The Challenge of Solving POMDPs for Control, Monitoring and Repair of Complex Systems

Emmanuel Benazera[1] and **Elodie Chanthery**[1,2]
[1] LAAS-CNRS, Université de Toulouse, 7 avenue du Colonel Roche,
31077 Toulouse Cedex 4,
[2] Université de Toulouse, INSA
ebenazer,elodie.chanthery@laas.fr

Abstract

A common view holds a clear separation between the act of diagnosing and that of planning/controlling. However, a framework exists, partially observable Markov decision problems (POMDPs), that is holistic for optimal decision and control in partially observable domains. In applications to systems with faults, its solution yields the best strategy, that includes repair and observation actions. In such a decision-based context, it is unclear what a diagnostic is, and under what conditions it is useful or even necessary. This paper discusses the use of POMDPs to the control, monitoring and repair of systems with faults, insisting on their structure and showing how it can be used to improve the scalability of POMDP solving techniques for application to the present domain.

1 A challenging problem

When facing the real world, agents encounter many unexpected situations. Often, their expectation, that is their anticipated future state in the world, is violated. Monitoring detects these violations. Diagnosis isolates one or more faults from the symptoms. Recovery brings an agent plan into alignment with observations [J.L. Fernandez and R. Simmons, 1998]. There are a number of unavoidable difficulties when articulating diagnosis, monitoring and recovery/control:

- A common view holds a clear separation between the act of diagnosing and that of planning/controlling. Whenever a fault is detected, diagnoses are found first, before a correcting action is taken [Ghallab *et al.*, 2001], [Nesnas *et al.*, 2003].
- The system state, and thus the diagnoses, are rarely identified with certainty.
- Short-term control or repair actions are preferred despite the fact they are myopic and fall into local optima.

Unfortunately, neither the failure occurrence nor detection points are hardly good points for reconfiguration and repair[1]. This is because some faulty contingencies require anticipation. For example, switching to a backup system may require it be initialized in advance. For time critical operations, the optimal repair action is thus to preventively start the backup system at some point before the fault is detected. Additionally, any equipment wears over time and becomes more likely to fail. Often, preventive actions can be taken that simply avoids the occurrence of a fault. Finally, planning for a long-term horizon avoids the pitfall of locally good actions whose effects might prove counterproductive later. Typically, shutting off a potentially faulty equipment will shut sensor readings about its health, and prevent a finer disambiguation. Overall, diagnosis and recovery are seen as a single, dynamic, and practical activity by an agent acting in an uncertain and changing world [B. D'Ambrosio, 1996].

This short analysis suggests that diagnoses are in reality an artefact between observation and control. The optimal chain of controlling actions would most often not require the exact diagnoses to be known. Most certainly, the computation of this chain does require some knowledge of a link between faults and observations.

1.1 POMDPs as a holistic framework

A partially observable Markov decision process (POMDP) models control problems for which actions have stochastic effects and sensors provide imperfect and incomplete state information. This model has been widely adopted by the AI community as a framework for research in planning and control under uncertainty. Its generality allows to model sensor and action uncertainty, uncertainty in the state of knowledge, and multiple objectives. As such, and under the Markov assumption, it is the holistic model for control, monitoring and repair of complex systems. This is because a solution to a POMDP embodies the optimal chain of actions for all possible beliefs over the world states. This optimal chain of actions is referred to as the optimal policy, solution to the POMDP. If there exists any gain to be made by avoiding or mitigating certain faulty effects, the solution of a POMDP would capture it in its optimal policy.

Another useful advantage of POMDPs is their support of reasoning about whether to select actions that change the state, provide state information, or both. In domains with faults, this proves a key feature since it allows actions to be used to remove ambiguities over the true system state. This is

[1] A well admitted problem in both scheduling and planning.

especially true of actions that carry no utility or cost (mostly used for responding to unlikely situations) and that most planners handle with difficulty since these actions can be inserted anywhere in a plan. In a POMDP framework, any action that changes the state of the world is likely to generate informative observations, and therefore be positioned with accuracy.

Unfortunately despite recent improvements, current solution algorithms still limits the application of the POMDP framework to real world problems. The standard approach to solving a POMDP involves two steps. First the POMDP is transformed into a fully observable Markov decision process with a state space that consists of all probability distributions over the core states of the POMDP. Second, it is solved in this form. For a POMDP with n core states, the transformed state space is the n-dimensional simplex, also called belief simplex. This continuous state space is a challenge but can be tackled in practice, either optimally or through approximations. Today, the most promising methods can solve problems with a few thousand states, and give approximation solutions to problems with millions of states.

1.2 POMDPs for control, monitoring and repair

This paper discusses how to utilize the POMDP framework for control real-world artefacts that are subjected to many unforeseen faulty events. These systems can be characterized as being governed by a number of hard constraints:

- Most actions are potentially risky [Kurien and Nayak, 2000]. Typically an action exhibits a few nominal effects, and a high number of faulty outcomes.
- Many faults, even with low probability of occurrence, can occur anytime. Consequently, they exponentially increase the complexity of the control problem. Such faults are referred to as anytime faults.
- As a consequence the state space is in general orders of magnitude larger than that of currently solvable POMDPs.
- A side effect of anytime faults is that the whole space of faulty states is almost fully reachable from any belief state. This is because the same fault may occur in many different contexts.

This suggests that new approaches could advance the articulation of control, monitoring and repair to a point where it can consider real-world applications with focus on models with faults. This paper is tentative to make the point that techniques for diagnosis and efficient state estimation can benefit the solving of POMDPs. It observes that models of the real-world artefacts that contain faults exhibit a characterized structure:

- Action effects are naturally partitioned into nominal and faulty outcomes.
- Each fault occurrence builds up and becomes more probable over time. Consequently, the probability of faulty states is likely to be higher near the end of an artefact's life.
- Under an assumption of single fault independence, states that embody multiple faults are expected to be orders of magnitude less likely than others. For this reason the order of a state refers to the number of faults it embodies.
- Under a permanent fault assumption, each state of a given order may lead to states of a limited number of other orders.
- Not all faults can be modeled or are even known. These events are modeled as putting the system into a special *unknown* state.

This structure may be used to design tailored computational methods capable of handling this class of problems. The applicability of the POMDP framework has been recognized for domains with faults such as ours. However, the structure of the models and problems that are typical of these domains do seem to have been at most briefly considered by the operation research and planning communities. It is not sure the structural properties mentioned above can generalize to control and planning problems of other domains.

2 Previous Work

There has been considerable work on POMDPs, too many to discuss here. But research on efficient computational solutions to POMDPs has made great progress over the past decade.

2.1 Background on POMDPs

We give background on a standard POMDP as found in the literature. The relationship between an agent and its environment is modeled as a discrete-time POMDP with a finite set of states S, a finite set of actions A, and a finite set of observations $\mathcal{O}$. Each time period, the environment is in some state $s \in S$, the agent takes an action $a \in A$, and receives a reward for it, with expected value $r(s, a)$. Taking action a makes the environment transition to state s' with probability $P(s' \mid s, a)$. In s', the agent observes $o \in \mathcal{O}$ with probability $P(o \mid s', a)$. Let note b the vector of state probabilities. $b(s)$ denotes the probability that the environment is in state s. Given $b(s)$, then after taking a and observing o,

$$b'(s') = \frac{1}{P(o \mid b, a)} P(o \mid, s', a) \sum_{s \in S} P(s' \mid s, a) b(s) \quad (1)$$

where $P(o \mid b, a) = \sum_{s' \in S} P(o \mid s', a) \sum_{s \in S} P(s' \mid s, a) b(s)$ is a normalizing constant. This belief update is noted $b' = \tau(b, a, o)$.

The agent's policy π specifies an action $\pi(b)$ for any belief b. It is assumed the objective is to maximize the expected total discounted reward over an infinite horizon. Thus the expected reward for π starting from belief b is defined as

$$J^{\pi}(b) = E\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid b, \pi\Big] \quad (2)$$

where $\gamma < 1$ is the discount factor. The optimal policy π^* is obtained by optimizing the long-term reward.

$$\pi^* = \underset{\pi}{\text{argmax}} J^{\pi}(b_0) \quad (3)$$

where b_0 is the initial belief.

Exact algorithms for solving this optimization problem are intractable for all but trivial problems. These algorithms solve the Bellman optimality equation that yields the optimal value function,

$$V^*(b) = \max_{a \in A} \Big[r(b,a) + \gamma \sum_{o \in \mathcal{O}} P(o \mid b,a) V^*(\tau(b,a,o)) \Big] \tag{4}$$

For finite-horizon POMDPs, the optimal value function is piecewise-linear and convex [E.J. Sondik, 1971]. It can be represented as a finite set of vectors. In the infinite-horizon formulation, a finite vector set can approximate V^* arbitrarily closely, whose shape remains convex. Value iteration applies dynamic programming update to gradually improve on the value until convergence to an ϵ-optimal value function, and preserves its piecewise linearity and convexity. A single dynamic programming pass is often referred to as a backup operation. By improving the value, it implicitely improves the policy as well. Another dynamic programming called policy iteration explicitely represents and improves the policy instead [E. Hansen, 1998].

2.2 Computational solutions

Most of the useful structures and methods have been exploited. As a matter of fact, most solvers now yield very good approximations of the true optimal controls. The approximations are of at least three types: piecewise linear value functions; fixed and variable-resolution grids; belief state space compression. These approximation techniques make use of a set of properties of the POMDP framework. They can be summarize as: reachability analysis; heuristic search; factored representations.

Many algorithms employ a piecewise linear convex representation of the value function and use gradient backups to solve the Bellman optimality equation [E.J. Sondik, 1971; A.R. Cassandra *et al.*, 1997; Kaelbling *et al.*, 1998; E. Hansen, 1998]. These techniques perform gradient backups over the full belief state space. Doing this, the main problem remains the elevated number of vector components, that prevent efficient pruning and linear programming but for a few hundreds of states, at best. Models that include faults have state spaces that are order of magnitude larger, and out of the reach of these representations and techniques.

Another host of algorithms prefer to approximate value functions by focusing their backups onto the relevant belief states instead. The technique is named Point-based dynamic programming. [J. Pineau *et al.*, 2003] present Point-Based Value Iteration (PBVI), an algorithm that scaled the solving of POMDPs up to problems with a thousand states. Point-based techniques in general must select a set of relevant belief points to be backed up. Generation of these points is done through sampling from the action and observation models to generate reachable beliefs. Sampling from a uniform distribution over the full belief simplex has been reported to perform poorly [J. Pineau *et al.*, 2003]. In POMDP models with faults, techniques that make use of reachability suffer from two symptoms. First, as already mentioned, anytime faults make the full state space reachable after a few time steps. Second, point-based techniques would fail backing up the value at most of the low probability states. Recall that most of the state-space is composed of a very high number of low probability states. Therefore missing low probability events inevitably leads to losing a large fraction of the probability mass.

Similar point-based updates can be found in grid-based methods where points form a fixed [W. Lovejoy, 1991] or variable grid over the belief state space [R. Brafman, 1997; R. Zhou and E.A. Hansen, 2001]. While these techniques avoid the pitfalls of stochastic simulation, they do not scale well due to the so-called curse of dimensionality: discretization of the belief simplex scales exponentially with the number of states.

Therefore, another batch of approximated computational solutions try to abstract or compress the POMDP state-space or belief space, respectively. Abstraction uses a reduced (most often factored) representation of the POMDP states in local operations and backups. The background idea is the following: whenever some variables do not affect either some decision of a policy or equivalently, the plan value, they can be marginalized away in the local computations of this policy or value function. Of course the difficulty is to detect these situations at a sufficiently low cost. A popular technique is the exploitation of a factored representation that avoids enumeration of the state-space. [C. Boutilier and D. Poole, 1996] show how to exploit such a representation for POMDPs. An improved version for both value and policy iteration was detailed in [E. A. Hansen and Z. Feng, 2000]. [T. Smith *et al.*, 2007] progresses one step further and builds an abstraction of the factored representation. But the method relies on a structural decomposition of upstream and downstream variables where the former are known by the agent and transition deterministically and the later cannot influence the former. In systems with faults this is almost never the case: stochasticity is at its highest in the upstream variables whereas downstream dynamics are assumed to follow some fault models. Additionnally, upstream variables are almost never directly observed.

Compression tries to compact and reduce the belief state-space directly. A pioneer work is the dimensionality-reduction technique of [P. Poupart and C. Boutilier, 2002] dubbed Value-directed compression (VDC). It computes a low-dimensional representation of a POMDP directly from its model by finding the Krylov subspace for the reward function under belief propagation. To summarize, it finds the smallest subspace that compresses the value function such that only beliefs that have different values are distinguished. Two drawbacks of this technique is that the Krylov subspace is constrained to be linear and that its computation is as difficult as the exact solving of the underlying POMDP. [P. Poupart and C. Boutilier, 2004] mixes a policy reduction mechanism with VDC and reports on tackling a very large POMDP model of network maintenance (see next section).

[N. Roy *et al.*, 2005] observes that the beliefs are unlikely to lie on a low-dimensional hyperplane. Therefore they propose a non-linear compression scheme, E-PCA. E-PCA is non-linear principal component analysis that extracts a low dimensionality belief space from a heuristic belief distribution. This distribution must be generated for each given prob-

	Piecewise linear convex	grid	heuristic search	compression
No faults	[A.R. Cassandra *et al.*, 1997], [Kaelbling *et al.*, 1998], [J. Pineau *et al.*, 2003], [M.T.J. Spaan and N. Vassis, 2005], [T. Smith and R.G. Simmons, 2005]	[W. Lovejoy, 1991],[R. Brafman, 1997],[M. Hauskrecht, 1997; 2000],[R. Zhou and E.A. Hansen, 2001]	[M. Hauskrecht, 1997], [R. Brafman, 1997], [T. Smith and R. Simmons, 2004], [G. Shani *et al.*, 2007]	[E. A. Hansen and Z. Feng, 2000],[N. Roy *et al.*, 2005]
Faults	[Cassandra, 1998]	[B. D'Ambrosio, 1996]	[Joshi *et al.*, 2005]	[P. Poupart and C. Boutilier, 2002],[B. D'Ambrosio, 1996]

Table 1: Approximation techniques and fault models.

lem. Doing so requires a heuristic controller. In the case of a navigating robot, this means moving the robot around with a heuristic controller and recording the observations. While this technique scales up to problems with a few thousand states, it is not applicable to models with faults. The reason being that a heuristic controller is unlikely to make most faults appear. In other words, it is likely the belief space would compress to its nominal counterpart, leaving most if not all faulty dimensions away.

A class of algorithms combine heuristic (forward) search with dynamic programming backups. [M. Hauskrecht, 1997] describes an algorithm for incrementally calculating the upper and lower bounds a POMDP value function. This bound is initialized with the value for the underlying MDP. It provides a popular way of initializing an upper bound [M.L. Littman *et al.*, 1995], focusing forward exploration [T. Smith and R. Simmons, 2004], heuristic action selection [G. Shani *et al.*, 2007] or value based clustering [Y. Virin *et al.*, 2007]. Heuristic Search Value Iteration (HSVI) [T. Smith and R. Simmons, 2004] has shown some of the best performances yet over large scale problems. HSVI maintains both an upper and a lower bounds over the value function. It greedily selects belief points through forward exploration and execute backups at the selected points in reversal order on the traversed path. HSVI avoids the pitfall of sampling the next belief state to be explored and relies on an informed heuristic instead. Its drawback is that it does not compress the belief space so that backups and the computation of its upper bound to the value function are very time consuming and prevent its scaling beyond systems with twenty thousand states or so.

2.3 POMDPs for models with faults

Of all these techniques, we have not found even a handful of them that are reported to have been applied to POMDP models that contain faults. Table 1 report on these works. Among them [B. D'Ambrosio, 1996] is maybe the oldest and the most insightful. It proposes a qualitative compression technique and applies it to a gate-circuit with faults. The author notes how the formulation of diagnosis describes a static, detached process from recovery or maintenance in general. In contrast, he formulates diagnosis as dynamic, practical activity by an agent engaged in an uncertain world, referred to as an on-line maintenance task, a follow-up of his previous work [B. d'Ambrosio, 1992]. Interestingly, he mentions how it is not obvious what elements of a diagnosis are relevant to decision. The gate-circuit model includes an unknown behavioral mode whose behavior is stochastic. [Joshi *et al.*, 2005] formulates a special POMDP whose non directly observable states are fault hypotheses. The optimal policy minimizes the cost of the potential system faults. Of course this work suffers from the size of the problem and falls back on an approximated solution. The sole originality of this solution is the use of a variable horizon: dynamic programming recursion stops whenever the recovered faulty value function falls back within fixed bounds of its sane counterpart. Finally, [M. Hauskrecht, 2001] relies on Monte-Carlo simulation and value function approximations for producing plans with application to the management of patients and medical treatment planning.

3 Examples of control, monitoring and repair of complex systems

This section presents examples of control, monitoring and repair found in the literature and solved by POMDP. These examples have not been designed with monitoring and repair in mind. Thus they lack some key structural properties that are of interest for tackling real world control and repair problems.

3.1 Gate digital circuits

The gate digital circuit example was introduced [B. d'Ambrosio, 1992]. It comes in two flavors: a four gates circuit or half adder, and a seven gates circuit. Within each circuit, a gate has four possible states: *ok*, *stuck-at0*, *stuck-at1* and *unknown*. In the latter, the gate output is a stochastic function, independent of the input and uniformly distributed over $\{0, 1\}$. Failure probabilities are uniformly distributed over the four states. The agent observes inputs and outputs. Overall, the model is similar to the classical Polybox example. However, the agent can act in several ways: by replacing any component; probing the output of components (in which case the respective value was added to the observation set for the next cycle), or to perform no action. The reward is -1 for each cycle in which at least one component is faulted, -6 for a replacement, and -1 for a probe action. Ignoring the stochastic behavior of the *unknown* state, the four gates problem exhibits 256 states. As multiple faults are possible, all 256 states are reachable. A policy for this problem anticipates the fault occurrence if the cost of replacing a gate is compensated by the gain of not having several cycles in which at least one component was faulted.

The gate circuit is an interesting model since it is realistic and explicitly models the unknown state of behavior. However, its replacement action is deterministic. Also, there is no wear to the board, so the fault rate remains constant (e.g. there is no modeled wear to the board).

3.2 Machine Maintenance of Cassandra

The machine maintenance problem is an early but difficult application model for POMDPs [Cassandra, 1998]. The model

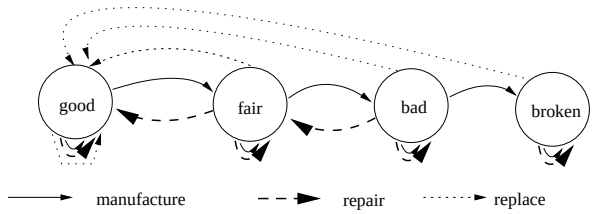

Figure 1: State transitions for the machine maintenance

is very generic: it describes a machine made of c internal components. The machine is used to produce a product. The quality of the product is a function of the state of the internal components. Every component is modeled by four possible states: *good, fair, bad* (it can be repaired) and *broken* (it must be replaced). As each components has four possible states, the problem has 4^c states. Components can only be observed if the machine is disassembled: this requires an expenditure of time and personnel while rendering the machine unproductive for the duration of the inspection.

The agent seeks the maintenance of the machine and can utilize a finite set of actions. The *manufacture* action proceeds with production while each component deteriorates with probability 0.03 after each day. The only possible transitions are from *good* to *fair*, from *fair* to *bad* and from *bad* to *broken*. This is represented Figure 1. The *inspect* action does not change the underlying state but yields information about the hidden state, see below. The *repair* action improves every component's condition with probability 0.8. Possible transitions are from *fair* to *good* and *bad* to *fair*. A *broken* component cannot be repaired. The *replace* action deterministically puts a whole new set of components in *good* condition. Taking the *manufacture* action allows to observe the quality of the products, either *good* or *bad*. A *good* component always performs properly during the day. A *fair* component has probability 0.95 of performing properly, while a *bad* component has probability 0.75. A *broken* component never performs properly. Taking the *inspect* action, the observation is a composite of individual observation for each component, each of which is observed to be in a either *good* or *bad* condition. The probabilities depend on the actual condition of the component. A *good* component will yield a *good* observation with probability 0.97, a *fair* component looks *good* with probability 0.80, *bad* with probability 0.05 and *broken* with probability 0.02. There is no observation for the *repair* and *replace* actions. The agent is rewarded every the day (1 for each good product). The *inspect* action costs -1. Repair costs -3. Replacing the machine costs -15.

This example is very complete, generic, and can be easily extended with other components. However it does not model wear since the failure rate remains constant.

3.3 The network maintenance example

This example is presented in [P. Poupart and C. Boutilier, 2004]. A system administrator (SA) maintains a network of machines. Each machine has a 0.1 probability of failing at any stage; but this increases to 0.333 when a neighboring machine is down. The SA only observes the status of a machine (with 0.95 accuracy) if he reboots or pings it. At each stage, he can either reboot or ping a machine, or do nothing. The SA receives a reward of 1 per working machine and 2 per working server. Costs are 2.5 (rebooting), 0.1 (pinging), and 0 (doing nothing). An n-machine network induces to a POMDP with 2^n states, $2n + 1$ actions and $2n$ observations. This example is scalable and can exhibit over 33 millions states (for 26 machines). Of interest here is the failure rate, that is no more constant. In fact it depends on the topology of the network. The model comes in two flavors: a ring topology in which each computer communicates with at most two neighbors; a star-like topology: a tree of three branches joined at the root. While this is somehow realistic, the failure rates remain constant over time.

3.4 A novel example: the taxi maintenance

We introduce a novel example that embodies the most realistic features of a control, monitoring and repair problem. The taxi maintenance problem builds on Cassandra's machine. A driver gets money each day by driving its taxi around. His car is made of several critical parts, and is modeled has a machine with c components. In addition to the component states, the taxi maintenance models the age of the car. The taxi maintenance problem has several important features:

- the failure rate of every component is a function of the age of the car;
- the car ages faster when driven with one or more bad or broken components;
- the decision problem ends whenever the car must be replaced.

The car wears normally everyday so its age increases linearly with time. But driving the taxi with some faulty internal components wears the car faster. Very importantly, the failure rate is an increasing function of the car's age. Here, the fault probability distribution of every component is modeled by a Weibull distribution with a shape parameter > 1 [W. Weibull, 1951]. The model considers the problem over a finite horizon that is a number of days d. The true age of the car is bounded by $l \geq d$ (after which it must be replaced). The taxi maintenance problem can be seen as a machine maintenance problem for every possible age value of the car. Thus the number of states is $l4^c$.

The taxi maintenance problem poses a certain number of problems to existing POMDP techniques. First, approximation methods that select the most likely belief states would lose track of the true car's age. The consequence is a disastrous cascading effect: as age determines the true failure rates, future failure rates are wrongly estimated, augmenting the drift from the true car's age. Second, due to potential large values of l, the state-space is very large: considering the problem over five years yields 1825 days, and 467200 states. Third, since the car's age remains a function of time, the belief over the true car's age is pushed forward over time. In consequence, compression techniques able to mitigate the high number of states would need to leverage on reachability and be dynamic over time to remain efficient.

4 Approaches

The idea defended here is that solution approaches could benefit from a diagnostic and repair stance on the monitoring and control problem. Thus some of the intuitions in the DX community that led to good solution techniques could act as the root basis for targeted approaches to the solving of POMDPs.

4.1 Belief point selection

Typically, the control of a POMDP is decomposed into two parts: a state estimator that performs the belief update of equation (1); a planner, that finds the optimal policy (4). State estimation of models with faults is a topic that has been successfully tackled by the Control, FDI and DX communities. Of special interest here, the blowup in state estimates of complex plants modeled as discrete and hybrid systems has received a large attention. Solution techniques employ reasoning and statistical methods that leverage the number of estimates. Two examples from the literature are worth considering.

[Hofbaur and Williams, 2004] mitigates the number of state estimates at each time step. It utilizes an A*-like search procedure with an information-based heuristic to select the set of most likely estimates. A similar strategy can be employed in POMDP solution techniques. The difference is that all observations must be considered. Thus a forward search procedure would retain a fixed size set of most likely combination of actions and observations. In a way this is similar to HSVI's expansion phase that produces trajectories within the belief space. But the generated traversals carry single belief points. Instead, the proposed extension would select and project sets of most likely points.

[Verma *et al.*, 2003] introduces a variable resolution particle filter. The resolution of state space is dynamically varied by region, depending on the belief that the true state lies within a region. Where the belief is strong, the resolution is fine, where the belief is low, the resolution is coarse, abstracting multiple similar states together. This abstraction is predefined while the resolution results of a bias-variance tradeoff. Taking this mechanism to POMDP is akin to a variable resolution dynamic compression of the state-space. The underlying intuition however slightly differs. Where in the belief simplex the belief is strong, i.e. at and around its corners, the resolution is coarse. This reflects that the statistic contains enough information for near optimal control. The resolution is fine in and around the pit of the belief convex hull. This is because in this belief region the agent is very uncertain about where the true underlying state of the world: selection of an appropriate action demands a reduced bias.

4.2 Simple rank-based compression

In a way, uncertainty in action outcomes can be viewed as a halo of unwanted, sometime unforeseen, consequences that surround a desired outcome. This hoped-for outcome is here dubbed *nominal*. It can be modeled as having a higher probability of occurrence than other outcomes, a reflection of the original (or nominal) intention hidden behind the action that produces it. Often it is simply modeled as the mean expected outcome. Note that a consequence is that the value of states

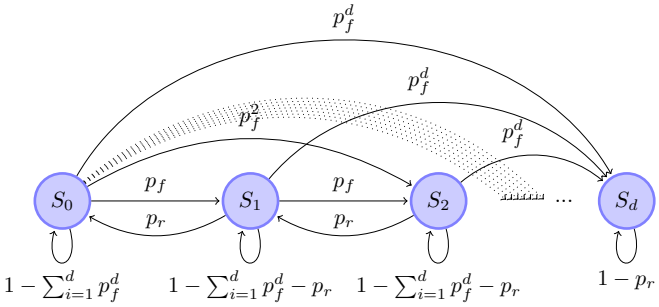

Figure 2: Rank automaton with a one step repair: multiple fault hypotheses with up to d faults and a single repair per time step.

with high rewards often "trickle down" to neighboring states [Y. Virin *et al.*, 2007]. Under an assumption of single fault independence, it is clear that states that embody deviations from the nominal outcomes become order of magnitude less likely than the nominal state over time. Of course in certain cases deviation effects can cancel out. This is akin to the case of non exoneration of multiple faults in diagnosis.

So an approach is to utilize the notion of rank [M. Goldszmith and J. Pearl, 1996] for representing an order of magnitude probability scale for faults. The nominal states are said to be of rank 0. Failure states are one or more orders of magnitude less likely and thus of ranks 1 to n. This qualitative representation also acknowledges the fact that real probabilities are often either inaccessible or unknown. Thus the notion of rank can be considered as an expression of the degree of doubt.

Assume a POMDP models n individual component faults. It is possible to abstract this POMDP state space according to ranks and to build a compressed POMDP over the ranks. This *rank-based* POMDP has at least $2^n + 1$ states: all possible faults combinations plus at least one nominal state. Let note the compressed state space $S = \{S_1, S_2, \cdots, S_n\}$ where S_k is the abstract state of rank k. Each S_k contains all original states that model exactly k faults. S_k contains $\binom{n}{k}$ fault combinations.

Within this compressed state space, each action has a nominal outcome of rank 0, and some deviation outcomes of rank ranging from 1 to n. Considering a single fault per time step, action $a \in A$ outcomes can be compressed into at most $2n+1$ compressed outcomes: n faulty, n repair, and one nominal outcomes. Thus actions for the rank-based POMDP are reduced to simple jumps among compressed states of ranks: k to $k+1$ in case of a fault occurrence; k to $k-1$ in case of fault repair; k to itself if nothing happens (e.g. nominal outcome realized from a faulty state). The compression of actions easily extends to $d \leq n$ faults per time step. Figure 2 pictures a compressed automaton for such a POMDP whose actions can all be compressed into faulty outcomes of deepest rank d at each time step, and a one step repair outcome, of respective probabilities p_f and p_r. Note how the Machine Maintenance example is a variant of this model with a single fault per time step, and a repair action outcome is a jump over up to four ranks. Both the observation and reward function are com-

pressed over S. Simply, $R(S_k, a) = \sum_{s \in S_k} R(s, a)$.

4.3 Rank-based dynamic compression

The compression scheme above regroups faults based on their likelihood of occurrence. In consequence it cannot distinguish among faults. In general this is bad for decision since some faults can have more deadly effects than others. So the agent lacks proper discernment and chooses actions greedily, oblivious of their true effect. Applied to the network maintenance example, this means that the agent would not be able to distinguish among the failed machines, but only among the number of failed machines. Applied to a star like topology on this same network problem, the strategy may be a compression that yields a useful approximated policy: it often makes sense to reboot the central machine so the ability to distinguish among all machines is not essential. What matters is for which belief level does it make sense to reboot this server machine. Applied to a ring topology however, the result is potentially disastrous: the best policy available to the agent would be something akin to randomly choose and reboot a machine. This is because the granularity of available observations (through pinging a machine) is abstracted away by the rank-based compression.

In practice this effect is slightly mitigated because the rank-based compression is dynamic. Dynamic compression is policy dependent. This means that the states in each S_k when following a policy are those states of rank k that are reachable by this policy. This is useful since some faults might simply never occur when certain chains of actions are taken. For example, regularly repairing an equipment of its taxi allows the driver to almost never experience a fault on this equipment. However, the approach is not scalable since the compression into ranks is ad-hoc and domain dependent. For example, it does not extend to robot navigation problems often found in the POMDP literature.

A more general approach starts from [M. Goldszmith and J. Pearl, 1996] and writes beliefs as polynomials of the form $b(s) = \sum_{k=1}^{n} b_k(s)\epsilon^k$ where the number ϵ is infinitesimal. The compression scheme builds a compressed belief $\tilde{b}_k = \sum_{s \in S} b_k(s)\epsilon^k$. This compression is lossy but ϵ and n control its accuracy w.r.t. the original POMDP. Error carried by a compressed belief point is bounded by ϵ^n. The compressed POMDP has now n states. The b_k define a basis for the vector space of the value function. Importantly, compression is dynamic: $\tilde{b}_k$ and $\tilde{b}'_k$ at different time steps define different basis. This means an adaptive compression is applied at each time step that conserves the order of magnitude relatively among the states. The bad news is that the value function is now expressed in a different basis at each time step. This means the vectors that compose it must be transformed from one basis to the other during backup operations. The transformation function is a compressed version of the action model of the original POMDP.

Overall, this approach is close to the qualitative belief space described in [B. D'Ambrosio, 1996]. However, [B. D'Ambrosio, 1996] applies a Kappa calculus point-based abstraction to a model-free POMDP. Thus it does not carry a full representation of the value function as a set of vectors, and does not compress the action model, but learns it from a set of value points instead. The main difference of the outline approach w.r.t. VDC is that it is dynamic and leverages on the reachability of the belief states. Thus compression near the end of a policy abstracts away unreachable states. In our application domain, this means more weight is given to the faulty events that are more likely to occur over time.

Conclusion

This paper has discussed both the advantages and the challenges of modeling control, monitoring and repair problems as POMDPs and solving them. A novel example is given, the taxi maintenance problem, that embodies most of the features of real world control problems for artefacts subjected to wear and multiple faults. Research on POMDPs remains a very active fields. Many recent advances allow to close the gap between reality and its problems on one side, and representation and computational techniques on the other side. Let us mention a few more very recent works in decision and control in partially observable domains and that should be of interest to the DX and FDI community. In [H. Itoh and K. Nakamura, 2007] is presented a version of the POMDP framework with imprecise (intervals) parameters, [S. Ross *et al.*, 2007] similarly allows uncertain parameters but models and learns them with Dirichlet distributions. Finally, [E. Brunskill *et al.*, 2008] mixes a dynamical model of a system's dynamics (in the form of differential equations) with the standard POMDP framework, in what we believe is a necessary step for the robust and intelligent control of future machines.

References

[A.R. Cassandra *et al.*, 1997] A.R. Cassandra, M.L. Littman, and N.L. Zhang. Incremental pruning: A simple, fast, efficient method for partially observable Markov decision processes. In *Proc. of the 13th International Conf. on Uncertainty In Artificial Intelligence*, 1997.

[B. d'Ambrosio, 1992] B. d'Ambrosio. Value-driven real-time diagnosis. In *3d International Workshop on the Principles of Diagnosis*, 1992.

[B. D'Ambrosio, 1996] B. D'Ambrosio. POMDP learning using qualititative belief spaces. In *Neural Information Processing Systems*, 1996.

[C. Boutilier and D. Poole, 1996] C. Boutilier and D. Poole. Computing optimal policies for partially observable decision processes using compact representations. In *Proc. of the 14th National Conf. on Artificial Intelligence*, 1996.

[Cassandra, 1998] A.R. Cassandra. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes*. PhD thesis, Brown University, 1998.

[E. A. Hansen and Z. Feng, 2000] E. A. Hansen and Z. Feng. Dynamic Programming for POMDPs using a Factored State Representation. In *Proc. of the 5th International Conf. on Artificial Intelligence Planning and Scheduling*, 2000.

[E. Brunskill *et al.*, 2008] E. Brunskill, L. Kaelbling, T. Lozano-Perez, and N. Roy. Continuous state POMDP with hybrid dynamics. In *International Symposium on Artificial Intelligence and Mathematics*, 2008.

[E. Hansen, 1998] E. Hansen. Solving POMDPs by searching in policy space. In *Proc. of the 14th International Conf. on Uncertainty In Artificial Intelligence*, 1998.

[E.J. Sondik, 1971] E.J. Sondik. *The optimal control of partially observable Markov processes*. PhD thesis, Stanford University, 1971.

[G. Shani *et al.*, 2007] G. Shani, R.I. Brafman, and S.E. Shimony. Forward Search Value Iteration for POMDPs. In *Proc. of the 20th International Joint Conf. on Artificial Intelligence*, 2007.

[Ghallab *et al.*, 2001] M. Ghallab, F. Ingrand, S. Lemai, and F. Py. Architecture and tools for autonomy in space. In *ISAIRAS*, 2001.

[H. Itoh and K. Nakamura, 2007] H. Itoh and K. Nakamura. Partially observable Markov decision processes with imprecise parameters. *AI*, 171:453–490, 2007.

[Hofbaur and Williams, 2004] M.W Hofbaur and B.C Williams. Hybrid estimation of complex systems. *IEEE Transactions on Systems Man and Cybernetics Part B: Cybernetics*, 34(5):2178–2191, 2004.

[J. Pineau *et al.*, 2003] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. of the 18th International Joint Conf. on Artificial Intelligence*, 2003.

[J.L. Fernandez and R. Simmons, 1998] J.L. Fernandez and R. Simmons. Robust execution monitoring for navigation plans. In *Proc. of the Conference on Intelligent Robots and Systems*, 1998.

[Joshi *et al.*, 2005] K.R. Joshi, M.A Hiltunen, W.H. Sanders, and R.D. Schlichting. Automatic model-driven recovery in distributed systems. In *Proc. of the 24th IEEE Symposium on Reliable Distributed Systems*, 2005.

[Kaelbling *et al.*, 1998] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *AI*, 101:99–134, 1998.

[Kurien and Nayak, 2000] J. Kurien and P. P. Nayak. Back to the future for consistency-based trajectory tracking. In *Proc. of the 16th National Conf. on Artificial Intelligence*, 2000.

[M. Goldszmith and J. Pearl, 1996] M. Goldszmith and J. Pearl. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *AI*, 84:57–112, 1996.

[M. Hauskrecht, 1997] M. Hauskrecht. Incremental methods for computing bounds in partially observable Markov decision processes. In *Proc. of the 15th National Conf. on Artificial Intelligence*, 1997.

[M. Hauskrecht, 2000] M. Hauskrecht. Value-function approximations for partially observable Markov decision processes. *jair*, 13:33–94, 2000.

[M. Hauskrecht, 2001] M. Hauskrecht. Evaluation and optimization of management plans for stochastic dynamic systems with imperfect observations. In *Twelth International Workshop on Principles of Diagnosis*, 2001.

[M.L. Littman *et al.*, 1995] M.L. Littman, A.R. Cassandra, and L.P. Kaelbling. Learning policies for partially observable environments: scaling up. In *International Conf. on Machine Learning*, 1995.

[M.T.J. Spaan and N. Vassis, 2005] M.T.J. Spaan and N. Vassis. Perseus: Randomized point-based value iteration for POMDPs. *jair*, 24:195–220, 2005.

[N. Roy *et al.*, 2005] N. Roy, G. Gordon, and S. Thrun. Finding Approximate POMDP Solutions Through Belief Compression. *Journal of Artificial Intelligence Research*, January 2005.

[Nesnas *et al.*, 2003] I.A. Nesnas, A. Wright, M. Bajracharya, R. Simmons, T. Estlin, and Won Soo Kim. CLARAty: An architecture for reusable robotic software. In *SPIE Aerosense Conf.*, 2003.

[P. Poupart and C. Boutilier, 2002] P. Poupart and C. Boutilier. Value-Directed Compression of POMDPs. In *Neural Information Processing Systems*, 2002.

[P. Poupart and C. Boutilier, 2004] P. Poupart and C. Boutilier. VDCBPI: an approximate scalable algorithm for large POMDPs. In *Neural Information Processing Systems*, 2004.

[R. Brafman, 1997] R. Brafman. A heuristic variable grid solution method for POMDPs. In *Proc. of the 15th National Conf. on Artificial Intelligence*, 1997.

[R. Zhou and E.A. Hansen, 2001] R. Zhou and E.A. Hansen. An Improved Grid-Based Approximation Algorithm for POMDPs. In *Proc. of the 17th International Joint Conf. on Artificial Intelligence*, 2001.

[S. Ross *et al.*, 2007] S. Ross, B. Chaib-draa, and J. Pineau. Bayes adaptive POMDP. *Neural Information Processing Systems*, 2007.

[T. Smith and R. Simmons, 2004] T. Smith and R. Simmons. Heuristic Search Value Iteration for POMDPs. In *Proc. of the 20th International Conf. on Uncertainty In Artificial Intelligence*, 2004.

[T. Smith and R.G. Simmons, 2005] T. Smith and R.G. Simmons. Point-based POMDP algorithms: improved analysis and implementation. In *Proc. of the 21th International Conf. on Uncertainty In Artificial Intelligence*, 2005.

[T. Smith *et al.*, 2007] T. Smith, D.R. Thompson, and D.S. Wettergreen. Generating exponentially smaller POMDP models using conditionally irrelevant variable abstraction. In *Proc. of the 17th International Conf. on Automated Planning and Scheduling*, 2007.

[Verma *et al.*, 2003] V. Verma, R. Simmons, and S. Thrun. Variable resolution particle filter. In *IJCAI-2003*, 2003.

[W. Lovejoy, 1991] W. Lovejoy. Computationally feasible bounds for partially observable Markov decision processes. *Operations Research*, 39:162–175, 1991.

[W. Weibull, 1951] W. Weibull. A statistical distribution function of wide applicability. *Journal of Applied Mechanics*, 18:293–297, 1951.

[Y. Virin *et al.*, 2007] Y. Virin, G. Shani, S.E. Shimony, and R. Brafman. Scaling Up: Solving POMDPs through Value Based Clustering. In *Proc. of the 22d National Conf. on Artificial Intelligence*, 2007.