RESOURCE CONTENTION, SYNCHRONIZATION, AND INFORMATION STRUCTURE IN DISTRIBUTED SYSTEMS

FINAL REPORT

by

Professors P. Varaiya and E. Wong

to

Office of Naval Research
Contract NO0014-80-C-0507

July 1, 1980 to July 31, 1984

ELECTRONICS RESEARCH LABORATORY
College of Engineering
University of California, Berkeley
94720

84 07 12 007
Resource Contention, Synchronization, and Information Structure in Distributed Systems

P. Varaiya and E. Wong

Electronics Research Laboratory
University of California
Berkeley, CA 94720

Office of Naval Research
Mathematical and Information Science
800 N. Quincy Street, Arlington, VA 22217

June 1984

unlimited

Our research has focused on two generic problems. The first deals with the situation where there is a finite number of discrete resources which must be allocated among a number of competing processes (users). The second problem concerns the (centralized) control of a stochastic system where information is collected by severa15 "local" sensors. The local sensors constitute a distributed data base. The control must be based on information obtained from this data base under the constraint of limited communication capacity.
1. Introduction

This is the final technical report for ONR Contract N00014-80-C-0507, entitled "Resource Contention, Synchronization, and Information Structure in Distributed Systems".

Since details of the research sponsored by this contract are available in the various published journal articles and working papers, we limit ourselves here to a summary of this research.

Our research has focused on two generic problems. The first deals with the situation where there is a finite number of discrete resources which must be allocated among a number of competing processes (users). The second problem concerns the (centralized) control of a stochastic system where information is collected by several "local" sensors. The local sensors constitute a distributed data base. The control must be based on information obtained from this data base under the constraint of limited communication capacity.

2. Resource Sharing

We have followed two approaches.

To introduce the first approach think of a multi-programming environment where there are several processes (programs) which access a common resource, say the CPU. Each process may be quite complex to describe since, in addition to accessing the CPU, it will access various other devices, interact with the user, and so on. However, the interaction among processes is quite simple: there is conflict only when two or more processes simultaneously request the CPU. In this situation it is reasonable to look for resource allocation strategies that require only a highly aggregated information about each process. For instance, one may look for strategies that only need to know which processes are requesting the CPU, together with a statistical description of each process which gives how frequently it will request access to the CPU and how long it will hold it once it has been given this access.

We have studied this situation in a series of papers. The basic model considers a single resource which is either idle or busy. The state of the $i$th process, $z_i(t)$, is either 0, indicating that the process is "thinking", or it is 1, indicating it is requesting access to the resource. The $i$th process is then described by the distribution of the amount of time that it spends in the thinking state before it requests the resource again, and the distribution of time that it will hold the resource before it relinquishes it and resumes thinking. Observe that the thinking state is in reality an aggregate category that covers all activities of the process when it is not requesting or holding the resource. The state of the entire system is given by the vector $X(t) := (z_1(t), \ldots, z_N(t))$.

At any time $t$ the "active" processes constitute the set $A(t) := \{i \mid z_i(t) = 1\}$. The problem is to select at each $t$ the process $i$ in $A(t)$ to which the resource should be allocated. The answer depends on (a) the thinking and resource holding time distributions, (b) the performance criterion adopted, and (c) the restrictions imposed on the permissible allocation rules, chiefly whether one is or is not allowed to preempt a process.

In [2] the problem is addressed from a game-theoretic view. Two processes are considered, and a first come first served allocation strategy is assumed. We ask if it is possible to cooperatively re-divide the thinking and holding times to improve the performance of both processes. It is shown that this is impossible. In other words, every division will favor one or the other process.

In [4,5] we allow an arbitrary number of processes sharing the same resource. We seek to make precise the intuition that since these processes
interact only when they simultaneously request the resource, therefore many statistics of the system must be insensitive to the allocation strategy employed. We find the remarkable result that if the average thinking time is the same for all resources, then the average utilization of the resource is independent of the resource allocation strategy. In fact, the result is deeper in that several hitting time distributions are shown to have this invariance property.

The same model is further investigated in [3]. We concentrate on a particular objective—the maximization of resource utilization. A very counter-intuitive result is shown: utilization is maximized by the strategy that allocated the resource to the process that has least average thinking time. In particular, the optimal strategy does not depend on the resource holding times.

To introduce the second approach consider the following abstract problem. There are $N$ processes, and one resource. Process $i$ is described by a sequence $(X^i(s), F^i(s)), s = 1, 2, \ldots$, where $X^i(s)$ is the immediate (random) reward obtained when process $i$ uses the resource for the $s$th time, and $F^i(s)$ is the $\sigma$-field representing the information about process $i$ obtained when it uses the resource for $(s-1)$ times. At each time $t$ only one process may use the resource. For any given allocation strategy, let $R(t)$ be the reward obtained at time $t$. The problem is to find the strategy that maximizes the expected discounted reward,

$$E \sum_{t=1}^\infty \beta^t R(t),$$

where $0 < \beta < 1$ is a fixed discount factor.

This is a sweeping generalization of the multi-armed bandit problem. For each process $i$ and $s \geq 0$ define the index

$$\nu(i) := \max_{\tau \geq 1} \frac{E \left\{ \sum_{t=s}^\tau a^i(t) \mid F^i(s) \right\}}{E \left[ \sum_{t=s}^\tau a^t \mid F^i(s) \right]}$$

where the maximization is over all stopping times $\tau$ of the filtration $\{F^i(s)\}$. In [8,9] we prove this remarkable result: The optimal strategy is to assign the resource to the process with the largest current index. This result is very significant because the index of a process is a property only of the process and does not depend upon the other processes. Thus the calculation of the optimal strategy involves solving $N$ "one-dimensional" problems rather than one "$N$-dimensional problem.

In [10], the arguments developed in [9] are used to extend the classical result of the $c\mu$ rule to arbitrary arrival processes.

3. Distributed Information

To fix ideas consider the following situation. A stochastic system is being controlled by a single decision maker (DM). There are $N$ local stations. At each time $t$ the $i$th station observes the random variable $y^i(t)$ which is a (possibly noise-corrupted) function of the state $X(t)$. Thus at time $t$ the $i$th station has observed the data sequence $Y^i(t) := \{y^i(1), \ldots, y^i(t)\}$. It is useful to think of

$$Y(t) := Y^1(t) \times \ldots \times Y^N(t)$$

as a distributed data base.

At time $t$ the DM must select a control value $u(t)$. This choice is guided by the DM's objective and on the information available to the DM. We concentrate
on the latter factor. Clearly, the choice of the control is improved by having
more information. However, allowing unrestricted information presupposes that
there are no constraints placed due either to limited communication capacity
or limited information processing capacity on the part of the DM. This is unreal-
istic in many situations. Out work has attempted to impose such constraints
explicitly.

In [1,11] we consider the situation where several local stations transmit
their current estimate of the state. Each station updates its current estimate
on the basis of its private information $Y(t)$ and the messages it has received
from the others. It is easy to see that when a station receives the estimates
from the others, as a first step it will attempt to “uncover” the other station’s
private observations. This is a computationally intensive procedure. However,
our results do show that the procedure does save communication capacity.
Second, the way a station “uncovers” this information depends on its interpreta-
tion of the other station’s model. In [1] it is assumed that all stations have the
same model of the system, in [11] we permit different stations to have different
models.

In [1,11] we made the ad hoc assumption that each station transmitted its
estimate to the others. In [6,7] we tackle directly the question of the optimum
message to be transmitted when there is a fixed communication capacity. In [6]
this question is formulated as one of finding an optimal code. In [7] it is formu-
lated within the context of optimal control.

4. References
