Stochastic Dynamic Programming Based Approaches to Sensor Resource Management

R.B. Washburn, M.K. Schneider, J.J. Fox
Fusion Technology and Systems Division
Alphatech, Inc.
Burlington, MA
{bob.washburn, michael.schneider, john.fox}@alphatech.com

Abstract – This paper describes a stochastic dynamic programming based approach to solve Sensor Resource Management (SRM) problems such as occur in tracking multiple targets with electronically scanned, multi-mode radar. Specifically, it formulates the SRM problem as a stochastic scheduling problem and develops approximate solutions based on the Gittins index rule. Novel results include a hybrid state stochastic model for the information dynamics of tracked targets, an exact index rule solution of the SRM problem for a simplified tracking model, and use of approximate dynamic programming to extend the index rule solution to more realistic models.

Keywords: Sensor resource management, dynamic programming, Gittins index, multi-armed bandit.

1 Introduction

Modern sensor suites contain sensors that are agile, allowing fast redirection (e.g., electronic scanning) and fast switching between modes (e.g., radar modes such as SAR and MTI), giving a large number of control options. However, allocation of sensor resources is also subject to a variety of constraints: there may be many things to look at (complex environments with multiple targets), limited time to sense targets because of deadlines on the usefulness of information (urgent requests) or exposure to threat environments, sensors may have a limited field of view, and so on. Developing sensor control algorithms that satisfy such constraints and maximize the value of information obtained is the goal of Sensor Resource Management (SRM) research.

Stochastic dynamic programming [1] provides a principled, systematic approach to formulating and solving SRM problems [8]. However, its application requires paying the price of developing mathematical models for the problem formulation and then finding efficient algorithmic solutions. In this paper we describe new results in developing efficient solutions based on stochastic dynamic programming to solve SRM problems such as occur in tracking multiple targets with electronically scanned, multi-mode radar.

Specifically, we formulate the SRM problem as a stochastic scheduling (or multi-armed bandit) problem and develop approximate solutions based on the Gittins index rule [2]. Although there has been much research on index rules [7], [6], [3], [9], [5], there appear to be no applications to SRM except for the recent work by Krishnamurthy and Evans [4]. The reason for this is that the SRM problem, formulated as stochastic scheduling, doesn’t usually satisfy the conditions necessary for the existence of an index rule solution. Thus, one is forced to develop approximations (e.g., the “restless bandit” work of [9], [5]). In this paper we show how to relax one of the “usual” conditions and to develop index rule-based SRM algorithms for kinematic tracking.

The new contributions of this paper include:

1. Stochastic scheduling formulation of SRM and a corresponding extension of the index rule solution to more general reward functions, more suitable for SRM problems.

2. Hybrid state stochastic model of information dynamics for tracking problems, which facilitates efficient solutions of stochastic dynamic programs and alleviates the “curse of dimensionality.”

3. Exact index rule solution for a simple tracking model. This solution provides a basis for efficient approximations of more realistic tracking models.

4. Lookahead approximation of the stochastic dynamic program solution based on an index rule solution, applicable to more realistic tracking models.

Section 2 formulates the SRM problem as a stochastic scheduling problem and shows how to extend the index rule solution to more general types of reward functions. This section also discusses generalization of the formulation to controlling sensors with variable dwell times and
selectable modes. Section 3 describes hybrid state models of information dynamics to use for tracking problems. These models use discrete-valued random variables to drive a continuous-valued information state. This approach helps reduce the dimensionality of the resulting stochastic dynamic programming equations. In Section 4 we find the optimal index rule solution for a simple version of the tracking problem and use this solution in Section 5 as the basis for approximate solutions of more complex models. Section 5 also describes numerical results of the lookahead approximation for a multi-target tracking example.

2 SRM Problem Formulation

2.1 Stochastic Scheduling Formulation

We formulate SRM as a stochastic control problem in which sensor actions \( u(t) \) are the controls and the controlled state, denoted \( x(t) \), is an information state. Conceptually, \( x(t) \) denotes the accuracy of the information in the track. Most generally, it can be the conditional probability of the target state given previous observations (as in [4]). In our problem \( x(t) \) will be an estimate of the track error covariance. In this paper we will assume that the sensor action applies to one target at a time and that time is discrete (e.g., \( t = 1, 2, \ldots \)). The control \( u(t) = i \) indicates which target \( i \) to look at time \( t \). The information state is initially \( x_i(0) = \xi_i \), and \( x_i(t+1) \) is given by a stochastic equation

\[
x_i(t+1) = f_i(x_i(t), w_i(t), u(t))
\]  

(1)

for \( i = 1, \ldots, n \), where \( w_i(t) \) are random variables representing uncertainties in target dynamics and sensor measurements. We assume that \( w_i(t) \) are independent random variables with distribution \( p_i(w_i) \).

At each time \( t \) we must determine an action \( u(t) \) for the sensor to take depending only on the information state

\[
x(t) = (x_1(t), \ldots, x_n(t))
\]  

(2)

of all targets. The SRM problem is to determine \( u(t) \) as a function of \( x(t) \) which maximizes the expected value of the total discounted reward, namely

\[
J(\xi) = \max_{\phi} \mathbb{E} \left\{ \sum_{k=0}^{\infty} \alpha^k \sum_{i=1}^{n} R_i(x_i(t))dt|x(0) = \xi \right\}
\]  

(3)

where the optimization is taken over all stationary control policies \( \phi \) (i.e., \( u(t) = \phi(x(t)) \)). The parameter \( \alpha \) is the discount factor. The dynamic programming equation is

\[
J(\xi) = \max_u \left\{ \sum_{i=1}^{n} R_i(\xi) + \alpha \int J(f(\xi, w, u)) \, p(w) \, dw \right\}
\]  

(4)

where

\[
p(w) = \prod_{i=1}^{n} p_i(w_i).
\]  

(5)

The optimal control policy is given by finding the action \( u = \phi(\xi) \) which achieves the maximum \( J(\xi) \).

2.2 SRM and Index Rules

In the conventional stochastic scheduling problem (for which there are index rule solutions), one receives the discounted reward

\[
\alpha^t R_u(t) (x_u(t), u(t))
\]  

(6)

at time \( t \). In the SRM problem one receives the reward

\[
\alpha^t \sum_{i=1}^{n} R_i(x_i(t)).
\]  

(7)

In the conventional scheduling problem, the state \( x_i(t) \) of a machine evolves via

\[
x_u(t+1) = f_u(t) (x_u(t), w_u(t))
\]  

(8)

and

\[
x_i(t+1) = x_i(t) \text{ if } i \neq u(t).
\]  

(9)

In the SRM problem (8) is still true, where the equation represents the prediction and update of the target track state over one time step. The noise \( w_i(t) \) expresses uncertainty in how the target state actually changes as well as uncertainties in the measurement of the target state. However, if the sensor doesn’t look at the target, the evolution of the state is

\[
x_i(t+1) = g_i(x_i(t), w_i(t)) \text{ if } i \neq u(t)
\]  

(10)

where the equation represents the prediction of the target track state over one time step, without a measurement update. The noise \( w_i(t) \) expresses uncertainty in how the target state changes in one time step. In general, \( g_i(x_i, w_i) \neq x_i \).

Thus, the SRM problem differs from the stochastic scheduling problem in two critical respects:

1. The reward depends on unsensed targets (in the conventional scheduling problem idle machines contribute no reward).

2. The state of an unsensed target can change (in the conventional scheduling problem the state of idle machines remain frozen).

In some SRM problems it is possible to have

\[
g_i(x_i, w_i) = x_i
\]  

(11)

so that the state of unsensed targets remains the same, just as in the scheduling problem. For example, this is the case
if $x_t$ are statistics of a static parameter (e.g., target classification problems, where $x_t$ is a probability vector of possible target types). If (11) is true, we can reformulate the SRM problem with an equivalent reward function so that the reward doesn’t depend on unsensed targets, again just like the scheduling problem, and thus we can get index rules which are optimal solutions of the SRM problem.

**Theorem 1.** Suppose that

$$x_i (t + 1) = f_i (x_i (t), w_i (t), u (t))$$

for $i = 1, \ldots, n$, where $w_i (t)$ are independent for all $i, t$ and $w_i (t)$ has distribution $p_i (w_i)$ for all $i, t$. Suppose that the control $u (t) \in \{1, \ldots, n\}$ and $u (t)$ depends on the state $x (t)$. Suppose that

$$f_i (x_i, w_i, u) = x_i$$

for all $u \neq i$. Define

$$\hat{R}_i (x_i) = \int R_i (f_i (x_i, w_i, i)) p_i (w_i) dw_i - R_i (x_i).$$

Then a policy optimizes

$$E \left\{ \sum_{i=0}^{\infty} \alpha^t \hat{R}_{u(t)} (x_{u(t)} (t)) \mid x(0) \right\}$$

if and only if it optimizes

$$E \left\{ \sum_{i=0}^{\infty} \alpha^t \sum_{i=1}^{n} R_i (x_i (t)) \mid x(0) \right\}.$$  

*Proof.* For any control $u (t)$ show that

$$E \left\{ \sum_{i=0}^{\infty} \alpha^t \hat{R}_{u(t)} (x_{u(t)} (t)) \mid x(0) \right\} = a E \left\{ \sum_{i=0}^{\infty} \alpha^t \sum_{i=1}^{n} R_i (x_i (t)) \mid x(0) \right\} + b$$

for constants $a = (\alpha - 1) > 0, b$.

**Corollary 2.** If the target state remains unchanged when the target is not viewed by the sensor, then there is an index rule which is an optimal solution of the SRM problem. That is, there are functions $m_i (x_i)$, for each target $i$, such that the optimal SRM policy is given by

$$u (t) = \ar \max_i \{ m_i (x_i (t)) \}.$$  

If $J_i (x_i, M)$ is the solution of

$$J_i (x_i, M) = \max \left\{ M, \hat{R}_i (x_i) \right\} + \alpha \int J_i (f_i (x_i, w_i, M) p_i (w_i) dw_i,$$

where

$$\hat{R}_i (x_i) = \int R_i (f_i (x_i, w_i)) p_i (w_i) dw_i - R_i (x_i),$$

then the index function is defined by

$$m_i (x_i) = \min \{ M : J_i (x_i, M) = M \}.$$  

*Proof.* Follows from the theorem above and results for finding index functions in [1].

The result shows that the assumption that idle machines do not contribute to the reward is not a critical assumption. As long as the states of idle machines are frozen, the theorem shows that it is possible to find an equivalent reward such that idle machines do not contribute. Thus, ‘frozen states of idle machines’ is the key assumption in order to have an optimal index rule solution to the stochastic scheduling problem. Of course, the frozen state assumption is not realistic for tracking SRM problems. However, one can use index rules as the basis for good suboptimal approximations of the more general problem.

### 2.3 Extension of SRM Formulation

It is necessary to extend the problem formulation to model more realistic situations in which sensors have selectable modes and in which different modes require different amounts of time to complete (e.g., consider the difference between SAR and MTI modes for a radar). In such situations we formulate the scheduling problem so that the control $u (t)$ and the information state $x_i (t)$ are constant over the time interval $\tau (k) \leq t < \tau (k + 1)$. Let $x_i (k)$ denote the information state of target $i$ at stage $k$. Stage $k$ begins at time $\tau (k)$ and ends at time $\tau (k + 1)$. At each stage $k$ we must determine a control $u (k)$ for the sensor to use over the period $\tau (k) \leq t < \tau (k + 1)$. In this case the control $u (k)$ has the form

$$u (k) = (i (k), \theta (k))$$

where $i$ is the target $i = 1, \ldots, n$ to look at and $\theta$ is a sensor mode to use. Time $\tau (0) = 0$, and $\tau (k + 1)$ is given by

$$\tau (k + 1) = \tau (k) + \Delta (\theta (k)),$$

so that the time duration $\Delta (\theta (k))$ of stage $k$ depends on the sensor mode $\theta (k)$ used. This control $u (k)$ depends only on the information state at the beginning of the state, namely,

$$x (k) = (x_1 (k), \ldots, x_n (k)).$$

The information state evolves according to

$$x_i (k + 1) = f (x_i (k), w_i (k), u (k))$$

We can assume that $w_i (k)$ depends only on $x_i (k)$ and $u (k)$ and has conditional distribution $p_i (w_i | x_i, u).$
3.1 Modeling Detection Uncertainty

The SRM problem is to determine $u(t)$ as a function of $x(t)$ which maximizes the expected value of the total discounted reward, namely

$$J(\xi) = \max_{\phi} \left\{ \int_0^\infty e^{-\alpha t} \sum_{i=1}^n R_i(x_i(t)) \, dt \mid x(0) = \xi \right\}$$  \hspace{1cm} (25)

where the optimization is taken over all stationary control policies $\phi$ (i.e., $u(t) = \phi(x(t))$). The parameter $\alpha$ is the discount rate. Because the information state is piecewise constant, the total discounted value is

$$\int_0^\infty e^{-\alpha t} \sum_{i=1}^n R_i(x_i(t)) \, dt = \sum_{k=0}^\infty \frac{e^{-\alpha \Delta(k)} - e^{-\alpha \Delta(k+1)}}{\alpha} \sum_{i=1}^n R_i(x_i(k)).$$

Thus, we obtain an equivalent discrete time problem with the dynamic programming equation

$$J(\xi) = \max_u \left\{ \frac{1-e^{-\alpha \Delta(u)}}{\alpha} \sum_{i=1}^n R_i(\xi_i) + e^{-\alpha \Delta(u)} \int J(f(\xi, w, u)) p(w|\xi, u) \, dw \right\}$$  \hspace{1cm} (26)

where

$$p(w|\xi, u) = \prod_{i=1}^n p_i(w_i|\xi_i, u).$$  \hspace{1cm} (27)

3 Model of Information Dynamics for Tracking Problems

3.1 Modeling Detection Uncertainty

To design the SRM algorithm we need to model how the information state responds to sensor actions, as modeled by the equation

$$x_i(t+1) = f_i(x_i(t), w_i(t), u(t)).$$  \hspace{1cm} (28)

Modeling the information state as a conditional probability distribution of the target state variable is one possible model, but for many problems it gives a large dimensional stochastic evolution equation and a correspondingly large dimensional dynamic program to solve. Thus, we try to model the information dynamics with smaller number of salient statistics in order to obtain a tractable stochastic dynamic program.

For tracking problems, in which the SRM algorithm is trying to maximize the quality of kinematic tracking for multiple targets, the track error covariance is an effective information state $x_i(t)$ and hybrid state equations, in which $w_i(t)$ takes a finite number of values, provide a low complexity description of the information dynamics.

The simplest instance of the model has $w_i(t)$ just indicate target detections so that $w_i(t)$ takes just two possible values (denote, $w_i(t) = D$ if the target is detected and $w_i(t) = \bar{D}$ if the target isn’t detected). Let $\beta_i$ denote the probability that $i$ is detected, assuming that the sensor looks at target $i$. Then $f_i$ has the following form.

$$f_i(x_i, D, i) = F_i \left( x_i(t)^{-1} + H_i^T R^{-1} H_i \right)^{-1} F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (29)

$$f_i(x_i, \bar{D}, i) = F_i x_i(t) F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (30)

If the sensor doesn’t look at target $i (u \neq i)$, then

$$f_i(x_i, w_i, u) = F_i x_i(t) F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (31)

whether $w_i = D$ or $\bar{D}$. Here $x_i(t)$ denotes an error covariance and the matrices $F_i, G_i, Q_i, H_i, R_i$ model the effect of prediction and update on the error covariance.

3.2 Modeling Association Uncertainty

We can also include the effect of misassociations by considering a four state process $w_i(t)$ as follows. Suppose that $w_i$ can be one of the four states

$$(A, D), (A, \bar{D}), (\bar{A}, D), (\bar{A}, \bar{D})$$  \hspace{1cm} (32)

where $D$ stands for detection, $\bar{D}$ for missed detection, $A$ for association, and $\bar{A}$ for misassociation. Then $f_i(x_i, w_i, u)$ has the following form.

$$f_i(x_i, (A, D), i) = F_i \left( x_i(t)^{-1} + H_i^T R_i^{-1} H_i \right)^{-1} F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (33)

$$f_i(x_i, (A, \bar{D}), i) = F_i x_i(t) F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (34)

$$f_i(x_i, (\bar{A}, D), i) = F_i (x_i(t) + S_i)^{-1} F_i^T + G_i Q_i G_i^T$$  \hspace{1cm} (35)

The idea is that $S_i \geq 0$ models the bias induced by a misassociation. If the sensor doesn’t look at target $i (u \neq i)$, then

$$f_i(x_i, w_i, u) = F_i x_i(t) F_i^T + G_i Q_i G_i^T.$$  \hspace{1cm} (36)

A simple model for $S_i$ is

$$S_i = \frac{a_i^2}{d_i} + 2 K_i \hat{r}_i K_i^T$$  \hspace{1cm} (37)

where $a_i$ is a gate size parameter (e.g., $a_i = 3$ or $a_i = 4$), $d_i$ is the dimension of the measurement, $K_i$ is the Kalman gain

$$K_i = \left( x_i(t)^{-1} + H_i^T R_i^{-1} H_i \right)^{-1} H_i^T R_i^{-1}$$  \hspace{1cm} (38)
and
\[ \hat{R}_i = H_i x_i(t) H_i^T + R_i \] (39)
is the predicted measurement covariance.

The probabilities of the discrete events are given by
\[ \Pr \{ w_i(t) = (A,D) \} = P_{A|D} \beta_i. \] (40)
\[ \Pr \{ w_i(t) = (A,\tilde{D}) \} = P_{A|\tilde{D}} (1 - \beta_i). \] (41)
\[ \Pr \{ w_i(t) = (\tilde{A},D) \} = (1 - P_{A|D}) \beta_i. \] (42)
\[ \Pr \{ w_i(t) = (\tilde{A},\tilde{D}) \} = (1 - P_{A|\tilde{D}}) (1 - \beta_i). \] (43)
For \( d_i = 2 \), the association probability models are given by
\[ P_{A|D} = \frac{1 - \frac{e^{-\frac{\sigma^2}{2}}}{\sigma^2}}{b_i + \frac{1}{2}} \] (44)
and
\[ P_{A|\tilde{D}} = 1 - e^{-\frac{a_i^2}{b_i}}, \] (45)
where
\[ b_i = \rho_i \pi \sqrt{\text{det} \hat{R}_i}, \] (46)
where \( a_i \) is the desired gate size, \( \bar{R} \) is the predicted measurement covariance, and \( \rho_i \) is the expected density of confusers in the measurement gate of target \( i \). Note that \( b_i \) is the expected number of confusers in the one-sigma predicted measurement gate. This is sometimes referred to as the “contention” for the target.

4 Tracking SRM Index Rule
4.1 Target Location Example
Suppose that each target \( i \) has a location and \( x_i(t) \) is the root mean square error of its location estimate. Define the goal set \( G \) to be
\[ G = [0, \delta] \] (47)
for some \( \delta > 0 \). Define the individual target track reward function as
\[ R(x_i) = 1 (x_i \in G) \] (48)
where \( 1 (C) \) denotes the indicator function, equal to 1 if the condition \( C \) is true and equal to 0 otherwise.

The sensor looks at just one target, \( i = u(t) \) at time \( t \). If the sensor looks at target \( i \) and the target is detected, then
\[ x_i(t + 1) = \left( x_i(t)^{-2} + \sigma^{-2} \right)^{-1/2}. \] (49)
Assume that \( \beta \) is the probability of detecting the target, if the sensor looks at it. If the target is not detected or if the sensor doesn’t look at target \( i \), then
\[ x_i(t + 1) = x_i(t). \] (50)
Let \( \xi \) denote the initial location error—i.e., \( x_i(0) = \xi \).

4.2 Equations for Index Function
Transform \( x_i \) to \( z_i = x_i^{-2} \). Then we can express the reward as
\[ R(z_i) = 1 (z_i \in [\delta^{-2}, \infty)) \] (51)
with the evolution equation
\[ f(z_i, w_i) = \begin{cases} z_i + \sigma^{-2} & \text{if target is detected} \\ z_i & \text{if target is not detected} \end{cases}. \] (52)
To find the index rule to a sum of individual target rewards, we need to replace the reward function \( R(z_i) \) with \( \bar{R}(z_i) \) defined by
\[ \bar{R}(z_i) = \int R(f(z_i, w_i)) p_i(w_i) dw_i - R(z_i) \]
\[ = \begin{cases} \beta & \text{if } \delta^{-2} - \sigma^{-2} \leq z < \delta^{-2} \\ 0 & \text{if } z < \delta^{-2} - \sigma^{-2} \text{ or } z \geq \delta^{-2} \end{cases}. \]
To find the index function corresponding to \( \bar{R}(z_i) \), we solve
\[ J_i(z_i, M) = \max \left\{ \begin{array}{ll} M, \beta \cdot 1 (z_i \in [\delta^{-2} - \sigma^{-2}, \delta^{-2}]) + \alpha \beta J_i (z_i + \sigma^{-2}, M) + \alpha (1 - \beta) J_i (z_i, M) \end{array} \right\}. \] (53)
by successive approximation, noting that the value function at any iteration is piecewise constant. The successive approximation iterations converge to the optimal value function
\[ J_i(z_i, M) = \sum_{k > \frac{\delta^{-2}}{\sigma^{-2}}} \Phi_k(M) 1 (z_i \in [a + k \Delta, a + (k + 1) \Delta]) + M 1 (z_i \in [a + \Delta, \infty)) \]
where
\[ \Phi_k(M) = \begin{cases} k^{k+1/\alpha-1} + k^{k+1} M \text{ if } M \leq \frac{k^{k+1/\alpha-1}}{1 - k^{k+1}} \\ 0 \text{ if } M > \frac{k^{k+1/\alpha-1}}{1 - k^{k+1}} \end{cases} \] (54)
for \( k \leq 0 \) and
\[ \kappa = \frac{\alpha \beta}{1 - \alpha (1 - \beta)} < 1. \] (55)

4.3 Index Rule
The index function \( m(z_i) \) is defined as the smallest \( M \) such that
\[ J_i(z_i, M) = M. \] (56)
For \( z_i \in [a + k \Delta, a + (k + 1) \Delta) \), this is the smallest \( M \) such that \( \Phi_k(M) = M \). Thus,
\[ m(z_i) = \frac{k^{k+1/\alpha-1}}{1 - k^{k+1}} \] (57)
for \( z_i \in [a + k \Delta, a + (k + 1) \Delta) \). Similarly, for \( z_i \in [a + \Delta, \infty) \), we have
\[
m(z_i) = 0. \tag{58}
\]
In terms of the parameters \( \delta \) and \( \sigma \), we can express the index function as
\[
m(z_i) = \begin{cases} 
\frac{1}{1 - \sigma^2} & \text{if } \delta^{-2} + (k - 1) \sigma^{-2} \leq z_i \\
\frac{1}{1 - \sigma^2} & \text{if } \delta^{-2} + k \sigma^{-2}, k \leq 0 \\
0 & \text{if } z_i \geq \delta^{-2}
\end{cases}
\]
Thus, the index rule control is
\[
u(t) = \arg \max \{ m(z_i) \}, \tag{60}
\]
which is equivalent to
\[
u(t) = \arg \max \{ z_i(t) : z_i(t) < \delta^{-2} \}. \tag{61}
\]
If we assign different goals \( \delta_i \) as well as different values \( V_i \) to each target, then it is easy to see that the corresponding index functions are \( m_i(z_i) \) given by
\[
m_i(z_i) = \begin{cases} 
V_i \frac{\delta_i^{-1} \alpha^{-1}}{1 - \alpha^2 \beta^2} & \text{if } \delta_i^{-2} + (k - 1) \sigma^{-2} \leq z_i \\
\frac{1}{1 - \sigma^2} & \text{if } \delta_i^{-2} + k \sigma^{-2}, k \leq 0 \\
0 & \text{if } z_i \geq \delta_i^{-2}
\end{cases}
\]
where
\[
\kappa_i = \frac{\alpha \beta_i}{1 - \alpha (1 - \beta_i)} < 1. \tag{63}
\]
4.4 Extension to Higher Dimensions

We can extend the previous scalar results to more general situations where the information state is a covariance matrix. Thus, let \( x_i \) be the target track error covariance for target \( i \). Assume that \( x_i \) evolves according to
\[
f(x_i, w_i, u) = \begin{cases} 
f(x_i) & \text{if } i = i \text{ and } w_i = 1 \\
x_i & \text{otherwise}
\end{cases}
\]
where \( w_i(t) \) is a \( 0-1 \) random variable and
\[
\Pr \{ w_i(t) = 1 \} = \beta_i. \tag{65}
\]
Assume that \( f(x) \) satisfies
\[
f(x) \leq x \tag{66}
\]
in the sense of positive definite matrices \( x \) and \( f(x) \). Define the reward function \( R_i(x_i) \) by
\[
R_i(x_i) = \begin{cases} 
V_i & \text{if } \rho_i(x_i) \leq \delta_i \\
0 & \text{otherwise}
\end{cases}
\]
where \( \delta \) is also monotonic in the sense that
\[
\rho_i(x_i) \leq \rho_i(x_i') \tag{68}
\]
if \( x_i \leq x_i' \) in the sense of positive definite matrices. For example, \( \rho_i(x_i) \) is monotonic in this sense if it is the trace or maximum eigenvalue of the matrix \( x_i \). Finally, define the integer function \( \nu_i(x_i) \) to be the minimum \( n \) such that
\[
\rho_i(f^n(x_i)) \leq \delta_i. \tag{69}
\]
The index rule for the scheduling problem is given by
\[
m_i(x_i) = \begin{cases} 
V_i \frac{\nu_i(x_i)}{1 - \kappa_i^{-1}(x_i)} & \text{if } \nu_i(x_i) > 0 \\
0 & \text{otherwise}
\end{cases}
\]
where \( \kappa_i \) is defined as above.

Proof. One can prove this extension by using the same approach as for the simple scalar case—e.g., by solving the dynamic programming equation by successive approximation and computing the index rule. A more elegant approach can be obtained by using the insightful interpretation of the index function \( m_i(x_i) \) given by (71) as the maximization over stopping times \( \tau \). That is,
\[
m_i(x_i) = \max_{\tau > 0} \frac{\sum_{t=0}^{\tau-1} \alpha^t R_i(x_i(t)) | x_i(0) = x_i}}{\sum_{t=0}^{\tau-1} \alpha^t | x_i(0) = x_i}},
\]
where for the SRM problem \( \tilde{R}_i \) is the transformed reward function.

5 Approximations Based on Index Rule and Examples

5.1 Lookahead Approximation

The index rule is optimal only if the state of unsensed targets remains unchanged, which is not a realistic assumption for tracking problems. However, as noted in [4], it is a reasonable approximation to make when the state changes slowly. In our problem formulation, we require the track error covariance to grow slowly over the time scale \( \alpha^{-1} \) (the discount rate defines a “soft” time horizon for the problem). If the error covariance grows more rapidly over this time horizon, we can use the index rule as a basis for a better approximation of the stochastic dynamic programming problem. For example, we have used the index rule’s value function to compute \( n \)-step lookahead approximations to the dynamic programming equations [1].

The state of the dynamic program is the vector \( x(t) \) of target tracking error covariances (\( x_i(t) \) is the covariance for target \( i \)). The dynamic programming equation has the form
\[
J(x) = \max_u \left\{ R(x) + \int J(F'(x, u, w)) p(w) dw \right\}
\]
(72)
where \( R(x) \) is the immediate reward

\[
R(x) = \sum_{i=1}^{n} R_i(x_i),
\]

and \( F(x, u, w) \) is the dynamic equation

\[
x(t + 1) = F(x(t), u(t), w(t))
\]

with noise \( w(t) \). If \( J_0(x) \) denotes an initial approximation for the value function, then

\[
J_n(x) = \max_{u} \left\{ R(x) + \int J_{n-1}(F(x, u, w)) p(w) dw \right\}
\]

defines the \( n \)-step lookahead approximation for \( n \geq 1 \).

We will use the index rule to define \( J_0 \) and compute the one- and two-step lookahead approximations numerically from it. \( J_0 \) is computed by assuming that the undetected target covariance \( x_i \) don’t change. Let \( m_i(x_i) \) denote the index function of target \( i \), and define \( \kappa_i \) and \( \nu_i(x_i) \) as in the previous section. Moreover, let \( \mu_i(x_i) = m_i(x_i) \) if \( \nu_i(x_i) > 0 \) and \( \mu_i(x_i) = \infty \) if \( \nu_i(x_i) = 0 \). If we re-order the target indices \( i \) so that \( i > i' \) implies that \( \mu_i(x_i) \geq \mu_{i'}(x_{i'}) \), we can show that

\[
J_0(x) = (1 - \alpha)^{-1} \sum_{i=1}^{n} V_i \prod_{j=1}^{i} \kappa^i_j(x_i).
\]

Using this expression for \( J_0 \) we can compute \( J_n \) numerically from (75) for \( n = 1, 2 \). The following examples illustrates the performance for the lookahead approximations and compares them to the performance of the pure index rule (\( n = 0 \)) approximation.

### 5.2 Example

The one- and two-step lookahead and index rule controls are tested on a simple scenario with 10 targets, one of which has a high value and others which have low values. Each target moves according to a simple one-dimensional Brownian motion model. The simulation parameters are given in Table 1. Performance is measured for each target in two ways. The first is the time-averaged relative mean square error

\[
\frac{1}{(T - T_s + 1) \delta} \sum_{t=T_s}^{T} (x_i(t) - \hat{x}_i(t))^2
\]

where \( T \) is the length of the run, \( T_s \), is the time from which the system is assumed to be in steady-state, \( \delta \) is the error goal, \( x_i \) is the true target position, and \( \hat{x}_i \) is the tracker position estimate for a simple Kalman filter. For low-value targets, the MSE is averaged over the targets. The second performance measure is the time averaged frequency that the error goal is attained,

\[
\frac{1}{(T - T_s + 1) \delta} \sum_{t=T_s}^{T} 1(|x_i(t) - \hat{x}_i(t)| < \delta)
\]

For low-value targets, the frequency of goal attainment is averaged over the targets. These performance measures are computed for different values of the moving process noise. The results are plotted in Figures 1-4. Note that the one and two-step lookahead controls keep the high-value target errors down better than the index rule for moderate to low values of the process noise. At high values, the two-step lookahead does slightly worse. This is because the two-step lookahead control is devoting some time to low value targets, dramatically decreasing their errors.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Monte Carlo Runs</td>
<td>100</td>
</tr>
<tr>
<td>Time of Run (( T ))</td>
<td>200</td>
</tr>
<tr>
<td>Steady-state Start Time (( T_s ))</td>
<td>100</td>
</tr>
<tr>
<td>Discount Factor (( \alpha ))</td>
<td>0.9</td>
</tr>
<tr>
<td>High Target Value (( V_i ))</td>
<td>100</td>
</tr>
<tr>
<td>Low Target Value (( V_i ), ( i = 2..10 ))</td>
<td>1</td>
</tr>
<tr>
<td>High-Value Target Error Variance Goal (( \delta ))</td>
<td>0.25</td>
</tr>
<tr>
<td>Low-Value Target Error Variance Goal (( \delta ), ( i = 2..10 ))</td>
<td>4</td>
</tr>
<tr>
<td>Measurement Noise Variance (( q ))</td>
<td>1</td>
</tr>
<tr>
<td>Probability of Detection (( \beta ))</td>
<td>0.8</td>
</tr>
<tr>
<td>Initial Position Variance (( x_i(0) ))</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 1: Simulation Parameters.

![Figure 1: High-Value Target MSE](image)

### References

Figure 2: Low-Value Target MSE.

Figure 3: High-Value Target Goal Attainment Frequency.

Figure 4: Low-Value Target Goal Attainment Frequency.


