Risk aversion in Markov decision processes*

Lars Relund Nielsen Anders Ringgaard Kristensen

Department of Large Animal Sciences, The Royal Veterinary and Agricultural University, Grønnegårdsvej 8, DK-1870 Frederiksberg C, Denmark, e-mail: lars@relund.dk.

Abstract

The majority of the work in the area of Markov decision processes (MDPs) has focused on optimization criteria that are based on expected values of the rewards or costs. However, such risk-neutral approaches are not always applicable and expressive enough.

In this paper we present the preliminary results of a research project focussing on incorporating risk into MDPs by modelling MDPs using directed hypergraphs. Two risk measures are considered, namely, the variance of the total discounted reward given a policy and the expected total risk when assuming a separate risk for each time, state and action.

First, we consider recursive equations for calculating the variance/risk or maximum risk. Second, it is shown how a state-expanded directed hypergraph can be used to model a finite-horizon MDP. Here a policy corresponds to a so called hypertree. As a result, the optimal policy under e.g. the expected total cost criterion can be found solving a shortest hypertree problem on the state-expanded hypergraph. Finally, bicriterion solution techniques for directed hypergraphs are used to calculate the trade-off between risk and cost.

1 Introduction

MDPs model sequential decision-making problems. At a specified point in time, a decision maker observes the state of a system and chooses an action. The action choice and the state produce two results: the decision maker incurs an immediate reward or cost, and the system evolves probabilistically to a new state at a subsequent discrete point in time. At this subsequent point in time, the decision maker faces a similar problem. The goal is to find an optimal policy of choosing actions (dependent on the observations of the state) which is minimal with respect to a certain criterion.

The majority of the work in the area of Markov decision processes (MDPs) has focused on optimization criteria that are based on expected values of the rewards or costs, see e.g. Howard (1960); Puterman (1994). However, such risk-neutral approaches are not always applicable and expressive enough.

*This research was supported by a grant from SNS - the Nordic Forest Research Co-operation Committee

A risk-sensitive criterion has been addressed by e.g. Cavazos-Cadena (2003); Coraluppi and Marcus (1999) and Bielecki et al. (1999). Here an exponential utility function is assumed with risk sensitivity λ and the expected utility is maximized. This criterion my not be suitable for all practical applications since the decision maker must know the risk sensitivity parameter λ. Moreover, λ is constant for the MDP.

We consider a different way of modelling risk. Two risk measures are considered, namely, the variance of the total discounted cost given a policy and the total risk when assuming a separate risk for each time, state and action.

Directed hypergraphs are an extension of directed graphs and undirected hypergraphs introduced by Berge (1973). The concept of a hypertree and a shortest hypertree was introduced by Nguyen and Pallottino (1989) and later the definition of a hypertree in a directed hypergraph and a general formulation of the shortest hypertree problem were given by Gallo et al. (1993). For a general overview on directed hypergraphs see Ausiello et al. (2001). Recently, the study of directed hypergraphs has become an important aspect in finding optimal strategies/paths in stochastic time-dependent networks, see Nielsen (2004b); Nielsen et al. (2004); Pretolani (2000). Moreover, algorithms to solve bicriterion problems in stochastic time-dependent networks have been developed, see Nielsen (2004b); Nielsen et al. (2003).

By having a look on the hypergraph model for stochastic time-dependent networks it is apparent that, hypergraphs also can be used to model finite-horizon MDPs. However, to the authors' knowledge no one has considered this way of modelling finite-horizon MDPs. Here a MDP can be modelled using a state-expanded directed hypergraph and the problem of finding the optimal policy under different optimality criteria can be formulated as a shortest hypertree problem. Furthermore, we can used bicriterion solution techniques for directed hypergraphs to calculate the trade-off between the risk and cost.

The paper is organized as follows. Finite-horizon MDPs are introduced in Section 2 together with recursive equations for different criteria. Section 3 presents the hypergraph model for MDPs together with results on how the best policy can be found. Bicriterion solution techniques are discussed in Section 4. Finally, conclusions and directions for further research are pointed out in Section 5.

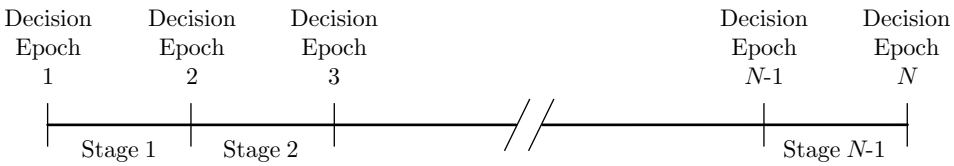

Figure 1: Decision epochs and stages.

2 Finite-horizon Markov decision processes

We consider a finite-horizon Markov decision process with $\{1, ..., N\}$ decision epochs and $N-1$ *stages* (see Figure 1). That is, the decision for epoch number n is taken at the beginning of stage n.

At stage n the system occupies a *state*. We denote the set of finite system states S_n. Given the decision maker observes state $s \in S_n$ at stage n, he may choose an *action* a from the set of finite allowable actions $A_{s,n}$ generating *cost* $c_n(s,a)$ (a reward if negative). Moreover, let $0 < \lambda_n(s,a) < 1$ denote the corresponding discount factor and let $p_n(\cdot \mid s,a)$ denote the *probability distribution* or *transition probabilities* of obtaining states $s' \in S_{n+1}$ at stage $n+1$.

Since no decision is made at the end of stage $N-1$, the cost at this time point is a function of the state $s \in S_N$ denoted $c_N(s, a_N)$ which is often referred to as the *salvage cost* or *scrap cost*. Here a_N denotes a deterministic (dummy) action.

A *deterministic Markovian decision rule* at stage n is a function $\mathfrak{d}_n : S_n \to A_{s,n}$ which specify the action choice given state s at stage n. It is called deterministic because the function chooses an action with certainty and Markovian (memoryless) since it depends only on the current system state. We let D_n denote the set of possible decision rules at stage n. Markovian decision rules are a subset of more general rules where the action may depend on the past history of the system and actions may not be chosen with certainty but rather according to a probability distribution.

A *policy* or *strategy* specifies the decision rule to be used at all stages and provide the decision maker with a plan of which action to take given stage and state. That is, a policy δ is a sequence of decision rules, $\delta = (\mathfrak{d}_1, ..., \mathfrak{d}_N)$ with $\mathfrak{d}_n \in D_n$ for $n = 1, ..., N$.

Let X_n denote the state of the system at stage n, i.e. X_n is a random variable taking values in S_n. Furthermore, let $Y_n = \mathfrak{d}_n(X_n)$ denote the action chosen given

X_n and policy $\delta = (\mathfrak{d}_1, ..., \mathfrak{d}_N)$. *The total discounted cost* for stage $n, ..., N$ is then

$$C_n^{\delta} = \sum_{i=n}^{N} \left(\prod_{j=n}^{i-1} \lambda_j \left(X_j, Y_j \right) \right) c_i \left(X_i, Y_i \right) \tag{1}$$

If the value of X_n is known with probability one, i.e. $X_n = s$, we denote the total discounted cost $C_n^{\delta}(s)$.

Note that two costs may be considered for each $s \in S_n$ and action $a \in A_{s,n}$. Here the first cost $c_n(s, a)$ may be equal to the *economic cost* of choosing action a and the second cost $r_n(s, a)$ may denote *the risk* of choosing action a, where the risk may be a relative value given by external experts or a value calculated mathematically etc. Hence *the total risk* for stage $n, ..., N$ is defined as

$$R_n^{\delta} = \sum_{i=n}^{N} r_i \left(X_i, Y_i \right) \tag{2}$$

2.1 Optimality criteria

Assume that the decision maker seeks a policy prior to knowing the initial state. We let $p_0(s)$ denote the probability of starting in state $s \in S_1$. This corresponds to defining a policy to $\delta = (\mathfrak{d}_0, \mathfrak{d}_1, ..., \mathfrak{d}_N)$ where $\mathfrak{d}_0$ is the decision rule corresponding to a deterministic dummy action a_0. That is, we define stage zero with $S_0 = \{s_0\}$ where s_0 represents the system before the state of the system at stage one is known and let $c_0(s_0, a_0)$ denote the cost. Moreover, we set $p_0(s' \mid s_0, a_0) = p_0(s')$.

Optimality criteria can now be defined using (1) and (2), e.g *the expected total discounted cost* given policy δ is

$$ETC^{\delta} = \mathbb{E}\left(C_0^{\delta}(s_0) \right) \tag{3}$$

and *the expected total risk* given policy δ is

$$ETR^{\delta} = \mathbb{E}\left(R_n^{\delta}(s_0) \right) \tag{4}$$

Note that from a mathematical point of view the economic cost (3) and risk measure (4) define the same recursive equations (the discount factor λ_n is set to one for each stage, state and action in (4)). In the following we will only consider (3). It is well known that the policy δ which minimize (3) is a deterministic Markovian policy, see e.g. Puterman (1994, chap. 4). Let $u_n^{\delta}(s)$ denote the expected total cost of policy δ at stage $n, ..., N$ given state s at stage n. $u_n^{\delta}(s)$ can be found using the

recursive equations, Bellman (1957).

$$u_n^{\delta}(s) = \begin{cases} c_n(s,a) + \lambda_n(s,a) \sum_{s' \in S_{n+1}} p(s' \mid s,a)\, u_{n+1}^{\delta}(s') & n < N \\ c_N(s, a_N) & n = N \end{cases} \tag{5}$$

That is, the optimal policy can be found by analyzing a sequence of simpler inductively defined single-stage problems. This is often referred to as *value iteration* or *dynamic programming.*

Another risk measure can be used by considering the *variance of the total discounted cost*

$$VTC^{\delta} = \mathbb{V}\left(C_0^{\delta}(s_0)\right) \tag{6}$$

Recursive equations for the variance of the total discounted cost (6) are found as follows

$$\begin{aligned} \mathbb{V}\left(C_n^{\delta}(s)\right) &= \mathbb{V}\left(c_n(s,a) + \lambda_n(s,a)\, C_{n+1}^{\delta} \mid X_n = s\right) \\ &= \lambda_n(s,a)^2\, \mathbb{V}\left(C_{n+1}^{\delta} \mid X_n = s\right) \\ &= \lambda_n(s,a)^2\, \mathbb{E}\left(\left(C_{n+1}^{\delta}\right)^2 - \mathbb{E}\left(C_{n+1}^{\delta}\right)^2 \mid X_n = s\right) \end{aligned}$$

Conditioning on $X_{n+1} = s'$ we get

$$\mathbb{V}\left(C_n^{\delta}(s)\right) = \lambda_n(s,a)^2 \sum_{s'} p(s' \mid s,a)\, \mathbb{E}\left(\begin{array}{c} C_{n+1}^{\delta}(s')^2 - \mathbb{E}\left(C_{n+1}^{\delta}(s')\right)^2 + \\ \mathbb{E}\left(C_{n+1}^{\delta}(s')\right)^2 - \left(\mathbb{E}\left(C_{n+1}^{\delta}\right)^2 \mid X_n = s\right) \end{array} \right) \tag{7}$$

Note that random variable Y with values $y_i = Y(s_i) = \mathbb{E}\left(C_{n+1}^{\delta}(s_i)\right)$, $s_i \in S_{n+1}$ and probability density $P(Y = Y(s')) = p(s' \mid s,a)$ satisfy that

$$\mathbb{E}(Y) = \mathbb{E}\left(C_{n+1}^{\delta} \mid X_n = s\right)$$

Hence

$$\begin{aligned} &\sum_{s'} p(s' \mid s,a) \left(\mathbb{E}\left(C_{n+1}^{\delta}(s')\right)^2 - \left(\mathbb{E}\left(C_{n+1}^{\delta}\right)^2 \mid X_n = s\right)\right) \\ &= \sum_{y_i} p(y_i \mid s,a) \left(y_i^2 - \mathbb{E}(Y)^2\right) \\ &= \mathbb{V}(Y) = E\left((Y - E(Y))^2\right) \\ &= \sum_{y_i} p(y_i \mid s,a) \left(y_i - \mathbb{E}(Y)\right)^2 \end{aligned}$$

Substituting y_i and Y we get

$$\sum_{s'} p(s' \mid s,a) \left(\mathbb{E}\left(C_{n+1}^{\delta}(s')\right) - \left(\mathbb{E}\left(C_{n+1}^{\delta}\right) \mid X_n = s\right)\right)^2$$

Moreover,

$$\mathbb{V}\left(C_{n+1}^{\delta}(s')\right) = \mathbb{E}\left(\left(C_{n+1}^{\delta}(s')\right)^2 - \mathbb{E}\left(C_{n+1}^{\delta}(s')\right)^2\right)$$

As a result (7) becomes

$$\mathbb{V}\left(C_{n}^{\delta}(s)\right) = \lambda_n(s,a)^2 \sum_{s'} p(s' \mid s,a)\left(\mathbb{V}\left(C_{n+1}^{\delta}(s')\right) + \left[\mathbb{E}\left(C_{n+1}^{\delta}(s')\right) - U_{n+1}\right]^2\right)$$

with U_{n+1} equal to

$$\begin{aligned} U_{n+1} &= \left(\mathbb{E}\left(C_{n+1}^{\delta}\right)^2 \mid X_n = s\right) \\ &= \sum_{s'} p(s' \mid s,a)\,\mathbb{E}\left(C_{n+1}^{\delta}(s')\right) = \sum_{s'} p(s' \mid s,a)\, u_{n+1}^{\delta}(s') \end{aligned}$$

Hence by letting $v_n^{\delta}(s) = \mathbb{V}\left(C_n^{\delta}(s)\right)$, we have the following recursive equations for the variance of the total discounted cost.

$$v_n^{\delta}(s) = \begin{cases} \lambda_n(s,a)^2 \sum_{s'} p(s' \mid s,a)\left(v_{n+1}^{\delta}(s') + \left[u_{n+1}^{\delta}(s') - U_{n+1}\right]^2\right) & n < N \\ 0 & n = N \end{cases} \tag{8}$$

3 A hypergraph model for finite-horizon Markov decision processes

A *directed hypergraph* is a pair $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = (v_1, ..., v_{|\mathcal{V}|})$ is the set of *nodes*, and $\mathcal{E} = (e_1, ..., e_{|\mathcal{E}|})$ is the set of *hyperarcs*. A hyperarc $e \in \mathcal{E}$ is a pair $e = (T(e), h(e))$, where $T(e) \subset \mathcal{V}$ denotes the set of *tail* nodes and $h(e) \in \mathcal{V} \setminus T(e)$ denotes the *head* node. Note that a hyperarc has exactly one node in the head, and possibly several nodes in the tail. We denote by

$$FS(v) = \{e \in \mathcal{E} \mid v \in T(e)\}, \quad BS(v) = \{e \in \mathcal{E} \mid v = h(e)\}$$

the *forward star* and the *backward star* of node v, respectively. A directed hypergraph $\tilde{\mathcal{H}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$ is a *subhypergraph* of $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, if $\tilde{\mathcal{V}} \subseteq \mathcal{V}$ and $\tilde{\mathcal{E}} \subseteq \mathcal{E}$. A subhypergraph is *proper* if at least one of the inclusions is strict.

We define a state-expanded hypergraph for a finite-horizon MDP as follows.

Definition 1 Let the *state-expanded hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ be obtained by defining the node and hyperarc set as follows

$$\mathcal{V} = \{v_{s,n} \mid n = 0, ..., N, s \in S_n\} \cup \{v_{N+1}\}$$

(n,s)	$(1,1)$	$(1,2)$	$(2,1)$	$(2,2)$	$(3,1)$	$(3,2)$
$c_n(s,a)$	-70	-50	-70	-50	-70	-50
s'	$\{1,2\}$	$\{2,3\}$	$\{1,2\}$	$\{2,3\}$	$\{1,2\}$	$\{2,3\}$
$p_n(\cdot \mid s,a)$	$\{\frac{6}{10},\frac{4}{10}\}$	$\{\frac{6}{10},\frac{4}{10}\}$	$\{\frac{5}{10},\frac{5}{10}\}$	$\{\frac{5}{10},\frac{5}{10}\}$	$\{\frac{2}{10},\frac{8}{10}\}$	$\{\frac{2}{10},\frac{8}{10}\}$

Table 1: Input data for the problem Example 1 given action *nmt*.

$$\mathcal{E} = \{e_{a,s,n} \mid n = 0, ..., N-1, s \in S_n, a \in A_{s,n}\} \cup \{e_{s,N} \mid s \in S_N\}$$

with

$$e_{a,s,n} = \left(\{v_{s',n+1} \mid s' \in S_{n+1},\, p_n(s' \mid s,a) > 0\}, v_{s,n}\right), \quad e_{s,N} = \left(\{v_{N+1}\}, v_{s,N}\right)$$

The following example illustrates how the state-expanded hypergraph is created.

Example 1 We consider a simple machine replacement problem. A machine may be in three states: good, average and not working. Given the machine state we may maintain the machine. In this case the state of the machine will be good at the next decision epoch. Otherwise the machines state will not be better at next decision epoch. The machine is always replaced after 4 decision epochs. Furthermore, if the machine is not working then the machine may be replaced before decision epoch 4. Finally, when the machine is bought it may be either in state good or average.

The problem of when to replace the machine can be modelled using a Markov decision process with $N = 4$ decision epochs. We use system states *good* (1), *average* (2) and *not working* (3) together with actions *buy*, *maintain* (*mt*), *no maintenance* (*nmt*) and *replace* (*rep*). The system state sets S_n and action sets $A_{s,n}$ becomes

$$S_n = \begin{cases} \{s_0\} & n = 0 \\ \{1,2\} & n = 1 \\ \{1,2,3\} & n = 2 \\ \{1,2,3,4\} & n = 3,4 \end{cases}, \quad A_{s,n} = \begin{cases} \{buy\} & n = 0,\ s = s_0 \\ \{mt, nmt\} & n = 1,2,3,\ s = 1,2 \\ \{mt, rep\} & n = 2,3,\ s = 3 \\ \{rep\} & n = 4,\ s = 1,2,3 \end{cases}$$

The state set S_0 contains a single dummy state s_0 representing the machine before knowing its initial state and $A_{s_0,0}$ containing the deterministic action *buy*. Moreover, $A_{s,4}$, $s \in S_4$ contains the deterministic action *rep*.

The cost of buying the machine is 100 with $p_0(1) = 0.7$ and $p_0(2) = 0.3$. The reward (scrap value) of replacing a machine is 30, 10 and 5 in state 1, 2 and 3, respectively. The reward of the machine given action *mt* becomes 55, 40 and 30 given state 1, 2 and 3, respectively. Moreover, the system enters state 1 with probability 1 at the next stage. Finally, Table 1 show the cost, transition states and probabilities given action *nmt*.

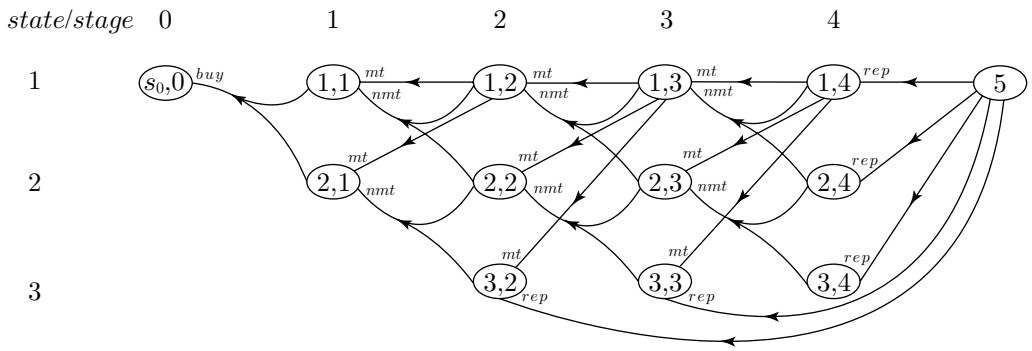

Figure 2: The state-expanded hypergraph.

The state-expanded hypergraph $\mathcal{H}$ is shown in Figure 2 with the subscript of node $v_{s,n}$ shown in each node and action a corresponding to the hyperarc $e_{a,s,n}$ shown beside it. $\mathcal{H}$ contains a hyperarc $e_{a,s,n}$ for each possible action a given stage n and $s \in S_n$ and a node $v_{s,n} \in \mathcal{V}$ for all stages n and states $s \in S_n$. The head node of a hyperarc corresponds to the state of the system before action a is taken at the tail nodes to the possible system states after action a is taken. Furthermore, $\mathcal{H}$ contains a dummy node v_{N+1} which may be considered as the final system state representing the system after the machine has been replaced. Note this is often modelled using a dummy state *replaced* at each stage where the system stays in this state if it first enters it. However, this is avoided in the state-expanded hypergraph. Moreover, $\mathcal{H}$ contain arcs $e_{s,N}$, with tail node v_{N+1}, corresponding to the deterministic action *rep* are used. Finally, note that the direction of the hyperarcs are backward in time. ■

Observe that there is a one to one correspondence between a policy δ and a *predecessor function* $\mathfrak{g} : \mathcal{V} \to \mathcal{E}$. Indeed, choosing $\mathfrak{g}(v_{s,n}) = e_{a,s,n}$ is equivalent to choosing $\mathfrak{d}_n(s) = a$. Moreover, $\mathfrak{g}(v_{s,N}) = e_{s,N}$ is the only possible predecessor for node $v_{s,N}$ indicating that only a deterministic dummy action a_N is possible at stage N. The same holds for node $v_{s_0,0}$.

Predecessor function $\mathfrak{g} : \mathcal{V} \backslash \{v_{N+1}\} \to \mathcal{E}$ define a *hypertree* (a subhypergraph of $\mathcal{H}$) with root v_{N+1} (see Nielsen (2004a) for a explicit definition of a hypertree). As a result an optimal policy can be found by finding a shortest hypertree on $\mathcal{H}$ using a specific weighting function corresponding to the recursive equations given in Section 2.

The weighting function of a hypertree is defined as follows. Assume that each hyperarc e is assigned nonnegative real weight vector $w(e) = (w_1(e), ..., w_L(e))$. Given a predecessor function $\mathfrak{g}$, a weighting function W is a node function assigning real weights $W(u)$ to all nodes in $\mathcal{H}$. We shall restrict ourselves to *additive weighting*

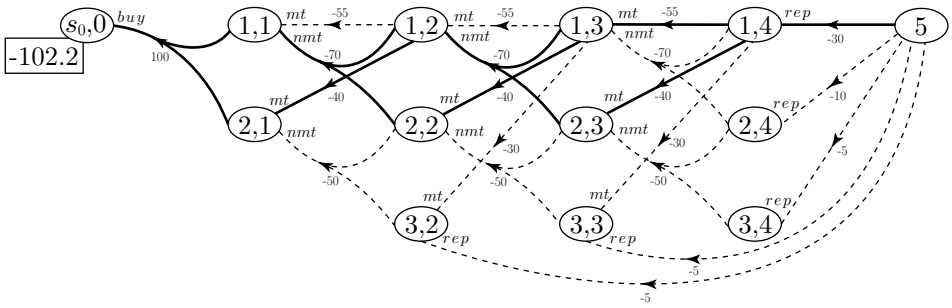

Figure 3: The optimal policy.

functions introduced by Gallo et al. (1993), defined by the recursive equations:

$$W(v) = \begin{cases} 0 & v = o \\ \mathfrak{l}\left(w(\mathfrak{g}(v))\right) + \mathfrak{f}(\mathfrak{g}(v)) & v \in \mathcal{V}_\pi \setminus \{o\} \end{cases} \tag{9}$$

Here $\mathfrak{l}(\cdot)$ denote a non-decreasing function of $w(e)$ and $\mathfrak{f}(\cdot)$ a non-decreasing function of the weights in the nodes of $T(e)$. Furthermore, let $m_e(v)$ denote a nonnegative *multiplier* defined for each hyperarc e and node v.

Finding a shortest hypertree can be viewed as a natural generalization of the shortest path problem and consists in finding the minimum weight for all nodes in $\mathcal{H}$. If $\mathcal{H}$ is acyclic (which is the case here) and the weighting function is additive a fast polynomial algorithm exist (see (Gallo et al., 1993)). We illustrate the approach by finding the policy minimizing the expected total discounted cost.

Example 1 (continued) For simplicity we assume that the discount rate λ_n equals one for all stages, states and actions. Assign weights to the hyperarcs of $\mathcal{H}$ as follows

$$w_1(e) = \begin{cases} c_n(s,a) & e = e_{a,s,n} \\ c_N(s,a_N) & e = e_{s,N} \end{cases}$$

Moreover, for each hyperarc e assign multipliers

$$m_e(v) = \begin{cases} p_n(s' \mid s,a) & e = e_{a,s,n},\ v = v_{s',n+1} \in T(e) \\ 1 & e = e_{s,N},\ v = v_{N+1} \end{cases}$$

Then the optimal policy minimizing the expected total cost can be found by finding a hypertree minimizing the recursive equations (9) with

$$\mathfrak{l}(w(e)) = w_1(e), \quad \mathfrak{f}(e) = \sum_{v \in T(e)} m_e(v)\, W(v)$$

The hypertree corresponding to the optimal policy is shown in bold in Figure 3. The expected total reward is 102.2. Note each time the machine reaches the *average* state it is maintained. ■

Similar functions $\mathfrak{l}(\cdot)$ and $\mathfrak{f}(\cdot)$ can be redefined so they correspond to the recursive equations for the variance (7).

Note that, value iteration could have been used to find the optimal policy above. However, modelling the MDP using the state-expanded hypergraph provide us with efficient ways calculate the optimal policy and to store the MDP. More important, specialized algorithms for directed hypergraphs can now be used on the state-expanded hypergraph. That is, we can now find the K best policies ranking the policies in nondecreasing order of e.g. the expected total cost Nielsen (2004a) and used bicriterion optimization techniques for directed hypergraphs to find the trade-off between to different criteria.

4 Bicriterion optimization

By representing a MDP using the state-expanded hypergraph we can calculate the optimal policy under various criteria such as the expected total discounted cost or the variance of the total discounted cost etc. Furthermore, by using bicriterion optimization techniques for directed hypergraphs it is now possible to calculate the trade-off between e.g. the expectation and the variance of the total discounted cost. That is, we are interested in finding *efficient policies* where the weight of one criterion cannot be reduced without increasing the weight of the second criterion.

Bicriterion optimization techniques for directed hypergraphs have been addressed by Nielsen (2004b). A *two-phase approach* is used which splits the search for efficient policies into two phases. In phase one the frontier is determined using a NISE approach (see Cohen (1978)); this defines the triangles in which further efficient policies may be found. Phase two proceeds to search the triangles one at a time using a K best policies procedure, see Nielsen (2004a).

Computationally, the two-phase approach requires to solve shortest hypertree and K shortest hypertree problems with respect to a *parametric weight*, that is a linear combination of the two criteria.

5 Conclusions and further work

This paper presented the preliminary results of a research project focussing on incorporating risk into MDPs by modelling MDPs using directed hypergraphs. First, recursive equations for finding the variance of the total discounted cost was derived. Next, it was pointed out that a MDP can be modelled using a state-expanded directed hypergraph $\mathcal{H}$. Moreover, an optimal policy can be found by finding a shortest hypertree on $\mathcal{H}$. As a result bicriterion optimization techniques for directed hypergraphs can be used to calculate the trade-off between e.g. risk and cost.

The research project still has may open questions which have to be examined during the next months. For instance, the bicriterion optimization techniques for directed hypergraphs have to be fine-tuned to the variance and expectation criteria. Moreover, it must be checked whether the results for finite-horizon MDPs can be extended to infinite-horizon MDPs.

References

Ausiello, G., P. Franciosa, and D. Frigioni (2001). Directed hypergraphs: Problems, algorithmic results, and a novel decremental approach. *Lecture Notes in Computer Science 2202*, 312–328.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

Berge, C. (1973). *Graphs and hypergraphs*. North-Holland.

Bielecki, T., D. Hernandez-Hernandez, and S. Pliska (1999). Risk sensitive control of finite state markov chains in discrete time, with applications to portfolio management. *Mathematical Methods of Operations Research 50*(2), 167–188.

Cavazos-Cadena, R. (2003). Solution to the risk-sensitive average cost optimality equation in a class of markov decision processes with finite state space. *Mathematical Methods of Operations Research 57*(2), 263–285.

Cohen, J. (1978). *Multiobjective Programming and Planning*. New York: Academic Press.

Coraluppi, S. and S. Marcus (1999). Risk-sensitive and minimax control of discrete-time, finite-state markov decision processes. *Automatica 35*(2), 301–309.

Gallo, G., G. Longo, S. Pallottino, and S. Nguyen (1993). Directed hypergraphs and applications. *Discrete Applied Mathematics 42*, 177–201.

Howard, R. (1960). *Dynamic Programming and Markov processes.* Cambridge, Massachusetts: The M.I.T. Press.

Nguyen, S. and S. Pallottino (1989). Hyperpaths and shortest hyperpaths. In *Combinatorial optimization (Como, 1986)*, Volume 1403 of *Lecture Notes in Math*, pp. 258–271. Springer.

Nielsen, L. (2004a, August). Finding the K best policies in finite-horizon Markov decision processes.

Nielsen, L. (2004b). *Route Choice in Stochastic Time-Dependent Networks.* Ph. D. thesis, Department of Operations Research, University of Aarhus.

Nielsen, L., K. Andersen, and D. Pretolani (2003). Bicriterion shortest hyperpaths in random time-dependent networks. *IMA Journal of Management Mathematics 14*(3), 271–303.

Nielsen, L., K. Andersen, and D. Pretolani (2004). Finding the K shortest hyperpaths. *To appear in Computers & Operations Research.*

Pretolani, D. (2000). A directed hypergraph model for random time-dependent shortest paths. *European Journal of Operational Research 123*, 315–324.

Puterman, M. (1994). *Markov Decision Processes.* Wiley Series in Probability and Mathematical Statistics. Wiley-Interscience.