutions of the no-memory ID have six inspections, and the minimum reliability obtained with these optimal solutions is also fairly similar, as evident from Fig. 9.

The optimal solution is obtained with the last-inspection ID, solved using the SPU algorithm. This is not surprising, since this is the only strategy that allows adapting the inspection times based on inspection results. The disadvantage of this strategy is that inspections cannot already be planned at the beginning of service life.

Tab. 3 Optimal solutions obtained with different algorithms.

Approach	Expected cost	Inspection times	CPU time (sec)
PI	14.91	2, 4, 6, 9, 11, 13	0.4
RT	14.70	2, 4, 6, 8, 10, 13	4.3
SPU (no-memory ID)	14.05	1, 2, 4, 5, 7, 9	164
Exact solution (no memory)	13.97	1, 2, 3, 5, 7, 10	313
SPU (last-inspection ID)	13.75	Policies for each decision node	165

Comparing the SPU solutions obtained for the two different IDs, it is seen that the minimum expected cost decreases from 14.05 to 13.75 when the observation from the last performed inspection is taken into account for deciding on the next inspection (the last-inspection ID). In this case, the SPU algorithm provides an adaptive policy for each decision node. For example, the resulting policy of the decision node D_{12} indicates that an inspection is to be carried out unless there was an inspection in the previous year (independent of what was observed) or two years ago without crack detection.

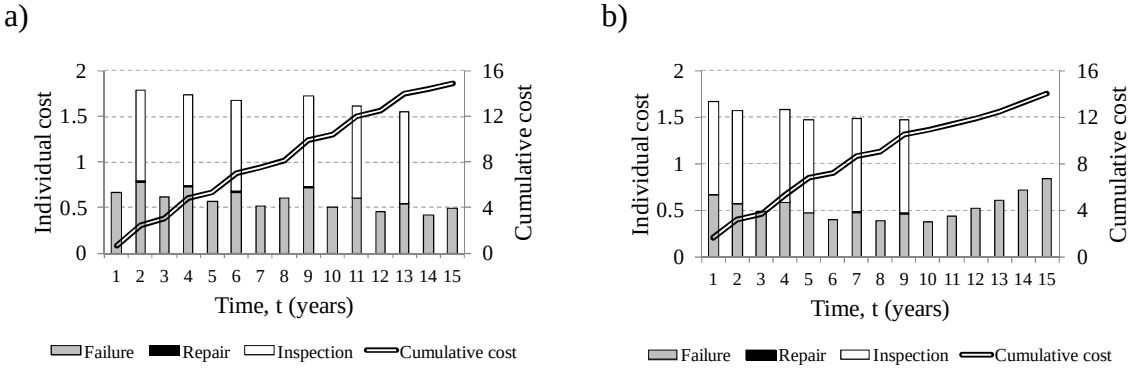

Fig. 8 Expected cost of optimal solution for: a) PI, and b) SPU (no-memory ID)

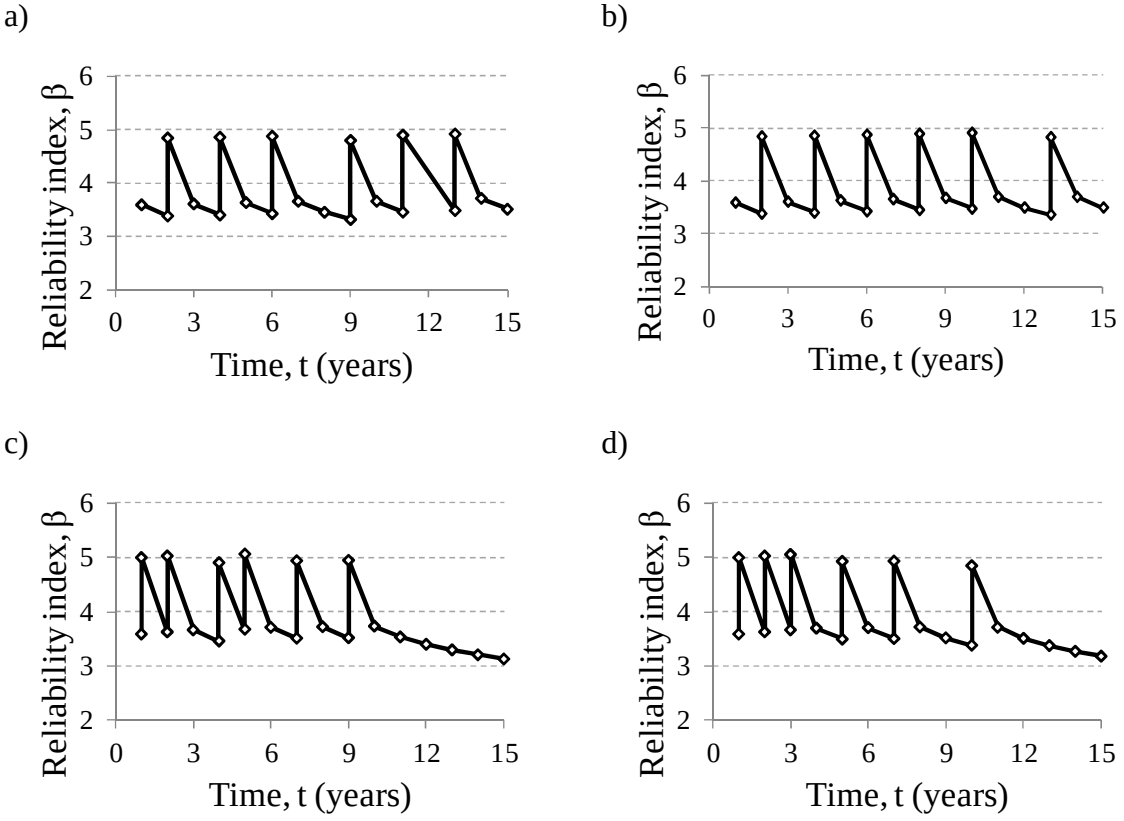

Fig. 9 Reliability index of the optimal solutions for (a) periodic inspections; (b) reliability threshold; (c) SPU (no-memory ID); and (d) the exact solution (no-memory ID).

The computation time for the SPU solution is considerably larger than for PI and RT. The same is observed when increasing the considered service life period T, and hence the number of steps in the DBN (see Fig. 10). However, all these algorithms show a similar com-

plexity order (a linear increase with the number of time steps). In contrast, the exact solution of the no-memory ID was obtained by a complete search among all possible combinations of inspection times. This procedure has an exponential complexity order with respect to the number of time steps. For illustration purpose, the last two points of the exact solution (no-memory) curve in Fig. 10 ($T = 25$ and $T = 50$) were estimated by extrapolation.

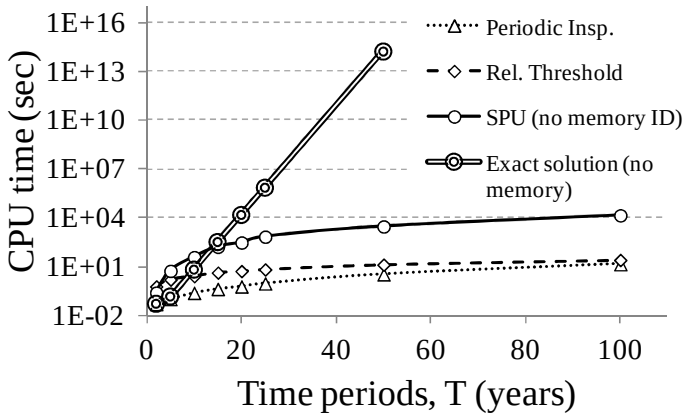

Fig. 10 CPU time for finding the optimal solution

While the SPU algorithm requires significantly more computation time than the PI and RT heuristics, it is more flexible because it allows to adapt the inspections to the results of previous observations, as shown for the case of the last-observation ID. If the observations from the previous inspections are considered before deciding to inspect, an adaptive policy is followed. Since any information can only increase the expected utility of optimal decisions [3,10], it follows that the maximum expected utility of the last-observation ID (or any other adaptive policy implemented through an ID with memory) must be larger or equal than that of a fixed inspection schedule based on the no-memory ID.

5 Conclusions and outlook

In this paper, the optimal inspection times for a structural element subject to fatigue are identified through a set of algorithms. The optimization problem is formulated through influence diagrams, whereby varying assumptions regarding the adaptivity of the inspection schedule were made. As expected, the adaptive inspection scheduling leads to lower expected costs. Among the algorithms for optimizing a non-adaptive inspection plan, the one obtained with the single policy updating (SPU) algorithm performs the best. However, the examined heuristics, which allow to significantly reduce the computational effort in the optimization, also perform well.

These results of this paper are useful in the investigation of inspection planning problems in systems with multiple elements, whose deterioration characteristics are correlated. The proposed DBN/ID framework can be extended to solve such problems, but the associated computational complexity will increase with increasing system size, and simple heuristics are no longer readily available. Therefore, it is essential to have efficient algorithms for solving these problems, and the SPU algorithm seems promising for this purpose.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) through Grant STR 1140/3-1 and the Consejo Nacional de Ciencia y Tecnología (CONACYT) through Grant No. 311700,.

References

DITLEVSEN O, MADSEN HO, 1996. Structural reliability methods. New York, NY: Wiley.

JENSEN F, NIELSEN T, 2007. Bayesian networks and decision graphs. 2nd ed. New York, NY: Springer.

JORDAAN I J, 2005. Decisions under uncertainty. Probabilistic analysis for engineering decisions. Cambridge: Cambridge University Press.

LAURITZEN S, NILSSON D, 2001. Representing and solving decision problems with limited information. Management Science, 47(9): 1235-1251.

NIELSEN J, SØRENSEN J, 2011. Risk-based operation and maintenance of offshore wind turbines using bayesian networks. Proceedings of the 11th International Conference on Applications of Statistics and Probability in Civil Engineering: 311-317, CRC Press LLC.

RAIFFA H, SCHLAIFER R, 1961. Applied Statistical Decision Theory. Cambridge: Cambridge University Press.

STRAUB D, DER KIUREGHIAN A, 2011. Reliability Acceptance Criteria for Deteriorating Elements of Structural Systems. Journal of Structural Engineering, Trans. ASCE 137 (12), 1573–1582.

STRAUB D, 2009. Stochastic modeling of deterioration processes through dynamic bayesian networks. Journal of Engineering Mechanics, Trans. ASCE; 135(10): 1089-99.

STRAUB D, FABER M H, 2006. Computational aspects of risk based inspection planning. Computer-Aided Civil and Infrastructure Engineering, 21(3): 179-192.

STRAUB D, 2014. Value of Information Analysis with Structural Reliability Methods. Structural Safety, in print.

SWANSON L, 2001. Linking maintenance strategies to performance. International Journal of Production Economics,70: 237–244.