THE STABILIZING PHILOSOPHER: ASYMMETRY BY MEMORY AND BY ACTION*

Mohamed G. Gouda
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712-1188

TR-87-12 April 1987

Abstract: Asymmetry in systems of synchronizing processes can be maintained by action or by memory. Systems that are asymmetric by action cannot be self-stabilizing, while those that are asymmetric by memory can be. We illustrate the last point by discussing a new class of systems of dining philosophers: each system in this class is both asymmetric by memory, and self-stabilizing at the same time. Our conclusion: asymmetry by memory is more reliable than asymmetry by action.

Keywords: Asymmetry, dining philosophers, self-stabilization, synchronization.

* This work is supported in part by the Office of Naval Research Contract No. N00014-86-K-0763, and by a contract with MCC.

0. Introduction

Asymmetry has to be maintained in systems where processes may synchronize with one another; examples of such systems are mutual exclusion [2], dining philosophers [3], drinking philosophers [1], resource allocation systems, etc. Dijkstra was the first to observe the need for asymmetry in such systems [4]; his published comments however were understated and their significance went unnoticed by the scientific community. Later Lehman and Rabin made the same observation, but drove the point to prominence [8].

There are two common approaches to maintain asymmetry in a system of synchronizing processes; we refer to these approaches as *asymmetry by action* and *asymmetry by memory.* In the first approach, the programs of different processes in the system are syntactically identical except that they differ in their initial local states, i.e. differ in the initial setting of their program counters or the initial values of their local variables. Thus, the processes start at an asymmetrical global state, and each of their actions is geared to preserving the asymmetry. In other words, asymmetry is a system invariant. For an example of a system that is asymmetric by action, see Chandy and Misra's solution to the drinking philosophers problem [1].

In the asymmetry by memory approach, the programs of different processes are syntactically identical except that they access local constants that have identical names (to satisfy the syntactic resemblance), and different values (to effect asymmetric behaviour). An example of a system that is asymmetric by memory is one that consists of identical processes $P_0, ..., P_{n-1}$, where the program of each P_i accesses its index i. In this case, the indices act as local constants that have identical names but different values. Because these local constants take part in defining the state space of the system, every global state of the system, whether reachable or not, is asymmetric. This trivially implies that asymmetry is a system invariant.

The distinction between asymmetry by action and asymmetry by memory can be summarized as follows. The state space of a system that is asymmetric by action contains at least one symmetric global state, i.e., one where all processes have identical local states. Clearly, since asymmetry is a system invariant, any such state is not reachable from the initial state of the system. On the other hand, as argued before, the state space of a system that is asymmetric by memory has no such states.

The importance of this distinction may go unappreciated until one considers thae property of self-stabilization. A system is said to be *self-stabilizing* iff starting at any state in its state space, it is guaranteed to converge to a "safe" state in a finite number of state transitions. Now, a system that is asymmetric by action cannot be self-stabilizing. This is because if such a system starts at a symmetric global state, it can never converge to an asymmetric global state by the result of Dijkstra, Lehman, and Rabin [4, 8]. On the other hand, systems that are asymmetric by memory can be self-stabilizing. In this paper, we establish this fact by discussing a new class of systems of dining philosophers [3]; each system in this class is both asymmetric by memory and self-stabilizing.

The dining philosophers is widely recognized as the icon of synchronization and resource-allocation systems (which should explain our choice of the problem), and the class of systems of dining philosophers that we present in this paper is indeed interesting in its own right.

1. Philosophers that are Asymmetric by Memory

We consider a *system* of n philosophers $P_0,\ldots,P_{n-1}$ arranged in a circle. The *left neighbor* of each philosopher P_i is $P_{i-1 \bmod n}$, and its *right neighbor* is $P_{i+1 \bmod n}$. Henceforth, we write the subscripts i-1 and i+1 to mean i-1 mod n and i+1 mod n, respectively.

A *state of a philosopher* P_i is a pair

$$(S_i,\ q_i)$$

where $S_i \in \{T,H,F,E\}$, and $q_i \in \{0,1\}$. The component q_i is called the *priority* of philosopher P_i. Informally,

$S_i = T$	indicates that P_i is thinking,
$S_i = H$	indicates that P_i is hungry,
$S_i = F$	indicates that P_i has its left fork, and
$S_i = E$	indicates that P_i has its two forks and is eating.

A *system state* is a string of n pairs:

$$(S_0,\ q_0)\ (S_1,\ q_1)\ \ldots\ (S_{n-1},\ q_{n-1})$$

where $(S_i,\ q_i)$ is a state of philosopher P_i.

The possible activities of a philosopher P_i are defined by *seven state transitions;* each of which is in one of the following *forms:*

$$
\begin{array}{llll}
f_0: & & (\text{present state of } P_i) & \rightarrow (\text{next state of } P_i) \\
f_1: & [\text{present state of } P_{i-1}] & (\text{present state of } P_i) & \rightarrow (\text{next state of } P_i) \\
f_2: & (\text{present state of } P_i) & [\text{present state of } P_{i+1}] & \rightarrow (\text{next state of } P_i)
\end{array}
$$

Notice that in these forms, circular brackets (...) enclose the present and next states of P_i, while rectangular brackets [...] enclose the present states of P_i's neighbors. The seven state transitions of P_i are as follows.

$$
\begin{array}{llll}
t_0: & & (E,\ q) & \rightarrow (T,\ q) \\
t_1: & & (T,\ q) & \rightarrow (H,\ q) \\
t_2: & [T,\ r] & (H,\ q) & \rightarrow (F,\ q) \\
t_3: & (F,\ q) & [T,\ r] & \rightarrow (E,\ q+1 \bmod 2) \\
t_4: & (F,\ q) & [H,\ r] & \rightarrow (E,\ q+1 \bmod 2)
\end{array}
$$

t_5: $[H, r]$ $(H, q) \rightarrow (F, q)$ provided $(r+q) \bmod 2 = v_i$
t_6: $[F, r]$ $(F, q) \rightarrow (H, q)$ provided $(r+q) \bmod 2 = w_i$

v_i and w_i are two constants local to P_i; their values depend on whether n, the number of philosophers in the system, is odd or even. For now, we assume that n is odd, and define the values of v_i and w_i as follows. (The values of v_i and w_i in the case of even n are defined in Section 3.)

$v_0 = 0$
$v_i = 1$ for $i = 1, \ldots, n-1$, and
$w_i = 0$ for $i = 0, \ldots, n-1$.

Notice that the value of v_i is not the same for each P_i; hence the philosophers are asymmetric by memory.

Informally, the philosopher's state transitions can be explained as follows. Transition t_0 means that an eating philosopher can stop eating and start thinking. t_1 means that a thinking philosopher can stop thinking and become hungry. t_2 means that a hungry philosopher whose left neighbor is thinking can "grab" its left fork. t_3 and t_4 mean that a philosopher who has a left fork and whose right neighbor is either thinking or hungry, can grab its right fork and start eating after flipping its own priority. t_5 means that a hungry philosopher whose left neighbor is also hungry can still grab its left fork provided that the sum of priorities of the two philosophers equals v_i. t_6 means that a philosopher that has a left fork and whose left neighbor also has a left fork can give up its left fork provided that the sum of priorities of the two philosophers equals w_i.

Formally, the system has the usual *interleaving semantics*, where one state transition, selected arbitrarily from those that are currently being enabled, is executed at a time. We expect that the reader is familiar with the concepts of: a state transition being enabled at a system state, a system state following another over some state transition, a system state being reachable from another, etc. Based on this semantics, we show next that our system satisfies some interesting properties including self-stabilization.

2. Self-Stabilization and Other Properties

In this section, we prove that our system satisfies the five useful properties of: liveness, self-stabilization, safety, progress, and individual progress. Informally, these properties can be defined as follows. Liveness means that at least one state transition can be executed at each system state. Self-stabilization means that the system is guaranteed to reach a safe state after executing a finite number of state transitions. Safety means that once the system reaches a safe state, then all its subsequent states are safe. Progress means that at least one philosopher eats infinitely often. Individual progress means that every philosopher eats infinitely often. Clearly, individual progress implies progress; however we prove both properties since our proof of individual progress is based on the fact that the system will progress infinitely often.

In order to establish these properties, some notion of fairness has to be assumed. We base our proofs on the following *fairness assumption.* Every state transition that is continuously enabled is eventually executed. Only the proofs of self-stabilization, Theorem 1 below, and individual progress, Theorem 4, make use of this assumption.

Theorem 0: *(Liveness)* At least one state transition is enabled at each system state.

Proof: Let $s = (S_0, q_0) \ldots (S_{n-1}, q_{n-1})$ be any system state. If there is an i, $0 \leq i < n$, such that $S_i = E$ or $S_i = T$ then transition t_0 or t_1 of P_i is enabled at s. Thus, we need only to consider the cases where each S_i is either H or F. There are three such cases, and we show that in each of them at least one transition is enabled at s.

Case 0 (for each i, $S_i = H$): Assume that transition t_4 of each P_i, $i = 1,\ldots,n-1$, is disabled at s; then there are two possibilities to consider, depending on whether $q_0 = 0$ or $q_0 = 1$:

either $(q_0, q_1, q_2, q_3, \ldots, q_{n-1}) = (0, 0, 0, 0, \ldots, 0)$,
or $(q_0, q_1, q_2, q_3, \ldots, q_{n-1}) = (1, 1, 1, 1, \ldots, 1)$.

In either case, transition t_4 of P_0 is enabled at s.

Case 1 (for each i, $S_i = F$): Assume that transition t_5 of each P_i, $i = 1,\ldots,n-1$, is disabled at s; then there are two possibilities to consider, depending on whether $q_0 = 0$ or $q_0 = 1$:

either $(q_0, q_1, q_2, q_3, \ldots, q_{n-1}) = (0, 1, 0, 1, \ldots, 0)$,
or $(q_0, q_1, q_2, q_3, \ldots, q_{n-1}) = (1, 0, 1, 0, \ldots, 1)$.

Recall that n is odd; thus $q_{n-1} = 0$ in the first possibility, and $q_{n-1} = 1$ in the second. In either case, transition t_5 of P_0 is enabled at s.

Case 2 (there exist i and j, $S_i = H$ and $S_j = F$): In this case, there exist two adjacent philosophers P_m and P_{m+1} whose states are such that $S_m = F$ and $S_{m+1} = H$. Hence, t_6 of P_m is enabled. ❑

In order to specify the next two system properties, we need first to define the concept of a safe system state. A system state $(S_0, q_0) \ldots (S_{n-1}, q_{n-1})$ is called *safe* iff for each i, if $S_i = E$ then $S_{i+1} \neq F$ and $S_{i+1} \neq E$.

Theorem 1: *(Self-stabilization)* The system will reach a safe state after a finite number of state transitions.

Proof: Define a *ranking function* f that computes a natural number f(s) for each system state $s = (S_0, q_0) \ldots (S_{n-1}, q_{n-1})$ as follows.

$$f(s) = \sum_{i=0}^{n-1} f_i(s)$$

where $f_i(s) = 1$ if (S_{i-1} = E and S_i = F) or
(S_{i-1} = E and S_i = E) or
(S_i = E and S_{i+1} = F) or
(S_i = E and S_{i+1} = F)
$= 0$ otherwise

Notice that a system state s is safe iff $f(s) = 0$.

Now each execution of a state transition either keeps every f_i unchanged, or reduces the values of some f_i's from one to zero. Moreover, for every i, if $f_i(s) = 1$, then there is a state transition that is continuously enabled until it is executed, and whose execution makes $f_i(s) = 0$. (For example, if S_{i-1} = E and S_i = F then $f_{i-1} = f_i = 1$ and t_0 of P_{i-1} is continuously enabled, and its execution makes $f_{i-1} = f_i = 0$.) By our fairness assumption, this transition is eventually executed yielding $f_i = 0$. After a finite number of state transitions, the system will reach a state s where $f(s) = 0$, i.e., s is safe. ❑

Theorem 2: *(Safety)* Any state that is reachable from a safe state is safe.

Proof: It is sufficient to show that for every pair of system states s and s', if s is safe and s' follows s over some state transition t, then s' is also safe. This can be established in a straightforward way by inspecting the seven cases: $t = t_0, \ldots, t = t_6$. ❑

Theorem 3: *(Progress)* At least one philosopher eats infinitely often.

Proof: It is sufficient to show that transition t_3 or t_4 of at least one philosopher is executed infinitely often. This can be established by (i) defining a ranking function f that computes a natural number for each system state, and (ii) showing that each state transition other than t_3's and t_4's reduces the value of f.

Define f(s) for any system state $s = (S_0, q_0) \ldots (S_{n-1}, q_{n-1})$ as follows:

$$f(s) = \sum_{i=0}^{n-1} f_i(s)$$

where the value of $f_i(s)$ depends on whether $i = 0$ or $i \neq 0$. (That $f_0(s)$ is different from the rest of the $f_i(s)$ is a consequence of our design decision to make $v_0=0$ and $v_i=1$, for each $i \neq 0$, which causes philosopher P_0 to behave differently from the rest of the philosophers.)

For $i = 0$, $f_i(s)$ = 5 iff S_i = E
= 4 iff S_i = T
= 3 iff S_i = H and (S_{i-1} = E or S_{i-1} = T)
= 2 iff S_i = F and (S_{i-1} = E or S_{i-1} = T)
= 1 iff (S_i = H and S_{i-1} = H) or (S_i = F and S_{i-1} = F)
= 0 iff (S_i = H and S_{i-1} = F) or (S_i = F and S_{i-1} = H).

For $i \neq 0$, $f_i(s) = 7$ iff $S_i = E$
$= 6$ iff $S_i = T$
$= 5$ iff $S_i = H$ and $(S_{i-1} = E$ or $S_{i-1} = T)$
$= 3$ iff $S_i = F$ and $(S_{i-1} = E$ or $S_{i-1} = T)$
$= 2$ iff $(S_i = H$ and $(S_{i-1} = H$ or $S_{i-1} = F)$
and $q_i + q_{i-1} \bmod 2 = v_i)$ or
$(S_i = F$ and $(S_{i-1} = H$ or $S_{i-1} = F)$
and $q_i + q_{i-1} \bmod 2 = w_i)$
$= 0$ iff $(S_i = H$ and $(S_{i-1} = H$ or $S_{i-1} = F)$
and $q_i + q_{i-1} \bmod 2 \neq v_i)$ or
$(S_i = F$ and $(S_{i-1} = H$ or $S_{i-1} = F)$
and $q_i + q_{i-1} \bmod 2 \neq w_i)$.

It remains now to show that each state transition, other than t_3 and t_4, of P_i reduces the value of $f_i + f_{i+1}$ by one at least. (Clearly, each of these transitions keeps the values of other f_j's unchanged.) We divide the proof into three cases.

Case 0 (i = 0): Each of the transitions t_0, t_2, t_5, and t_6 of P_0 reduces f_0 by one, and keeps f_1 unchanged. Transition t_1 of P_0 reduces f_0 by one, and does not increase f_1, but may decrease it.

Case 1 (i = n-1): Transition t_0 of P_{n-1} reduces f_{n-1} by one, and keeps f_0 unchanged. Transition t_1 of P_{n-1} reduces f_{n-1} by one, and does not increase f_0. Each of the transitions t_2, t_5, and t_6 of P_{n-1} reduces f_{n-1} by two, and does not increase f_0 by more than one.

Case 2 (i = 1, ..., n-2): Transition t_0 of P_i reduces f_i by one, and keeps f_{i+1} unchanged. Transition t_1 of P_i reduces f_i by one and does not increase f_{i+1}. Finally, each of the transitions t_2, t_5, and t_6 reduces f_i by two, and keeps f_{i+1} unchanged. ❑

Theorem 4: *(Individual progress)* Every philosopher eats infinitely often.

Proof: From Theorem 3, it is sufficient to prove that if a philosopher eats infinitely often, then its left neighbor will also eat infinitely often. Assume that P_i eats infinitely often starting from some system state s, and that P_{i-1} does not eat at s. We show that P_{i-1} will eat at some later state. Since $S_{i-1} \neq E$ at s, there are two cases to consider.

Case 0 ($S_{i-1} = T$ at s): In this case, transition t_1 of P_{i-1} is enabled at s and at all subsequent states until it is executed, by the fairness assumption, yielding $S_{i-1} = H$ at some later state s'. State s' satisfies the following case.

Case 1 ($S_{i-1} = H$ or $S_{i-1} = F$ at s): Assume that P_{i-1} does not eat after s, i.e., $S_{i-1} = H$ or $S_{i-1} = F$ at each state after s. Thus, transition t_2 of P_i can never be executed after s. Moreover, since the priority of P_i is flipped each time P_i eats, transition t_4 of P_i can be executed at most once after s. Since neither t_2 nor t_4 of P_i can be executed infinitely often after s, P_i

cannot eat infinitely often after s, contradicting our original assumption. Therefore, P_{i-1} will eat after s. ❑

3. Extensions and variations

So far our discussion has been limited to systems with odd number of philosophers, i.e., odd n; we now turn our attention to systems with even n. The only difference between systems with even n, and those with odd n is in the values of the constants v_i and w_i. Define the values of v_i and w_i for a system with even n as follows.

$$
\begin{array}{lll}
v_0 & = 0 & , \\
v_i & = 1 & \text{for } i = 1, \ldots, n-1, \\
w_0 & = 1 & , \text{ and} \\
w_i & = 0 & \text{for } i = 1, \ldots, n-1.
\end{array}
$$

With these values, all the proofs in the previous section remain the same except for the proof of Theorem 3; which is modified as follows.

Proof of progress for even n: Define f(s), for any system state $s = (S_0, q_0) \ldots (S_{n-1}, q_{n-1})$ as follows:

$$f(s) = \sum_{i=0}^{n-1} f_i(s)$$

$$
\begin{array}{lll}
\text{where } f_i(s) & = 5 & \text{if } S_i = E \\
& = 4 & \text{if } S_i = T \\
& = 3 & \text{if } S_i = H \text{ and } (S_{i-1} = E \text{ or } S_{i-1} = T) \\
& = 2 & \text{if } S_i = F \text{ and } (S_{i-1} = E \text{ or } S_{i-1} = T) \\
& = 1 & \text{if } (S_i = H \text{ and } (S_{i-1} = H \text{ or } S_{i-1} = F) \\
& & \quad \text{and } q_i + q_{i-1} \bmod 2 = v_i) \text{ or} \\
& & \quad (S_i = F \text{ and } (S_{i-1} = H \text{ or } S_{i-1} = F) \\
& & \quad \text{and } q_i + q_{i-1} \bmod 2 = w_i) \\
& = 0 & \text{if } (S_i = H \text{ and } (S_{i-1} = H \text{ or } S_{i-1} = F) \\
& & \quad \text{and } q_i + q_{i-1} \bmod 2 \neq v_i) \text{ or} \\
& & \quad (S_i = F \text{ and } (S_{i-1} = H \text{ or } S_{i-1} = F) \\
& & \quad \text{and } q_i + q_{i-1} \bmod 2 \neq w_i)
\end{array}
$$

It remains now to show that each transition, other than t_3 and t_4, of P_i reduces the value of $f_i + f_{i+1}$ by one at most. (Clearly each of these transitions keeps the values of other f_j's unchanged.) Each of the transitions t_0, t_2, t_4, and t_5 reduces f_i by one, and keeps f_{i+1} unchanged. Transition t_1 reduces f_i by one, and does not increase f_{i+1}, but may decrease it. This completes the proof. ❑

Notice that in case of even n, $v_i \neq w_i$ for each i. Therefore, it is possible in this case to get rid of the w_i's completely by replacing the condition $(r+q) \bmod 2 = w_i$ by $(r+q) \bmod 2 \neq v_i$ in transition t_6.

Finally, we observe that our fairness assumption can be relaxed somewhat. In particular, any of the t_1 transitions can be allowed to be continuously enabled without ever being executed. Thus, a thinking philosopher can continue to think indefinitely, and never become hungry. With this relaxation, the statements of Theorems 3 and 4 need to be changed as follows. (The statements of the other theorems remain unchanged.)

(Progress) If a philosopher is ever hungry, then eventually at least one philosopher eats.

(Individual progress) If a philosopher is ever hungry, then eventually this philosopher eats.

The proofs in the previous section need to be modified slightly to accommodate these changes.

4. Concluding remarks

We have discussed a class of systems whose members are both asymmetric by memory and self-stabilizing. Since systems that are asymmetric by action cannot be self-stabilizing, we conclude that asymmetry by memory is a better strategy than asymmetry by action in designing synchronization systems. For other examples of systems that are both asymmetric by memory and self-stabilizing, we refer the reader to Lamport's paper [7].

The discriminating power of self-stabilization is worth noting. Indeed, if it was not for self-stabilization, the important distinction between asymmetry by memory and asymmetry by action could have gone unnoticed, or unappreciated.

Finally, we observe that our endeavor to design self-stabilizing dining philosophers has paid off in an unexpected way: the resulting systems are extremely simple, despite being highly reliable. This observation is in complete agreement with the previous experience in designing self-stabilizing systems [0, 4, 5, 6, 7].

Acknowledgement: My debt to Edsger W. Dijkstra is an invariant. The comments of G. M. Brown, M. Broy, K. M. Chandy, S. S. Lam, J. Misra, and N. Multari were helpful in improving the presentation.

References

[0] G. M. Brown, M. G. Gouda, and C. L. Wu, "Token systems that self-stabilize," submitted to *IEEE Trans. on Computers*, 1986.

[1] K. M. Chandy and J. Misra, "The drinking philosophers problem," *ACM TOPLAS*, Vol. 6, No. 4, Oct. 1984, pp. 632-646.

[2] E. W. Dijkstra, "Solution of a problem in concurrent programming control," *CACM*, Vol. 8, No. 9, Sept. 1965, p. 569.

[3] E. W. Dijkstra, "Hierarchical ordering of sequential processes," in *Operating Systems Techniques*, C. A. R. Hoare and R. H. Perrott, Eds., Academic Press, New York, 1972.

[4] E. W. Dijkstra, "EWD391 Self-stabilization in spite of distributed control," 1973. Reprinted in *Selected Writings on Computing: A personal perspective*, Springer-Verlag, Berlin, 1982, pp. 41-46.

[5] E. W. Dijkstra, "Self stabilizing systems in spite of distributed control," *CACM*, Vol. 17, No. 11, Nov. 1974, pp. 643-644.

[6] N. J. Multari, "Self-stabilizing protocols," *Ph.D. Thesis*, Dept. of Computer Sciences, University of Texas at Austin, in preparation 1987.

[7] L. Lamport, "The mutual exclusion problem: Part II—Statement and solutions," *JACM*, Vol. 33, No. 2, April 1986, pp. 327-348.

[8] D. Lehman and M. Rabin, "On the advantages of free choice: a symmetric and fully distributed solution of the dining philosophers problem," *Proceedings of the 8th Annual ACM Symposium on Principles of Programming Languages*, ACM, New York, 1981, pp. 133-138.