Room 14-0551
77 Massachusetts Avenue
Cambridge, MA 02139
Ph: 617.253.5668 Fax: 617.253.1690
Email: docs@mit.edu
http://libraries.mit.edu/docs

DISTRIBUTED DYNAMIC PROGRAMMING*

Dimitri P. Bertsekas
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Abstract

We consider distributed algorithms for solving dynamic programming problems whereby several processors participate simultaneously in the computation while maintaining coordination by information exchange via communication links. A model of asynchronous distributed computation is developed which requires very weak assumptions on the ordering of computations, the timing of information exchange, the amount of local information needed at each computation node, and the initial conditions for the algorithm. The class of problems considered is very broad and includes shortest path problems, and finite and infinite horizon stochastic optimal control problems. When specialized to a shortest path problem the algorithm reduces to the algorithm originally implemented for routing of messages in the ARPANET.

1. Introduction

Recent advances in microcomputer technology have intensified interest in distributed computation schemes. Aside from modular expandability, other potential advantages of such schemes are a reduction in computation time for solving a given problem due to parallelism of computation, and elimination of the need to communicate problem data available at geographically dispersed data collection points to a computation center. The first advantage is of crucial importance in real time applications where problem solution time can be an implementation bottleneck. The second advantage manifests itself for example in applications involving communication networks where there is a natural decentralization of problem data acquisition.

The structure of dynamic programming naturally lends itself well to distributed computation since it involves calculations that to a great extent can be carried out in parallel. In fact it is trivial to devise simple schemes taking advantage of this structure whereby the calculation involved in each iteration of the standard form of the algorithm is simply shared by several processors. Such schemes require a certain degree of synchronization in that all processors must complete their assigned portion of the computation before a new iteration can begin. As a result complex protocols for algorithm initiation and processor synchronization may be necessary, and the speed of computation is limited to that of the slowest processor. These drawbacks motivate distributed algorithms whereby computation is performed asynchronously at various nodes and independently of the progress in other nodes. Their potential advantages are simpler implementation, faster convergence to a solution and, possibly, a reduction in information exchange between computation nodes.

This paper considers an asynchronous distributed algorithm for a broad class of dynamic programming problems. This class is described in Section 2. The distributed computation model is described in Section 3. It is shown in Section 4 that the algorithm converges to the correct solution under very weak assumptions. For some classes of problems convergence in finite time is demonstrated. These include shortest path problems for which the distributed algorithm of this paper turns out to be essentially the same as the routing algorithm originally implemented in the ARPANET in 1969 [1]. To our knowledge there is no published proof of convergence of this algorithm.

2. Problem Formulation

We use an abstract framework of dynamic programming, first introduced in [2], [3] which includes as special cases a number of specific problems of practical interest.

Let S and C be two sets referred to as the <u>state space</u> and the <u>control space</u> respectively. Elements of S and C are referred to as <u>states</u> and <u>controls</u> and are denoted by x and u respectively. For each $x\varepsilon S$ we are given a subset $U(x)\subset C$ referred to as the <u>control constraint set at x</u>. Let F be the set of all extended real valued functions $J: S\rightarrow[-\infty,\infty]$ on S. For any two functions $J_1, J_2\varepsilon F$ we use the notation

$$J_1 \leq J_2 \quad \text{if} \quad J_1(x) \leq J_2(x), \quad \forall\, x\varepsilon S, \qquad (1a)$$

$$J_1 = J_2 \quad \text{if} \quad J_1(x) = J_2(x), \quad \forall\, x\varepsilon S. \qquad (1b)$$

Let $H: S \times C \times F \rightarrow [-\infty,\infty]$ be a mapping which is <u>monotone</u> in the sense that for all $x\varepsilon S$ and $u\varepsilon U(x)$ we have

*This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with partial support provided by the National Science Foundation Grant No. NSF/ECS 79-19880.

To be presented at the 20th IEEE Conference on Decision and Control, San Diego, CA, December 1981.

$$H(x,u,J_1) \le H(x,u,J_2), \quad \forall J_1, J_2 \varepsilon F, \text{ with } J_1 \le J_2. \tag{2}$$

Given a subset $\overline{F} \subset F$ the problem is to find a function $J^* \varepsilon \overline{F}$ such that

$$J^*(x) = \inf_{u \varepsilon U(x)} H(x,u,J^*), \quad \forall x \varepsilon S. \tag{3}$$

By considering the mapping $T: F \to F$ defined by

$$T(J)(x) = \inf_{u \varepsilon U(x)} H(x,u,J) \tag{4}$$

the problem is alternately stated as one of finding a fixed point of T within $\overline{F}$, i.e., a function $J^* \varepsilon \overline{F}$ such that

$$J^* = T(J^*). \tag{5}$$

We will assume throughout that T has a unique fixed point within $\overline{F}$.

We provide some examples that illustrate the broad scope of the problem formulation just given.

Example 1 (Shortest Path Problems): Let (N,L) be a directed graph where $N = \{1,2,\ldots,n\}$ denotes the set of nodes and L denotes the set of links. Let $N(i)$ denote the downstream neighbors of node i, i.e., the set of nodes j for which (i,j) is a link. Assume that each link (i,j) is assigned a positive scalar a_{ij} referred to as its length. Assume also that there is a directed path to node 1 from every other node. Then is is known ([4], p.67) that the shortest path distances d_i^* to node 1 from all other nodes i solve uniquely the equations

$$d_i^* = \min_{j \varepsilon N(i)} \{a_{ij} + d_j^*\}, \quad \forall i \ne 1 \tag{6a}$$

$$d_1^* = 0. \tag{6b}$$

If we make the identifications

$$S = C = N, \; U(x) = N(x), \; \overline{F} = F, J^*(x) = d_x^* \tag{7}$$

$$H(x,u,J) = \begin{cases} a_{xu} + J(u) & \text{if } x \ne 1 \\ 0 & \text{if } x = 1 \end{cases} \tag{8}$$

we find that the abstract problem (3) reduces to the shortest path problem.

Example 2 (Infinite Horizon Stochastic Optimal Control Problems): Let H be given by

$$H(x,u,J) = E\{g(x,u,w) + \alpha J[f(x,u,w)] \,|\, x,u\} \tag{9}$$

where the following are assumed:

(1) The parameter w takes values in a countable set W with given probability distribution $p(dw|x,u)$ depending on x and u, and $E\{\cdot|x,u\}$ denotes expected value with respect to this distribution.

(2) The functions g and f map $S \times C \times W$ into $[-\infty,\infty]$ and S respectively.

(3) The scalar α is positive.

Because the set W is assumed countable the expected value in (9) is well defined for all $J \varepsilon F$ in terms of infinite summation provided we use the convention $+\infty - \infty = +\infty$ (see [3], p.31). It is possible to consider a more general probabilistic structure for W (see [3]) at the expense of complicating the presentation but this does not seem worthwhile in view of the computational purposes of the paper.

It is shown in [3] that with this definition of H the abstract problem (3) reduces under more specific assumptions to various types of standard stochastic optimal control problems. Thus if g is uniformly bounded above and below by some scalars and $0 < \alpha < 1$ the problem is equivalent to the standard infinite horizon discounted stochastic optimal control problem with bounded cost per stage (see [5] Sections 6.1-6.3). Under these circumstances the mapping T of (4) has a unique fixed point J^* in the class of all bounded real valued functions on S and J^* is the optimal value function of the corresponding stochastic optimal control problem.

Is we assume that $0 \le g(x,u,w)$ or $g(x,u,w) \le 0$ for all $(x,u,w) \varepsilon S \times C \times W$ then we obtain stochastic optimal control problems of the type discussed extensively, for example, in [5], Sections 6.4-6.6, 7.1-7.4, and [3], Chapter 5. If J^* is the optimal value function for such a problem then J^* is the unique fixed point of T over all functions $J \varepsilon F$ such that $0 \le J \le J^*$ if $0 \le g(x,u,w)$ for all (x,u,w), or $J^* \le J \le 0$ if $g(x,u,w) \le 0$ for all (x,u,w) (see [5], p. 256).

Example 3 (Finite Horizon Stochastic Optimal Control Problems): Let $S, C, U(x)$, W, $p(dw|x,u)$, g and f be as in Example 2 and consider the set of equations

$$J_N(x_N) = 0 \quad , x_N \varepsilon S \tag{10a}$$

$$J_k(x_k) = \inf_{u_k \varepsilon U(x_k)} E\{g(x_k,u_k,w_k) + J_{k+1}[f(x_k,u_k,w_k)] \,|\, x_k,u_k\}, \tag{10b}$$

$$k = 0,1,\ldots,N-1, \; x_k \varepsilon S,$$

where N is a positive integer. These are the usual dynamic programming equations associated with finite horizon stochastic optimal control problems with zero terminal cost and stationary cost per stage and system function. It is possible to write these equations in the form (3) by defining a new state space consisting of an $(N+1)$-fold Cartesian product of S with itself, writing $J^* = (J_0, J_1, \ldots, J_N)$, and appropriately defining H on the basis of (10). In fact this is a standard procedure for converting a finite horizon problem to an infinite horizon problem (see [5], p.325). This reformulation can also be trivially generalized to finite horizon problems involving a nonzero terminal cost and a nonstationary system and cost per stage.

3. A Model for Distributed Dynamic Programming

Our algorithm can be described in terms of a collection of n computation centers referred to as nodes and denoted $1,2,\ldots,n$. The state space S is partitioned into n disjoint sets denoted $S_1,\ldots,S_n$. Each node i is assigned the responsibility of computing the values of the solution function J^* at all states x in the corresponding set S_i. A node j is said to be a neighbor of node i if $j \neq i$ and there exist a state $x_i \varepsilon S_i$ and two functions J_1, $J_2 \varepsilon \overline{F}$ such that

$$J_1(x) = J_2(x), \ \forall x \notin S_j \tag{11a}$$

$$T(J_1)(x_i) \neq T(J_2)(x_i). \tag{11b}$$

The set of all neighbors of i is denoted N(i). Intuitively j is not a neighbor of i if, for every $J \varepsilon \overline{F}$, the values of J on S_j do not influence the values of T(J) on S_i. As a result, for any $J \varepsilon \overline{F}$, in order for node i to be able to compute T(J) on S_i it is only necessary to know the values of J on sets S_j, $j \varepsilon N(i)$, and, possibly, on the set S_i.

At each time instant, node i can be in one of three possible states compute, transmit, or idle. In the compute state node i computes a new estimate of the values of the solution function J^* for all states $x \varepsilon S_i$. In the transmit state node i communicates the estimate obtained from the latest compute phase to one or more nodes m for which $i \varepsilon N(m)$. In the idle state node i does nothing related to the solution of the problem. It is assumed that a node can receive a transmission from neighbors simultaneously with computing or transmitting, but this is not a real restriction since, if needed, a time period in a separate receive state can be lumped into a time period in the idle state.

We assume that computation and transmission for each node takes place in uninterupted time intervals $[t_1,t_2]$ with $t_1 < t_2$, but do not exclude the possibility that a node may be simultaneously transmitting to more than one nodes nor do we assume that the transmission intervals to these nodes have the same origin and/or termination. We also make no assumptions on the length, timing and sequencing of computation and transmission intervals other than the following:

Assumption (A): There exists a positive scalar P such that, for every node i, every time interval of length P contains at least one computation interval for i and at least one transmission interval from i to each node m with $i \varepsilon N(m)$.

Each node i also has one buffer per neighbor $j \varepsilon N(i)$ denoted B_{ij} where it stores the latest transmission from j, as well as a buffer B_{ii} where it stores its own estimate of values of the solution function J^* for all states $x \varepsilon X_i$. The contents of each buffer B_{ij} where $j=i$ or $j \varepsilon N(i)$ at time t are denoted J^t_{ij}. Thus J^t_{ij} is, for every t, a function from S_j into $[-\infty,\infty]$ and may be viewed as the estimate by node i of the restriction of the solution function J^* on S_j available at time t. The rules according to which functions J^t_{ij} are updated are as follows:

1) If $[t_1,t_2]$ is a transmission interval for node j to node i with $i \varepsilon N(j)$ the contents $J^{t_1}_{jj}$ of the buffer B_{jj} at time t_1 are transmitted and entered in the buffer B_{ij} at time t_2, i.e.

$$J^{t_2}_{ij} = J^{t_1}_{jj}. \tag{12}$$

2) If $[t_1,t_2]$ is a computation interval for node i the contents of buffer B_{ii} at time t_2 are replaced by the restriction of the function $T(J^{t_1}_i)$ on S_i where, for all t, J^t_i is defined by

$$J^t_i(x) = \begin{cases} J^t_{ii}(x) & \text{if } x \varepsilon S_i \\ J^t_{ij}(x) & \text{if } x \varepsilon S_j \text{ and } j \varepsilon N(i) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

In other words we have

$$J^{t_2}_{ii}(x) = T(J^{t_1}_i)(x) = \inf_{u \varepsilon U(x)} H(x,u,J^{t_1}_i), \ \forall x \varepsilon S_i. \tag{14}$$

3) The contents of a buffer B_{ii} can change only at the end of a computation interval for node i. The contents of a buffer B_{ij}, $J \varepsilon N(i)$, can change only at the end of a transmission interval from j to i.

Note that by definition of the neighbor set N(i), the value $T(J^t_i)(x)$ for $x \varepsilon S_i$ does not depend on the values of J^t_i at states $x \varepsilon S_m$ with $m \neq i$, and $m \notin N(i)$. We have assigned arbitrarily the default value zero to these states in (13). Our objective is to show that for all $i = 1,\ldots,n$

$$\lim_{t\to\infty} J^t_{ij}(x) = J^*(x), \ \forall x \varepsilon S_j, \quad j = i \text{ or } J \varepsilon N(i).$$

It is clear that an assumption such as (A) is necessary in order for such a result to hold. Since iteration (14) is of the dynamic programming type it is also clear that some restrictions must be placed on the mapping H that guarantee convergence of the algorithm under the usual circumstances where the algorithm is carried out in a centralized, synchronous manner (i.e., when there is only one computation node). The following assumption places somewhat indirect restrictions on H but simplifies the convergence analysis:

Assumption (B): There exist two functions $\underline{J}$ and $\bar{J}$ in F such that the set of all functions $J \in F$ with $\underline{J} \le J \le \bar{J}$ belongs to $\bar{F}$ and furthermore

$$\bar{J} \ge T(\bar{J}), \quad T(\underline{J}) \ge \underline{J} \tag{15}$$

$$\lim_{k\to\infty} T^k(\bar{J})(x) = J^*(x), \quad \forall\, x \in S$$

$$\lim_{k\to\infty} T^k(\underline{J})(x) = J^*(x), \quad \forall\, x \in S \tag{16}$$

where T^k denotes composition of the mapping T with itself k times.

Note that in view of the monotonicity of H [cf. (2)] and the fact $J^* = T(J^*)$, Assumption (B) implies

$$\bar{J} \ge T(\bar{J}) \ge T^2(\bar{J}) \ge \cdots \ge J^* \ge \cdots$$

$$\cdots \ge T^2(\underline{J}) \ge T(\underline{J}) \ge \underline{J}$$

Furthermore if $J \in F$ satisfies $\underline{J} \le J \le \bar{J}$ then $\lim_{k\to\infty} T^k(J)(x) = J^*(x)$ for all $x \in S$.

Assumption (B) leaves open the question of how to find suitable functions $\underline{J}$ and $\bar{J}$. On the other hand for most problems of interest the choices are clear. In particular we have the following:

1) For shortest path problems (Example 1) it is straightforward to verify that the choices

$$\underline{J}(i) = 0 \ , \ \forall\, i = 1,\dots,n \tag{17a}$$

$$\bar{J}(i) = \begin{cases} \infty & \text{if } i \ne 1 \\ 0 & \text{if } i = 1 \end{cases} \tag{17b}$$

satisfy Assumption (B).

2) For finite horizon stochastic optimal control problems (Example 3) for which the function g in (10) is uniformly bounded below it can be easily verified that the functions $\bar{J} = (\bar{J}_o, \bar{J}_1, \dots, \bar{J}_N)$ and $\underline{J} = (\underline{J}_o, \underline{J}_1, \dots, \underline{J}_N)$ where for all k and x

$$\bar{J}_k(x) = \infty \ , \ \underline{J}_k(x) = -\infty \tag{18}$$

satisfy Assumption (B).

3) For discounted infinite horizon stochastic optimal control problems with bounded cost per stage (Example 2 with $\alpha \in (0,1)$ and g uniformly bounded above and below) it is easily shown that every pair of functions $\bar{J}$, $\underline{J}$ of the form

$$\bar{J}(x) = \bar{\beta}, \quad \underline{J}(x) = \underline{\beta}, \quad \forall\, x \in S$$

where

$$\frac{1}{1-\alpha} \sup_{x\in S} \inf_{u\in U(x)} E\{g(x,u,w)\} \le \bar{\beta} < \infty$$

$$-\infty < \underline{\beta} \le \frac{1}{1-\alpha} \inf_{x\in S} \inf_{u\in U(x)} E\{g(x,u,w)\}$$

satisfy Assumption (B).

4) For infinite horizon stochastic optimal control problems with nonpositive cost per stage (Example 2 with $g \le 0$) it can be shown that the functions $\bar{J}$, $\underline{J}$ with

$$\underline{J}(x) = J^*(x), \quad \bar{J}(x) = 0, \quad \forall\, x \in S$$

satisfy Assumption (B) ([5], pp. 261, 298). If the cost per stage is nonnegative (Example 2 with $g \ge 0$) then, under a mild assumption (which is satisfied in particular if U(x) is a finite set for each x), it can be shown that the choices $\bar{J}$, $\underline{J}$ with

$$\underline{J}(x) = 0, \quad \bar{J}(x) = J^*(x), \quad \forall\, x \in S$$

satisfy Assumption (B) ([5], pp. 263, 298). The choice of $\underline{J}$ and $\bar{J}$ can be further sharpened and simplified under more specific assumptions on problem structure but we will not pursue this matter further.

Our convergence result will be shown under the assumption that the contents J_{ij}^o of the buffers B_{ij} at the initial time t=0 satisfy

$$\underline{J}(x) \le J_{ij}^o(x) \le \bar{J}(x), \quad \forall\, x \in S_j. \tag{19}$$

The broad range of initial conditions allowed by (19) eliminates the need to reset the contents of the buffers in an environment where it is necessary to execute periodically the algorithm with slightly different problem data as for example in routing algorithms for communication networks. This is particularly true for cases 1)-3) above where condition (19) implies that the initial buffer contents can be essentially arbitrary.

4. Convergence Analysis

Our main result is the following proposition.

Proposition 1: Let Assumptions (A) and (B) hold and assume that for all i = 1,...,n

$$\underline{J}(x) \le J_{ij}^o(x) \le \bar{J}(x), \quad \forall\, x \in S_j, \ j = i \text{ or } j \in N(i). \tag{20}$$

Then for all i = 1,...,n

$$\lim_{t\to\infty} J_{ij}^t(x) = J^*(x), \quad \forall\, x \in S_j, \ j = i \text{ or } J \in N(i).$$

Proof: Since the contents of a buffer can change only at the end of a computation or transmission interval at a node we can analyze convergence in terms of a discrete time process. We focus attention at the sequence of times $\{t_k\}$ with $0 < t_1 < t_2 < \dots$ where each t_k is the end of a computation interval for one or more nodes.

Let us consider for all $t \ge 0$

$\bar{J}^t_{ij}: S_j \to [-\infty,\infty]$, $\underline{J}^t_{ij}: S_j \to [-\infty,\infty]$, here

where for each $x \in S_j$, the value $\bar{J}^t_{ij}(x)$ [$\underline{J}^t_{ij}(x)$] represents the contents of buffer B_{ij} at time t if the algorithm were executed with the same timing and order of computation and transmission intervals but with the initial condition $\bar{J}(x)$ [$\underline{J}(x)$] instead of $J^o_{ij}(x)$ for each buffer B_{ij} and $x \in S_j$. The monotonicity assumption (2) and the definition of the algorithm [cf. (13), (14)] clearly imply that for all t

$$\underline{J}^t_{ij}(x) \le J^t_{ij}(x) \le \bar{J}^t_{ij}(x), \quad \forall x \in S_j, \quad i = 1,\dots,n, \ j = i \text{ or } j \in N(i). \tag{21}$$

It will thus suffice to show that

$$\lim_{t\to\infty} \underline{J}^t_{ij}(x) = J^*(x), \quad \forall x \in S_j, \ i=1,\dots,n, \ j \in N(i) \tag{22}$$

$$\lim_{t\to\infty} \bar{J}^t_{ij}(x) = J^*(x), \quad \forall x \in S_j, \ i=1,\dots,n, \ j \in N(i) \tag{23}$$

In view of the fact $\bar{J} \ge T(\bar{J})$ we have clearly

$$\bar{J}(x) = \bar{J}^o_{ij}(x) \ge \bar{J}^{t_1}_{jj}(x), \quad \forall x \in S_j, \quad i=1,\dots,n, \ j \in N(i) \tag{24}$$

with potential strict inequality only for nodes j for which t_1 was the end of a computation interval. For $t \in [t_1,t_2)$ the content of B_{ij} is either $\bar{J}^o_{ij}$ or $\bar{J}^{t_1}_{jj}$ so from (24) we must have

$$\bar{J}^t_{ij}(x) \ge \bar{J}^{t_1}_{jj}(x), \quad \forall x \in S_j, \quad i=1,\dots,n, \ j \in N(i), \quad t \in [t_1,t_2) \tag{25}$$

$$\bar{J}^{t_1}_{ij}(x) \ge \bar{J}^t_{ij}(x), \quad \forall x \in S_j, \quad i=1,\dots,n, \ j \in N(i), \quad t \in [t_1,t_2).$$

The last relation can also be written as

$$\bar{J}^{t_1}_{jm}(x) \ge \bar{J}^t_{jm}(x), \quad \forall x \in S_m, \quad j=1,\dots,n, \ m \in N(j), \quad t \in [t_1,t_2). \tag{26}$$

In view of the monotonicity of H it follows from (26) and (14) that

$$\bar{J}^{t_1}_{jj}(x) \ge \bar{J}^{t_2}_{jj}(x), \quad \forall x \in S_j \tag{27}$$

with potential strict inequality only for nodes j for which t_2 was the end of a computation interval. Combining (25) and (27) we obtain

$$\bar{J}^{t_1}_{ij}(x) \ge \bar{J}^{t_2}_{jj}(x), \quad \forall x \in S_j, \quad i=1,\dots,n, \ j \in N(i)$$

with potential strict inequality only for nodes j for which either t_1 or t_2 was the end of a computation interval. The preceding argument can be repeated to show that for all k, $i = 1,\dots,n$, and $j \in N(i)$ we have

$$\bar{J}^{t_k}_{ij}(x) \ge \bar{J}^t_{ij}(x) \ge \bar{J}^{t_k}_{jj}(x) \ge \bar{J}^{t_{k+1}}_{jj}(x), \quad \forall x \in S_j, \ t \in [t_k, t_{k+1}). \tag{28}$$

Let k_1 be the first integer for which $2P \le t_{k_1}$ where P is as in Assumption (A). Then each node must have completed at least one computation phase in the interval $[0,P]$ and at least one transmission phase to all nodes in the interval $[P,2P]$. It is easily seen that this together with (28), the monotonicity of H, and the definition of the algorithm implies that for all $t \in [t_{k_1}, t_{k_1+1})$

$$T(\bar{J})(x) \ge \bar{J}^t_{ij}(x) \ge \bar{J}^{t_{k_1}}(x), \quad \forall x \in S_j, \ i=1,\dots,n, \ j \in N(i).$$

This argument can be repeated and shows that if $m(k)$ is the largest integer m such that $2mP \le t_k$ then for all $t \in [t_k, t_{k+1})$

$$T^{m(k)}(\bar{J})(x) \ge \bar{J}^t_{ij}(x) \ge \bar{J}^{t_k}_{jj}(x), \quad \forall x \in S_j, \ j \in N(i). \tag{29}$$

Similarly we obtain for all $t \in [t_k, t_{k+1})$

$$\underline{J}^{t_k}_{jj}(x) \ge \underline{J}^t_{ij}(x) \ge T^{m(k)}(\underline{J})(x), \quad \forall x \in S_j, \quad i = 1,\dots,n \ j \in N(i). \tag{30}$$

By combining (21), (29), and (30) and using Assumption (B) we obtain (22), (23) and the proof of the proposition is complete. Q.E.D.

Note that (21), (29), and (30) provide useful estimates of rate of convergence. In fact by using these relations it is possible to show that in some special cases convergence is attained in finite time.

Proposition 2: In the cases of Examples 1 and 3 with the choices of $\bar{J}$, $\underline{J}$ given by (17) and (18) respectively and J^o_{ij} satisfying (20), there exists a time $\bar{t} > 0$ such that for all $i=1,\dots,n$ and $t \ge \bar{t}$ there holds

$$J^t_{ij}(x) = J^*(x), \quad \forall x \in S_j, \quad j = i \text{ or } j \in N(i).$$

Proof: For Example 3 it is easily seen that there holds

$$T^k(\bar{J})(x) = T^k(\underline{J})(x) = J^*(x), \quad \forall x \in S, \quad k \ge N+1$$

The proof follows from (21), (29) and (30). For example 1 it is easily seen that

$$T^k(\bar{J})(i) = J^*(i), \quad i = 2,\dots,n, \quad k \geq 1.$$

Also for each i, $T^k(\underline{J})(i)$ represents the length of a path starting from i with k links, and each link has positive length. Therefore there exists a $\bar{k}$ such that $T^{\bar{k}}(\underline{J})(i)$ represents length of a path from i to node 1, for otherwise the paths corresponding to $T^k(\underline{J})(i)$ would cycle indefinitely without reaching node 1 and we would have $T^k(\underline{J})(i) \to \infty$. Since $T^k(\underline{J})(i) \leq J^*(i)$ and $J^*(i)$ is the shortest distance from i to 1 we obtain

$$T^k(\underline{J})(i) = J^*(i), \forall i = 2,\dots,n \quad k \geq \bar{k}.$$

The result again follows from (21), (29) and (30).

Q.E.D.

It is possible to construct examples showing that in the case of the shortest path problem the number of iterations needed for finite convergence of $T^k(\underline{J})$ depends on the link lengths in a manner which makes the overall algorithm nonpolynomial.

In many problems of interest the main objective of the algorithm is to obtain a minimizing control law μ^*, i.e. a function $\mu^*: S \to C$ with $\mu^*(x) \in U(x)$ for all $x \in S$ such that

$$H[x,\mu^*(x),J^*] = \min_{u \in U(x)} H(x,u,J^*), \forall x \in S. \qquad (31)$$

It is thus of interest to investigate the question of whether control laws $\mu^t: S \to C$ satisfying

$$\mu^t(x) \in U(x), \quad \forall x \in S \qquad (32)$$

and

$$H[x,\mu^t(x),J_i^t] = \min_{u \in U(x)} H(x,u,J_i^t), \quad \forall x \in S_i, \ i=1,\dots,n \qquad (33)$$

where J_i^t is given for all t by (13), converge in some sense to a control law μ^* satisfying (31). The following proposition shows that convergence is attained in finite time if the sets U(x) are finite and H has a continuity property which is satisfied for most problems of practical interest. A related convergence result can be shown assuming the sets U(x) are compact (c.f. [5], Prop. 5.11).

Proposition 3: Let the assumptions of Proposition 1 hold. Assume also that for every $x \in S$, $u \in U(x)$ and sequence $\{J^k\} \subset \bar{F}$ for which $\lim_k J^k(x) = J^*(x)$ for for all $x \in S$ we have

$$\lim_{k \to \infty} H(x,u,J^k) = H(x,u,J^*). \qquad (34)$$

Then for each state $x \in S$ for which U(x) is a finite set there exists $\bar{t}_x > 0$ such that for all $t \geq \bar{t}_x$ if $\mu^t(x)$ satisfies (32), (33) then

$$H[x,\mu^t(x),J^*] = \min_{u \in U(x)} H(x,u,J^*).$$

Proof: See [6].

5. Conclusions

The analysis of this paper shows that natural distributed dynamic programming schemes converge to the correct solution under very weak assumptions on the problem structure, and the timing and ordering of computation and internode communication. The restrictions on the initial conditions are also very weak. This means that, for problems that are being solved continuously in real time, it is not necessary to reset the initial conditions and resynchronize the algorithm each time the problem data changes. As a result the potential for tracking slow variations in optimal control laws is improved, and algorithmic implementation is greatly simplified.

References

[1] J. McQuillan, G. Falk, and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm", IEEE Trans. on Communications, Vol. COM-26, 1978, pp. 1802-1811.

[2] D. P. Bertsekas, "Monotone Mappings with Application in Dynamic Programming", SIAM J. Control and Optimization, Vol. 15, 1977, pp. 438-464.

[3] D. P. Bertsekas, and S. E. Shreve, Stochastic Optimal Control: The Discrete-Time Case, Academic Press, N.Y., 1978.

[4] E. L. Lawler, Combinatorial Optimization: Networks and Matroids, Holt, Rinehart, and Winston, N.Y., 1976.

[5] D. P. Bertsekas, Dynamic Programming and Stochastic Control, Academic Press, N.Y., 1976.

[6] D. P. Bertsekas, "Distributed Dynamic Programming", Report LIDS-P-1060, Lab. for Information and Decision Systems, M.I.T., Cambridge, MA, 1980 (to appear in IEEE Trans. on Aut. Control).