[7] U. Shaked, "Transfer function approach to the fixed-point continuous smoothing problem," *IEEE Trans. Automat. Contr.*, vol. AC-23, pp. 945-948, 1978.
[8] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*. New York: Wiley, 1972.
[9] J. S. Meditch, "On optimal fixed point linear smoothing," *Int. J. Contr.*, vol. 6, pp. 189-199, 1967.
[10] T. Kailath, "An innovations approach to least-squares estimation Part II: linear smoothing in additive white noise," *IEEE Trans. Automat. Contr.*, vol. AC-13, pp. 655-660, 1968.
[11] A. E. Bryson and Y. Ho, *Applied Optimal Control*. New York: Wiley, 1975.
[12] L. E. Zachrisson, "On optimal smoothing of continuous-time Kalman processes," *Inform. Sci.*, vol. 1, pp. 143-172, 1969.
[13] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall, 1979.
[14] A. P. Sage and J. L. Melsa, *Estimation Theory with Applications to Communications and Control*. New York: McGraw-Hill, 1971.
[15] J. F. Barrett and T. J. Moir, "A unified approach to multivariable discrete-time filtering based on the Wiener theory," *Kybernetika*, vol. 23, no. 3, 1987.
[16] M. J. Grimble and M. A. Johnson, *Optimal Control and Stochastic Estimation*. New York: Wiley, 1988.
[17] T. J. Moir, "Optimal deconvolution smoother," *Proc. IEE*, vol. 133, pt. D, no. 1, pp. 13-18, Jan. 1986.

Stochastic Task Selection and Renewable Resource Allocation

PETER B. LUH, XI-YI MIAO, SHI-CHUNG CHANG, AND DAVID A. CASTANON

Abstract—**In this note, we study a class of renewable resource allocation problems for the processing of dynamically arriving tasks with deterministic deadlines. This class of problems has many applications. However, they conform to neither the standard resource allocation model nor the standard optimal control model. A new problem formulation has to be developed and analyzed to provide a satisfactory answer to these problems. The model presented here explicitly considers time available, time required, resource available, resource required, stochastic arrivals of multiple types of tasks, importance of tasks, timeliness of processing, and accuracy of resource allocation. After state augmentation, the problem becomes a Markovian decision problem, and can be solved, at least in principle, by using a dynamic programming (SDP) method. For a problem with infinite planning horizon, the optimal strategy is shown to be stationary under mild conditions. An SDP algorithm based on a successive approximation technique is developed to obtain the optimal stationary strategy. Effects of key system parameters on optimal decisions are investigated and analyzed through numerical examples. As the computational complexity of the algorithm is of exponential increase, practical application of the algorithm is limited to problems of small size. Two heuristic rules are therefore investigated and compared to the optimal policy. The result of this study serves as a starting point for further characterization of the optimal policy, for understanding and designing effective heuristic rules, and for developing (in conjunction with experimental studies) normative-descriptive models of human task selection and resource allocation.**

I. Introduction

In this note, we study a class of resource allocation problems for the processing of dynamically arriving tasks with deterministic deadlines. This class of problems has many applications. For example, consider the tactical operation of a Naval Battle Group with finite renewable resources (aircraft, warships, submarines, etc.). The battle group's mission is to process hostile threats, which can be of various types (air, surface, and underwater threats), have different strengths, and arrive stochastically. The battle group has a limited amount of time to process a threat before it causes substantial damage. Since available resources are finite, and a resource will be tied up for a certain amount of time once it is allocated to process a task, an effective task selection/resource allocation policy is highly desirable to maximize the survivability of the battle group.

The problems of task selection and resource allocation have long been studied from different perspectives and with different emphases. Queueing analysis and control theory have been used to study task priority assignment problems when values of servicing tasks do not depend on the service completion time, e.g., [3], [7], [14], [13], [11], [9], and [15]. Although queueing formulations provide useful insights, they do not consider the effects of soft or hard deadlines and the selection of resource levels to process tasks. On the other hand, many static models and deterministic dynamic models have been suggested and solved for renewable resource allocation problems in the operations research literature (e.g., [1], [4], [12], and [6]). However, since the stochastic aspect has not been included in these models, the resulting solution methodologies cannot be easily extended to problems which we are contemplating.

Motivated by decision making problems in scenarios such as Naval Battle Group/Battle force operations [8], we develop in this note a mathematical formulation for the deadline-driven, optimal task selection and resource allocation problem with multiple types of tasks and one class of renewable resource. There are two major purposes of our study. First, by setting up a simple but generic model that captures key ingredients of the problem, solution techniques and computational algorithms can be developed. Difficulties in solving a problem of realistic size can be exposed. Second, the result serves as a foundation in understanding the kinds of heuristic strategies employed in human task selection and resource allocation [10]. Although the present model is primitive, the results obtained do bring a fair amount of insight to key factors of the human decision making process.

The paper is organized as follows. In Section II, a new formulation for the task selection and renewable resource allocation problem is presented. The model explicitly considers time available, time required, resource available, resource required, importance of a task, stochastic arrivals of multiple types of tasks, timeliness of processing, and accuracy of resource allocation. As the tying up of resource in task processing implies delay in resource flow, state augmentation is employed to convert the delay problem into a Markovian decision problem. The optimal policy can thus be obtained, at least in principle, by applying the stochastic dynamic programming (SDP) method. However, since the system dynamics involves the evolution of sets, the implementation of the dynamic programming equation is by no means straightforward. For a problem with infinite planning horizon, the optimal strategy is shown to be stationary under mild conditions. In Section III, the development of an SDP algorithm for the infinite horizon case is presented, and a successive approximation technique is used to obtain numerical results. The implementation of the algorithm employs a special coding scheme to handle set variables, and utilizes a dominance property for computational efficiency. In Section IV, effects of key system parameters on optimal decisions are investigated through numerical examples. As the computational complexity of the SDP algorithm is of exponential increase, practical applications of the algorithm is limited to problems of small size. Two heuristic rules are therefore studied in Section V. Their results are compared to optimal results of Section IV.

II. Problem Formulation

A. Task States and Dynamics

Consider a discrete-time task selection and resource allocation problem with I types of tasks and one class of renewable resource. Tasks arrive stochastically and wait to be processed by a decision maker (DM). It is assumed that once a task appears, the DM knows perfectly the type of the

Manuscript received December 11, 1987. This paper is based on a prior submission of September 18, 1986. This work was supported in part by Alphatech, Inc. under Subcontract SC-000192-01 as part of the Office of Naval Research Program on Distributed Tactical Decision Making, and by the National Science Foundation under Grant ECS-8513163.

P. B. Luh, X.-Y. Miao, and S.-C. Chang are with the Department of Electrical and Systems Engineering, University of Connecticut, Storrs, CT 06268.

D. A. Castanon is with Alphatech, Burlington, MA 01803.

IEEE Log Number 8824569.

task, say type $i (1 \le i \le I)$, together with the following attributes:

1) the time period during which the task will stay in the system, i.e., initial time available, $T_{ao}(i)$;

2) the amount of resource required to process the task, $\bar{r}_i$ (strength of the task);

3) the time required to process the task, $T_r(i)$;

4) the reward for processing the task, $g(i)$ (importance of the task).

New tasks appear stochastically. At any time, there may be more than one new arrival. For simplicity, we assume that all new tasks at a particular instant are of different types. At the appearance of a new task, its time available is $T_{ao}(i)$, the initial time available. As time elapses, its time available decreases. A type i task with j units of time available is denoted as (i, j), where $1 \le i \le I$ and $1 \le j \le T_{ao}(i)$. The set of new tasks at time k is denoted by $a(k) = \{(i, T_{ao}(i))\}$, and the set of tasks selected to be processed is denoted by $u(k)$. We define the active task set $S(k)$ as the set of all existing [with $1 \le j < T_{ao}(i)$] but yet unprocessed tasks at time k. The evolution of the active task set follows the task flow equation.

$$S(k+1) \equiv \{(i, j) | (i, j+1) \in S(k) \cup a(k+1), j \ge 1, (i, j+1) \notin u(k)\}. \tag{2.1}$$

At time k, the DM is supposed to select from $S(k)$ a subset of tasks $u(k)$ to process. Note that the time available j of an unprocessed task decreases as time evolves until it reaches zero and the task leaves the system. Also, once a task is processed, it is removed from the active task set.

B. Resource States and Dynamics

The DM owns in total R units of a single class of renewable, discrete resource. When the DM allocates $r_{(i,j)}(k)$ units of resource to process task (i, j) at time k, this amount of resource will be tied up for $T_r(i)$ units of time before it can be utilized again. The flow of resource is therefore like a pipeline, and involves dynamics with time delay. State augmentation is employed so that resource flow can be described by standard dynamic equations without delay. At time k, let $x_m(k)$ denote the amount of tied up resource to be released at time $k + m$. For simplicity but without loss of generality, the types of tasks are arranged in the ascending order of $T_r(i)$, i.e.,

$$1 \le T_r(1) \le T_r(2) \le \cdots T_r(I) \equiv M \tag{2.2}$$

where M is largest time required in task processing. Then the resource flow equations are

$$\begin{aligned} x_{M-1}(k+1) &= \sum_{(i,j) \in u(k), T_r(i)=M} r_{(i,j)}(k), \\ x_{M-2}(k+1) &= x_{M-1}(k+1) + \sum_{(i,j) \in u(k), T_r(i)=M-1} r_{(i,j)}(k), \\ &\vdots \\ x_1(k+1) &= x_2(k) + \sum_{(i,j) \in u(k), T_r(i)=2} r_{(i,j)}(k). \end{aligned} \tag{2.3}$$

The vector $X(k) \equiv (x_1(k), x_2(k), \cdots, x_{M-1}(k))^T$ is called the resource usage vector, where superscript T denotes transpose. Since the total amount of resource owned by the DM is R, the following resource constraint holds for every k:

$$0 \le \sum_{m=1}^{M-1} x_m(k) + \sum_{(i,j) \in u(k)} r_{(i,j)}(k) \le R, \tag{2.4}$$

i.e., the total resource usage (tied up units plus newly allocated units) cannot be greater than R. The resource usage vector $X(k)$ and the active task set $S(k)$ together then form the state of the system.

C. Task Arrival Statistics and Reward Structure

Task arrivals are likely to depend on system states. For example, in Naval Battle Group tactical operations, new threats often appear in specific patterns based on current threats and enemy's estimation of our resource utilization situation. We therefore model the probability of task arrivals as $p(n(k + 1) | S(k), X(k))$, i.e., a Markov process.

The DM maximizes his rewards in task processing. Two aspects are emphasized in the reward structure: timeliness of processing and accuracy of resource allocation. Timely processing means that when task (i, j) is selected, its time available is no less than its time required, i.e., $j \ge T_r(i)$. Accurate processing means that the amount of resource allocated to task (i, j) is equal to the amount of resource required $\bar{r}_i$ (these definitions can be modified without much difficulty). Untimely and/or inaccurate processing result in diminishing rewards. Let $g(i)$ be the reward for timely and accurate processing of a type I task. The reward for processing task (i, j) is then described by

$$g(i) g_1(j - T_r(i)) g_2 \left[\frac{r_{(i,j)}}{\bar{r}_i} \right] \tag{2.5}$$

where $0 \le g_1(\cdot) \le 1$ and $0 \le g_2(\cdot) \le 1$ are discounting functions for untimely and inaccurate processing, respectively. They can be defined at the users' disposal, and in general should satisfy the following conditions:

$$g_1(j - T_r(i)) \begin{cases} = 1 & \text{if } j - T_r(i) \ge 0, \\ < 1 & \text{if } j - T_r(i) < 0, \end{cases} \tag{2.6}$$

$$g_2 \left[\frac{r_{(i,j)}}{\bar{r}_i} \right] \begin{cases} = 1 & \text{if } \frac{r_{(i,j)}}{\bar{r}_i} \ge 1, \\ < 1 & \text{if } \frac{r_{(i,j)}}{\bar{r}_i} < 1. \end{cases} \tag{2.7}$$

Since we assume that the DM has perfect information about existing tasks, and there is no reward for over-committing resource as described by (2.7), we therefore have $r_{(i,j)} \le \bar{r}_i$. Furthermore, since there is no uncertainty in task processing once resource is allocated, rewards can be counted at the time when resource is allocated. This thus justifies the deletion of $u(k)$ from the active task set as described by (2.1).

D. The Markovian Decision Problem and Problems with Infinite Planning Horizon

The formulation of a K-stage problem can now be summarized as

$$\max_{U,r} E \left[\sum_{k=0}^{K-1} \sum_{(i,j) \in u(k)} g(i) g_1(j - T_r(i)) g_2 \left[\frac{r_{(i,j)}}{r(i)} \right] \right] \tag{2.8}$$

subject to 1) the task flow equation (2.1)
2) the resource flow equations (2.3)
3) the resource constraint (2.4)

where $U \equiv (u(0), u(1), \cdots, u(K-1))$ and $r \equiv (r(0), r(1), \cdots, r(K-1))$. It is not difficult to see that this is a Markovian decision problem with stagewise additive objective function. The problem can therefore be solved, at least in principle, by using the stochastic dynamic programming (SDP) method.

Now consider an infinite horizon problem with the following discounted objective function:

$$\lim_{K \to \infty} E \left[\sum_{k=0}^{K-1} \alpha^k \sum_{(i,j) \in u(k)} g(i) g_1(j - T_r(i)) g_2 \left[\frac{r_{(i,j)}}{\bar{r}_i} \right] \right] \tag{2.9}$$

subject to the same set of constraints and with $0 < a < 1$. If the task arrival statistics $P(\cdot | X(k), S(k))$ is a countable measure and $g(i)$ is bounded above, then an optimal stationary policy exists following [2, ch. 6]. We shall next describe an algorithm in obtaining the optimal stationary policy for such a problem.

III. An SDP Algorithm

A. Stationary Dynamic Programming Equation

Consider an infinite horizon, discounted stochastic dynamic programming problem with the objective function described by (2.9). Let us define

$$J^*(S, X) = \max_{(u,r)} E\left[\sum_{k=0}^{\infty} \alpha^k \sum_{(i,j)\in u(k)} g(i) g_1(j - T_r(i)) g_2\left[\frac{r_{(i,j)}}{\bar{r}_i}\right]\right] \tag{3.1}$$

as the optimal value function for the given initial state (S, X). It is apparent that the problem involves both vector and set variables, with the latter including $\bar{S}(k)$ and $u(k)$. A special coding scheme is therefore developed to handle set variables. The idea is explained by using a simple example. Suppose that there are only two types of tasks, i.e., $I = 2$ with $T_{ao}(1) = 4$, $T_{ao}(2) = 3$. Clearly, the largest active task set is $\bar{S} \equiv \{(1, 1), (1, 2), (1, 3), (2, 1), (2, 2)\}$, and any active task set $S(k)$ is a subset of $\bar{S}$. The total number of possible active task sets is 2^5, and any active task set can be represented by a binary number

$$B(k) = b_5 b_4 b_3 b_2 b_1 \tag{3.2}$$

where b_n is a $0 - 1$ indication variable corresponding to the nth element of $\bar{S}$. In other words, $b_n = 1$ if the nth task of $\bar{S}(k)$ is in $S(k)$. We thus have a one-to-one mapping between all possible active task sets and binary numbers ranging from 0 to 11111. For instance, b = 01101 corresponds to $S(k) = \{(1, 2), (1, 3), (2, 2)\}$, and 00000 corresponds to $S(k) = \phi$, etc. Both set variables $S(k)$ and $u(k)$ are handled by using this scheme. For a finite-dimension problem, i.e., both X and S take finite values, the problem can be solved by using the successive approximation technique described in [2, p. 246]. The algorithm is implemented for a simplified version of the problem, in which the reward structure is of the following form:

$$g_1(j - T_r(i)) = \begin{cases} 1 & \text{if } j - T_r(i) \geq 0, \\ 0 & \text{if } j - T_r(i) < 0 \end{cases} \tag{3.3}$$

and

$$g_2\left[\frac{r_{(i,j)}}{\bar{r}_i}\right] = \begin{cases} 1 & \text{if } \frac{r_{(i,j)}}{\bar{r}_i} = 1, \\ 0 & \text{if } \frac{r_{(i,j)}}{\bar{r}_i} < 1, \end{cases} \tag{3.4}$$

i.e., no partial reward for an untimely and/or inaccurate processing. The statistics of new task arrivals is assumed to be independent, identically distributed for each type of task. These simplifying assumptions can be removed without much difficulty. From the reward structure (3.3), a task with time available less than time required deserves no attention, and the effective selection time for task (i, j) is $j - T_r(i) + 1$. A task worth considering, say (i, j), can thus be written as (i, j'), where $j' \equiv j - T_r(i) + 1$ is the effective selection time. The set of all such tasks is called the effective active task set $S_e(k)$. That is

$$S_e(k) \equiv \{(i, j') | j' \equiv j - T_r(i) + 1 > 0, (i, j) \in S(k)\}. \tag{3.5}$$

Equation (3.4) implies that if task (i, j') is selected for processing, then $r_{(i,j')} = \bar{r}_i$.

B. A Dominance Property

As of most SDP algorithms, the computational complexity of this problem is of exponential increase. To improve the computational efficiency, a dominance property of the optimal strategy under the special structure of (3.3) is used to reduce the admissible decision space: among the same type of active tasks, the task with the shortest effective selection time has the highest priority. The reason is that if we process a task of the same type but with a longer time, the reward and the resource tied up situation will be identical to the case if we process the one with the shortest time. However, we are more likely to lose the task with the shortest time. Starting with the most urgent task, we preserve the opportunity of processing other tasks in the future. Using this dominance property and also considering the resource constraint (2.4), the number of admissible decisions can be reduced, although the computational complexity of the problem remains to be of exponential increase. Further refinement in modeling and solution methodology is needed to make the algorithm more efficient.

IV. Numerical Results

Beyond the problem formulation and solution methodology, we are particularly interested in the effects of following system parameters on optimal decisions:

1) discounting factor α,
2) reward $g(i)$,
3) effective selection window $T_{ao}(i) - T_r(i)$.

The study is done through numerical examples under the simplified formulation (3.3) and (3.4) with two types of tasks and specific sets of system parameters. Note that in all the tables a task is represented by its type and effective selection time.

Example 1: With the nominal set of parameters.

In Table I, the following parameters are used. In addition, task arrival statistics are independent, identically distributed with $p(1) = 0.35$, $p(2) = 0.35$, and the total amount of available resource R is 3. Samples of optimal stationary decisions for $\alpha = 0.9$ are given in Table II.

Case 1 (row 1) in Table II says that with $S_e = \{(1, 1), (1, 3), (2, 1)\}$ and $x = (0, 0, 0)^T$, the best decision is to process tasks (1, 1) and (2, 1) with $r_{(1,1)} = 2$, $r_{(2,1)} = 1$. The remaining rows can be similarly explained. Note that all the cases here are samples of a big decision table, and there is no relationship among individual cases. Decisions for cases 1, 2, and 5 are intuitively clear. Urgent tasks are processed with all available resources. Decisions for cases 3 and 4 are not as easy to comprehend. Optimal policies select all type 2 tasks at the cost of losing the task (1, 1). Suppose in case 3 tasks (1, 1) and (2, 1) are selected instead. Although task (1, 1) will be processed in time, however, tasks (2, 2) and (1, 3) will surely be lost. According to the optimal decision, only task (1, 1) is surely lost. Thus, sometimes it is better to strategically abandon certain tasks. Decision for case 6 is optimal by considering the return of future resource. That is, with one unit of resource reserved, task (1, 2) can be processed at the next stage as one unit of resource is going to be released at that time; and at the following stage, task (2, 3) can be processed by using the resource currently allocated to task (2, 2). Consequently, chances for processing (1, 2) and (2, 3) are preserved.

Example 2: Varying the discounting factor α.

The discounting factor α reflects how far the DM looks into the future. Here we examine the effects of α by changing it from 0.9 to 0.1 while keeping all other parameters the same. Part of the results which are different from those of Table II are shown in Table III.

With $\alpha = 0.1$ as in Table III, the DM puts more emphasis on immediate rewards. In case 2, he selects tasks (1, 3) and (2, 1) to get 30 units of immediate rewards, which is higher than the choice of (2, 1) and (2, 2) as in Table II. This would not be good if $\alpha = 0.9$. Similar things happen in case 6.

Example 3: Varying the reward $g(i)$.

Consider the case where $g(2)$ is changed from 10 to 40, with all other parameters remaining as in Example 1. It is intuitively clear that if one type of task becomes more valuable, that type of task will be selected. Our results show that the optimal policy selects no type 1 task under any circumstance. The reason is that if a type 1 task is selected, the DM will lose the capability of processing the much valuable type 2 tasks for four stages.

Example 4: Varying the effect selection window.

The length of effective selection window for a type i task is $T_{ao}(i) - T_r(i)$. Consider a problem similar to Example 1 except that $T_{ao}(1) = 5$ (the length of effective selection window is reduced from 3 to 1). Samples of results are presented in Table IV. We see that more type 1 tasks are

TABLE I
PARAMETERS FOR EXAMPLE 1

	type 1	type 2
T_∞	7	5
T_r	4	2
$\bar{r}$	2	1
g	20	10

TABLE II
SAMPLES OF OPTIMAL POLICY WITH $\alpha = 0.9$

	Effective Active Task Set S_e	Tied Up Resources x_3	x_2	x_1	Decisions u
1	(1,1) (1,3) (2,1)	0	0	0	(1,1) (2,1)
2	(1,3) (2,1) (2,2)	0	0	0	(2,1) (2,2)
3	(1,1) (1,3) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)
4	(1,1) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)
5	(1,1) (2,2) (2,3)	0	0	1	(1,1)
6	(1,2) (2,1) (2,3)	0	0	1	(2,1)

TABLE III
SAMPLES OF OPTIMAL POLICY WITH $\alpha = 0.1$

	Effective Active Task Set S_e	Tied up Resources x_3	x_2	x_1	Decisions u
2	(1,3) (2,1) (2,2)	0	0	0	(1,3) (2,1)
6	(1,2) (2,1) (2,3)	0	0	1	(1,2)

TABLE IV
SAMPLES OF OPTIMAL POLICY WITH SHORT EFFECTIVE SELECTION WINDOW

	Effective Active Task Set S_e	Tied Up Resources x_3	x_2	x_1	Decisions u
4	(1,1) (2,1) (2,2) (2,3)	0	0	0	(1,1) (2,1)
5	(1,1) (2,2) (2,3)	0	0	1	(1,1)

selected as compared to Table II, as the only chance to process a type 1 task in $S_e(k)$ is to process it as it appears.

V. TESTING OF TWO HEURISTIC RULES

In previous sections, an SDP algorithm was used to obtain optimal decisions. From sample output, we see that the optimal policy is quite complicated, and depends on system state and parameters in a very perplexing way. Furthermore, because of computational complexity, the application of this algorithm is limited to problems of small size. As human decision making is generally believed to be heuristic, and there are many heuristic rules used in daily applications, we shall therefore apply two heuristic rules to Example 1.

A. The Myopic Policy

The first heuristic rule considered is the myopic policy, in which decisions are made to achieve the maximum present rewards without considering the future. It can be regarded as a solution of the following problem:

$$\max_{(u,r)} \sum_{(i,j)\in u(k)} g(i)g_1(j-T_r(i))g_2\left[\frac{r_{(i,j)}}{r(i)}\right] \tag{5.1}$$

subject to (2.1), (2.3), and (2.4).

In solving (5.1), the dynamic and stochastic aspects of the original problem are gone. Consequently, there is no computational complexity issue, and on-line implementation is also straightforward. Myopic decisions and expected rewards (assuming $\alpha = 0.9$ in calculating expected rewards) are presented in Table V. For comparison purposes, expected reward J^* obtained by using the SDP algorithm for $\alpha = 0.9$ are also presented in the table.

TABLE V
SAMPLES OF MYOPIC POLICY

	Effective Active Task Set S_e	Tied up Resources x_3	x_2	x_1	Decision u	Reward J	J^*
1	(1,1) (1,3) (2,1)	0	0	0	(1,1) (2,1)	90.5403	90.5775
2	(1,3) (2,1) (2,2)	0	0	0	(1,3) (2,1)	90.5403	91.4431
3	(1,1) (1,3) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)	100.5757	100.6092
4	(1,1) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)	96.6499	96.6855
5	(1,1) (2,2) (2,3)	0	0	1	(2,2) (2,3)	87.3330	88.5613
6	(1,2) (2,1) (2,3)	0	0	1	(2,1) (2,3)	87.3330	90.0229

TABLE VI
SAMPLES OF μc RULE

	Effective Active Task Set S_e	Tied up Resources x_3	x_2	x_1	Decision u	Reward J
1	(1,1) (1,3) (2,1)	0	0	0	(1,1) (2,1)	89.2510
2	(1,3) (2,1) (2,2)	0	0	0	(2,1) (2,2)	89.6104
3	(1,1) (1,3) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)	98.8982
4	(1,1) (2,1) (2,2) (2,3)	0	0	0	(2,1) (2,2) (2,3)	95.2213
5	(1,1) (2,2) (2,3)	0	0	1	(2,2) (2,3)	85.9389
6	(1,2) (2,1) (2,3)	0	0	1	(2,1) (2,3)	85.9389

Note that for each case, the difference in reward for the two policies is not big. The myopic policy may lose more existing tasks in the near future than the optimal one for being shortsighted. However, with processing those tasks lost, the resource will be left available. As probabilities of new task arrivals are quite high ($p(1) = p(2) = 0.35$), the available resources can then be used to process new tasks in a myopic fashion. Consequently, the difference in reward between the two policies would not be too large. The discounting factor tends to further reduce the difference.

B. The μc Rule

The second rule is the so-called μc rule in queueing theory [5]. The priority index of task $\{(i, j')\}$ is computed as $g(i)/T_r(i)r(i)$, which is the reward per unit processing time and per unit resource. For tasks of the same type, the one with the shortest effective selection time has the highest priority. Sample results are presented in Table VI.

This rule considers time required and resource required in addition to reward. However, since the priority is determined in an ad hoc way, and the resource tied up situation and task arrival probabilities are not considered, the μc rule is not always better than the myopic scheme. In fact, in the example it is inferior to the myopic rule for most of the cases shown.

VI. SUMMARY

A major difficulty in studying stochastic task selection and renewable resource allocation is to come up with the right problem formulation which captures key ingredients and also lends itself to systematic analysis. The model presented here explicitly considers time available, time required, resource available, resource required, stochastic arrivals of multiple types of tasks, importance of a task, timeliness of processing, and accuracy of resource allocation. After state augmentation, the problem becomes a Markovian decision problem, and can be solved by using the stochastic dynamic programming method. For a problem with infinite horizon, the optimal policy is stationary under mild conditions. An SDP algorithm is developed to find the optimal stationary policy based on a successive approximation technique. In implementing the algorithm, a specific coding scheme and a dominance property are used. Numerical results are obtained and effects of key system parameters are examined and analyzed. Two heuristic rules are tested and compared to the optimal policy. The results of this study serve as a starting point in further characterization of the optimal policy, in understanding and designing effective heuristic rules, and in developing (in conjunction with experimental studies) normative-descriptive models of human task selection and resource allocation.

References

[1] K. R. Baker, *Introduction to Sequencing and Scheduling*. New York: Wiley, 1974.
[2] D. P. Bertsekas, *Dynamic Programming and Stochastic Control*. New York: Academic, 1976.
[3] J. R. Carbonell, "A queueing model of many instrument visual sampling," *IEEE Trans. Hum. Factors Electron.*, vol. HFE-7, pp. 154-164, Dec. 1966.
[4] E. G. Coffman, *Computer and Job Shop Scheduling*. New York: Wiley, 1976.
[5] D. R. Cox and W. L. Smith, *Queues*. London: Methuen, 1961.
[6] M. D. Diamond, and O. M. Carducci, "Decision processes for large scale resource allocation problems," in *Proc. 8th MIT/ONR Workshop on c^3 Systems*, Cambridge, MA, June 1985, pp. 153-160.
[7] J. M. Harrison, "Dynamic scheduling of a multiclass queue: Discount optimality," *Operat. Res.*, vol. 23, pp. 70-82, Mar./Apr. 1975.
[8] D. K. Kleinman, D. Serfaty, and P. B. Luh, "A research paradigm for multi-human decision making," in *Proc. 1984 Amer. Contr. Conf.*, San Diego, CA, June 1984, pp. 6-11.
[9] K. R. Pattipati and D. L. Kleinman, "Priority assignment using dynamic programming for a class of queueing systems," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 1095-1106, Oct. 1981.
[10] X. Y. Miao, P. B. Luh, and D. A. Castanon, "A normative/descriptive model for distributed team resource allocation," in *Proc. 1987 Command and Control Research Symp.*, Washington, DC, June 1987.
[11] W. B. Rouse, *System Engineering Models of Human-Machine Interaction* (Series Vol. 6). Amsterdam, The Netherlands: North-Holland, 1980.
[12] M. K. Schaefer, "Optimal allocation of recoverable items," *Decision Sci.*, pp. 147-155, 1982.
[13] R. S. Walden, "A queueing model of pilot decision-making in a multitask flight management situation," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp. 867-875, Dec. 1978.
[14] J. A. White, J. W. Schmidt, and G. K. Bennett, *Analysis of Queueing Systems*. New York: Academic, 1977.
[15] Z. J. Wu, P. B. Luh, S. C. Chang, and D. A. Castanon, "Optimal task allocation for team of two decision makers with three classes of impatient tasks," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 42-49, Jan. 1988.

An Improved Adaptive Control for Robust Adaptation

I. H. MUFTI

Abstract—**Recently, Ioannou and Kokotovic [1] and Narendra and Annaswamy [2] have proposed some modifications in the adaptive law that ensure adaptive system robustness. In this note, it is shown that the performance of these adaptive schemes is improved if a proportional adaptation term is added to the control law. The scalar case is only discussed. Simulation results are presented to complement the theoretical developments.**

Introduction

In recent years, several modifications have been proposed for the adaptive control of an unknown plant in the presence of disturbances. One such modification due to Ioannou and Kokotovic [1], known as σ-modification, introduces an additional term—$\sigma\theta$, $\sigma > 0$, in the adaptive law for adjusting the parameter vector θ. Motivated by their work, Narendra and Annaswamy [2] proposed another adaptive law in which the term σ is replaced by a term $-\gamma\,|e_1|$, $\gamma > 0$, where e_1 is the output error. Although these modifications guarantee stability, they do not, however, ensure that the transient response is satisfactory.

To achieve this, an adaptive control law consisting of an integral and proportional adaptation is suggested in this note. A preliminary investigation carried on a first-order system shows that the additional term fex in the control does improve the performance of the above-mentioned adaptive schemes. Numerical experiments are presented to complement the theoretical results.

Manuscript received May 27, 1987; revised November 13, 1987 and May 8, 1988.
The author is with the Systems Laboratory, Division of Mechanical Engineering, National Research Council of Canada, Ottawa, Ont., Canada K1A OR6.
IEEE Log Number 8824570.

Adaptive Control of a First-Order System

A first-order system is described by the differential equation

$$\dot{x} = ax + u \tag{1}$$

where a is an unknown parameter and u is the control input. The system (1) may represent a reduced-order model of a second-order plant [1]

$$\dot{x} = ax + 2\eta + u \tag{2}$$

$$\mu\dot{\eta} = -\eta - \mu\dot{u} \tag{3}$$

where μ is the perturbation parameter and η is the parasitic state. Equation (2) reduces to (1) when $\mu = 0$. Let the reference model be given by

$$\dot{x}_m = -a_m x_m + r \tag{4}$$

where $a_m > 0$ and $r(t)$ is a bounded reference input. The purpose of the adaptive control is to determine $u = u(t)$ such that the error $e = x - x_m$ asymptotically approaches zero. Using (1) and (4), the equation for the error can be expressed as

$$\dot{e} = -a_m e + K_o x + u - r \tag{5}$$

where $K_o = (a + a_m)$. Let the adaptive controller be given by

$$u = -kx + r \tag{6}$$

where $k = K + fex$, $f \geq 0$, and K is given by

$$\dot{K} = gex \qquad g > 0. \tag{7}$$

The additional term fex in the control law is called proportional adaptation [3]. Substituting (6) into (5), the error equation can be written as

$$\dot{e} = -a_m e - (K - K_o)x - fex^2. \tag{8}$$

In order that the adaptive law (7) be robust in the presence of persistent excitations, bounded perturbations and unmodeled dynamics, (7) is modified as

$$\dot{K} = -\sigma K + gex \qquad \sigma > 0 \tag{9}$$

in [1] and as

$$\dot{K} = -\gamma|e|K + gex \qquad \gamma > 0 \tag{10}$$

in [2]. The modification (10) is called e_1-modification.

For the system of equations (8) and (9) [or (10)], it is easy to show that all solutions are bounded. In fact, let $V = 1/2\,(e^2 + (K - K_o)^2/g)$ be the Lyapunov function. Then, its total time derivative along the trajectory of (8) and (9) is given by

$$\begin{aligned}\dot{V} &= e(-a_m e - (K - K_o)x - fex^2) + (K - K_o)(-\sigma K + gex)/g \\ &= -[a_m e^2 + \sigma(K - K_o/2)^2/g - \sigma K_o^2/4g] - fe^2x^2.\end{aligned}$$

Therefore, $\dot{V} < 0$ outside the ellipse, given by the equation

$$a_m e^2 + \sigma(K - K_o/2)^2/g - \sigma K_o^2/4g = 0.$$

Hence, according to a result of LaSalle [4, p. 113], it follows that all solutions are bounded. It should be mentioned that this result holds even with $f = 0$. However, for $f > 0$, $\dot{V}$ becomes more negative. The boundedness of the solutions of the system (8) and (10) follows similarly.

In the remainder of the note, we demonstrate the effectiveness of the additional term fex through linear analysis and simulation. We consider the regulation case first.

0018-9286/89/0300-0339$01.00