Sharing Resources at Nonuniform Access Rates

Valmir C. Barbosa[*]
Mario R. F. Benevides[*†]
Felipe M. G. França[*]

[*]Programa de Engenharia de Sistemas e Computação, COPPE
[†]Instituto de Matemática
Universidade Federal do Rio de Janeiro
Caixa Postal 68511
21945-970 Rio de Janeiro - RJ, Brazil

Abstract

We introduce DPPr (for "Dining Philosophers Problem with rates") as a generalization of the heavy-load case of the Dining Philosophers Problem (DPP) in which processes are required to be scheduled to access shared resources with pre-specified relative frequencies. DPPr is an abstraction of resource-sharing problems to which the synchronization of some distributed algorithms for neural-network models and the generation of timing signals in asynchronous digital circuits, among other problems, are thought to be related. Two fully distributed, asynchronous solutions are given for DPPr in this paper. The first solution employs a reduction to heavy-load DPP and after that a distributed scheduling mechanism that has been used to solve the latter problem with optimal concurrency. The second solution tackles the DPPr instance directly by operating on a multigraph based on that instance.

1. Introduction

Let G be a connected undirected graph whose nodes represent processes and whose edges stand for the possibility of resource sharing between the processes that they interconnect. In the Dining Philosophers Problem (DPP) (Dijkstra, 1968; Chandy and Misra, 1984) a process requires access to all resources it shares with all of its neighbors whenever it requires access to any resources, and consequently neighbors in G are precluded from accessing resources concurrently. One variation of DPP that has been used frequently both to stress a solution's possibilities and to clarify the problem's intrinsic constraints is the heavy-load variation in which processes keep cycling through a state of resource demand continually, in such a way that they only refrain from nonstop access to resources because of the usual fairness requirement that all processes must acquire the resources that they need within a finite time after the need has arisen. Solutions to this variation usually result in equal rates of resource access for all processes in the long run (Chandy and Misra, 1984; Barbosa, 1986; Barbosa and Gafni, 1989).

In this paper we consider a generalization of this heavy-load variation. In our generalization, the rate at which a process is to be granted access to the resources it needs is pre-specified. In the long run, solutions to this generalized formulation are expected to yield a pattern of resource usage in which the various processes participate, relatively to one another, with frequencies that reflect the pre-specified rates. We refer to this problem as the Dining Philosophers Problem with rates (DPPr).

Our discussion is based on viewing G as a message-passing system. Under this view, the sole means of communication among processes is the sending of point-to-point messages on bidirectional communication channels that interconnect processes pairwise according to the structure of G. In addition, throughout most of the paper we assume a fully synchronous model of distributed computation. In this model, all processes are driven by a common global clock, local computation takes no time, and messages sent between neighbors at a clock pulse are guaranteed to be delivered before the next pulse. The assumption of such a model allows most of our discussion to be simplified considerably, without however hindering the extension of our conclusions to hold for the fully asynchronous model too, as we shall discuss later. In most of what follows, then, distributed computations on G are assumed to take place at synchronous pulses represented by the integer $s \geq 0$.

The following is how the remainder of the paper is organized. DPPr is stated precisely in Section 2, and in addition the heavy-load variation of DPP is re-introduced as the constant-rate particular case of DPPr. In Section 3, we review the method of Scheduling by Edge Reversal (SER) as a solution to this variation of DPP, and in Section 4 show how it can be employed to solve DPPr on an undirected graph that is derived from G by employing node multiplicity as a function of the pre-specified rates. Another solution, now employing edge multiplicity instead, is presented in Section 5. Section 6 contains a discussion of how to extend all results to hold under full asynchronism. Concluding remarks follow in Section 7.

2. The problem

Let N be the node set of G (i.e., N is the set of processes), with $n = |N|$, and E its set of edges. For $n_i \in N$, let r_i be a positive integer, and for $(n_i, n_j) \in E$ let $g_{ij} = \gcd(r_i, r_j)$. For $s \geq 0$, DPPr asks for a sequence $\langle S_s \rangle$ of subsets S_s of N to be determined representing the sets of nodes to be granted access to resources at pulse s. This sequence must be such that:

(i) S_s is an independent set;

(ii) For all $(n_i, n_j) \in E$ such that $r_i \geq r_j$, there exists $s_0 \geq 0$ such that n_i and n_j appear in the sequence $S_{s_0}, S_{s_0+1}, \ldots$ repeatedly in a pattern that alternates either $\lfloor r_i/r_j \rfloor$ or $\lceil r_i/r_j \rceil$ appearances of n_j with one appearance of n_i until n_i has appeared r_j/g_{ij} times. The choice between $\lfloor r_i/r_j \rfloor$ and $\lceil r_i/r_j \rceil$ must be such that $\lceil r_i/r_j \rceil$ is picked exactly $(r_i \bmod r_j)/g_{ij}$ times, while in the remaining $(r_j - r_i \bmod r_j)/g_{ij}$ times $\lfloor r_i/r_j \rfloor$ is picked. At this point, one of the possible patterns is started.

Condition (i) represents the requirement of mutual exclusion in the access to shared resources, while condition (ii) embodies the usual liveness and fairness requirements in the context of DPPr. Condition (ii) is worth examining in some more detail, as it yields the following interpretation of the integers r_i. For $(n_i, n_j) \in E$ and every integer $K \geq 1$, there exists $s > s_0$ such that the number of sets containing n_i or n_j in $S_{s_0}, \ldots, S_s$ is $K(r_i+r_j)/g_{ij}$. In this sequence of independent sets, the ratio of the number of sets containing n_i to the number of sets containing n_j is r_j/r_i. Condition (ii) is then a requirement that the rates at which n_i and n_j appear in the sequence of independent sets be related to each other proportionally to r_j/r_i.

For $(n_i, n_j) \in E$, if we let p_{ij} denote the number of independent sets containing either n_i or n_j necessary for one of the patterns alluded to in condition (ii) to be completed, then it follows from the observation we just made that

$$p_{ij} = \frac{r_i + r_j}{g_{ij}}. \tag{1}$$

In addition to guaranteeing the appearance of neighbors n_i and n_j in independent sets with rates proportional to r_j/r_i, condition (ii) also ensures, by means of a simple transitivity argument, that the same must also hold if n_i and n_j are not neighbors. However, in this case p_{ij} can be expected to be highly dependent upon the structure of G, as well as upon the algorithm used to generate the sequence of independent sets. We shall return to this issue in Section 4.

One should note that the sole requirement that nodes appear in the sequence of independent sets at relative rates as discussed would, by itself, entail a much simpler condition in lieu of condition (ii). By specifying condition (ii) as stated, we have also aimed at requiring that the appearances of n_i be intermingled with those of n_j as "evenly" as possible.

We believe DPPr to be an interesting abstraction of resource-sharing problems in which the orderly maintenance of relative rates with which processes access resources plays an important role. Two particular areas in which such a scenario seems to arise are the synchronization of some distributed algorithms for models of neural networks (França, 1993; 1994) and the control of asynchronous digital circuits (França, Alves, and Granja, 1995).

The case in which $r_i = r$ for some $r \geq 1$ and all $n_i \in N$ is clearly the case of heavy-load DPP with the additional requirement that neighbors in G take alternating turns to access shared resources. Before we address the general formulation, we pause in the next section to describe a fully distributed solution to this extended case of heavy-load DPP. It is on this solution that one of our solutions to DPPr will be based.

3. SER for heavy-load DPP

In the case of heavy-load DPP with the additional requirement of alternating turns between neighbors, the determination of the sets S_s for $s \geq 0$ can be approached as follows (Chandy and Misra, 1984; Barbosa, 1986; Barbosa and Gafni, 1989). Suppose that initially G is oriented by an acyclic orientation, and let the sinks according to this orientation be the members of S_0. For $s > 0$, S_s is obtained by turning all the sinks in S_{s-1} into sources, thereby yielding a new orientation of G, and then taking the sinks according to this new orientation to constitute S_s. Of course this can only be feasible if the new orientation is guaranteed to be acyclic, which must be true because a directed cycle according to the resulting orientation would have to go through at least one of the newly obtained sources, which is impossible.

In this sequence of acyclic orientations, the essential mechanism is the reversal of the directions of all edges incident to sinks to yield sources in the resulting orientation, thence the denomination of the method as Scheduling by Edge Reversal (SER). The SER method has several appealing properties, including its concurrency-wise optimality given the requirement that neighbors alternate, and the ease with which a fully distributed implementation can be carried out (the turning of a sink into a source is simply the sending of one message to each of the sink's neighbors) (Barbosa, 1986; Barbosa and Gafni, 1989). Two of its properties that will be instrumental in later sections are stated next as Lemmas 1 and 2. We let $\omega_0, \omega_1, \ldots$ be the sequence of acyclic orientations whose sets of sinks are $S_0, S_1, \ldots$.

Lemma 1. (Lemma 3.18 of Barbosa, 1986) *Consider a simple cycle of G comprising the nodes $n_1, \ldots, n_c$ for some c such that $3 \leq c \leq n$, and let ω_s, for some $s \geq 0$, be such that* *all edges on the cycle are oriented by ω_s from n_i to n_j if $i < j$ for $i, j \in \{1, \ldots, c\}$. Then $n_1, \ldots, n_c$ appear as sinks in $\omega_s, \omega_{s+1}, \ldots$ cyclically in the order of decreasing subscripts.* ■

Lemma 2 is concerned with how much concurrency can be attained by SER given the structure of G and ω_0. This concurrency, denoted by $Conc_D(\omega_0)$, is defined as the average, over n and t, of the number of sinks in the sequence $\omega_0, \ldots, \omega_t$ in the limit as $t \to \infty$. Before we state the lemma, though, it is important to recognize that the sequence $\omega_0, \omega_1, \ldots$ must become periodic at some pulse, after which a sequence of $p(\omega_0) \geq 2$ orientations is cyclically repeated endlessly. During each of these repetitions, every node can be shown to be a sink in exactly the same number of orientations; we let $m(\omega_0) \geq 1$ denote this number.

Lemma 2. (Theorem 11 of Barbosa and Gafni, 1989) $Conc_D(\omega_0) = m(\omega_0)/p(\omega_0)$. ■

4. A solution based on node multiplicity

In this section we present our first solution to DPPr. This solution is based on the construction of a heavy-load instance of DPP from the instance of DPPr given by G and the integers $r_1, \ldots, r_n$. We let $G' = (N', E')$ be the undirected graph associated with the resulting instance of DPP, denote by L the least common multiple of $r_1, \ldots, r_n$, and define $q_i = L/r_i$ for all $n_i \in N$. The following are the steps to obtain G'.

1. For each $n_i \in N$, let N' contain q_i nodes, denoted $n_i^1, \ldots, n_i^{q_i}$. Let $Q_i = \{n_i^1, \ldots, n_i^{q_i}\}$.

2. For each $(n_i, n_j) \in E$, let E' contain the $q_i + q_j$ edges needed for the nodes in $Q_i \cup Q_j$ to be arranged in a cycle. Assuming that $q_j \geq q_i$, these edges are to be deployed in such a way that, beginning at either $n_j^{q_j}$ or $n_i^{q_i}$, a traversal of the cycle in decreasing order or superscripts alternates either $\lfloor q_j/q_i \rfloor$ or $\lceil q_j/q_i \rceil$ nodes in Q_j with one node in Q_i, until $q_i/\gcd(q_i, q_j)$ nodes from Q_i have been traversed. The choice between $\lfloor q_j/q_i \rfloor$ and $\lceil q_j/q_i \rceil$ must be such that $\lceil q_j/q_i \rceil$ is picked exactly $(q_j \bmod q_i)/\gcd(q_i, q_j)$ times, while $\lfloor q_j/q_i \rfloor$ is picked in the remaining $(q_i - q_j \bmod q_i)/\gcd(q_i, q_j)$ times. At this point, the cycle's first segment is completed. The process continues likewise until the $\gcd(q_i, q_j)$ segments needed to use up all $q_i + q_j$ nodes have been built. Segments are linked to one another by taking them in the order they were built and connecting the last node added to a segment to the first node added to the next segment. At the end, the cycle returns to the first node added (either $n_j^{q_j}$ or $n_i^{q_i}$) from the last one added (either n_i^1 or n_j^1, respectively).

It should be noted right away that each of the segments described in step 2 comprises $q_j/\gcd(q_i,q_j)$ nodes from Q_j and $q_i/\gcd(q_i,q_j)$ nodes from Q_i. The first of these quantities is such that

$$
\begin{aligned}
\frac{q_j}{\gcd(q_i,q_j)} &= \frac{L/r_j}{\gcd(L/r_i, L/r_j)} \\
&= \frac{L/r_j}{(L/r_i r_j)g_{ij}} \\
&= \frac{r_i}{g_{ij}},
\end{aligned}
\tag{2}
$$

and the second, similarly, yields

$$
\frac{q_i}{\gcd(q_i,q_j)} = \frac{r_j}{g_{ij}}. \tag{3}
$$

It is also immediate to see, from (1), (2), and (3), that

$$
p_{ij} = \frac{q_i+q_j}{\gcd(q_i,q_j)}. \tag{4}
$$

Having built G', we next construct an orientation, call it ω_0', of the edges in E'. For every $(n_i,n_j)\in E$ such that $q_j \geq q_i$, the edges constituting the simple cycle described in step 2 are oriented by ω_0' in such a way that, in the subgraph induced in G' by $Q_i \cup Q_j$, the first node added to the cycle is the only sink and the last node added is the only source. What this amounts to is that, from the latter node, in that induced subgraph there are two directed paths to the former node, one of length 1 and the other of length q_i+q_j-1.

Figure 1 illustrates the process of constructing G' for two neighbors n_i and n_j such that $q_i=6$ and $q_j=16$. The figure shows two rows of nodes, the topmost corresponding to n_j, the other to n_i. Nodes in the topmost row are, left to right, $n_j^1,\ldots,n_j^{16}$, while those in the other row are $n_i^1,\ldots,n_i^6$. Edges employed within each of the segments are represented by dashed lines, while the edges that link segments together are solid. The number of segments is in this case two, each comprising three nodes from Q_i alternating with two groups of three nodes from Q_j and another of two nodes. Edges in the figure are oriented according to ω_0'.

One important aspect of orientation ω_0' is that its acyclicity depends on how the groups of nodes of size $\lfloor q_j/q_i \rfloor$ or $\lceil q_j/q_i \rceil$ are selected in step 2 of the construction of G'. Because there are selections that lead ω_0' to contain directed cycles, the remainder of this section is predicated upon the possibility of selecting groups of nodes in step 2 that render ω_0' acyclic. In Section 5, we show that such a possibility always exists and demonstrate how to perform step 2 correctly.

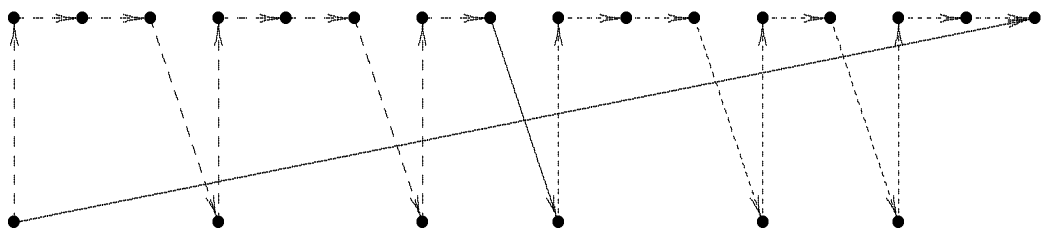

Figure 1. Construction of G'

The intended relationship between the original instance of DPPr and the resulting instance of DPP is that we say that a DPPr process, say $n_i \in N$, has succeeded in accessing shared resources whenever one of the q_i DPP processes $n_i^1, \ldots, n_i^{q_i}$ has succeeded in doing so. This is formalized next.

Theorem 3. *The use of SER from an acyclic ω_0' on the heavy-load instance of DPP given by G' solves the instance of DPPr given by G and $r_1, \ldots, r_n$.*

Proof: By the assumed acyclicity of ω_0', the SER mechanism can be used as described in Section 3 to solve the instance of DPP given by G'. What we have to show, under the interpretation that access to shared resources by some n_i^k means access to shared resources by n_i, is that conditions (i) and (ii) are satisfied. Let us then consider two nodes n_i and n_j such that $(n_i, n_j) \in E$. By construction of G' and of ω_0', it follows from Lemma 1 that no two nodes from $Q_i \cup Q_j$ can ever belong to the set of sinks at any pulse, so condition (i) is satisfied.

Lemma 1 is also the starting point to show that condition (ii) is satisfied as well, because it asserts that not only do the nodes in $Q_i \cup Q_j$ become sinks one at a time, but also that they become sinks cyclically in the order employed in step 2 of the generation of G'. By (2) and (3), this leads precisely to the requirement of condition (ii). Note, incidentally, that within each periodic round of $q_i + q_j$ orientations in which the nodes of $Q_i \cup Q_j$ are sinks we find the periodicity with which n_i and n_j can be taken to have accessed shared resources, whose duration is given by (4). ∎

Before ending this section, there are two remarks to be made. The first remark is that, as we mentioned in Section 2, (1) is only valid when n_i and n_j are neighbors. Otherwise, the only clue that we have concerning the value of p_{ij} comes from Theorem 3 and from our observation in Section 3 that the sequence of acyclic orientations from ω_0' under SER eventually becomes periodic with period $p(\omega_0')$. We then have $p_{ij} \leq p(\omega_0')$.

For the second remark, let us discuss, similarly to what we did for heavy-load instances of DPP under SER in Section 3, the issue of how much concurrency the solution asserted by Theorem 3 can be expected to yield. From the standpoint of the heavy-load instance of DPP represented by G', we have, by Lemma 2,

$$Conc_D(\omega'_0) = \frac{m(\omega'_0)}{p(\omega'_0)}. \tag{5}$$

Let us define a similar measure from the standpoint of DPPr, denoted by $Conc_R(\omega'_0)$, as the average, over n and over t, of the number of times nodes $n_i \in N$ have accessed shared resources as a result of nodes in Q_i having become sinks. Theorem 4 relates the two concurrency measures.

Theorem 4.

$$Conc_R(\omega'_0) = \frac{\sum_{n_i \in N} q_i}{n} Conc_D(\omega'_0).$$

Proof: Let $m_i(t)$ denote the number of times a sink from Q_i has appeared in the sequence of acyclic orientations $\omega'_0, \ldots, \omega'_t$ for some $t \geq 0$. By Theorem 3, and for $K = \lfloor t/p(\omega'_0) \rfloor$, $m_i(t) = Kq_i m(\omega'_0) + o(t)$. We then have, by (5),

$$\begin{aligned} Conc_R(\omega'_0) &= \lim_{t\to\infty} \frac{1}{nt} \sum_{n_i \in N} m_i(t) \\ &= \lim_{t\to\infty} \frac{Km(\omega'_0)\sum_{n_i \in N} q_i + o(t)}{nKp(\omega'_0) + o(t)} \\ &= \frac{\sum_{n_i \in N} q_i}{n} Conc_D(\omega'_0). \end{aligned}$$

■

5. A solution based on edge multiplicity

In this section we discuss another alternative to solve DPPr. The approach is no longer the creation of an instance of DPP by means of node multiplicity aiming at ensuring the relative frequencies with which nodes in DPPr access shared resources, but rather the transformation of G into a multigraph, i.e., a graph in which multiple edges are allowed between two nodes. Ensuring the occurrence of the desired relative frequencies is left to depend upon the number of edges between each pair of nodes.

The multigraph we employ is denoted by $M = (N'', E'')$ with $N'' = N$ and, for $(n_i, n_j) \in E$, e_{ij} edges in E''. Just as with the SER mechanism of Section 3, here too we resort to orientations of the multigraph to indicate the nodes whose turn it is to access shared resources at each pulse. Unlike that mechanism, however, in this case the notion of an acyclic orientation of M is not meaningful, as it would require all edges between the same two nodes to be always oriented in the same direction, thereby making the very use of multiple edges meaningless. Acyclic orientations of G, however, will also in this case play an important role, as we discuss later.

The mechanism we employ in this section is such that it allows $n_i \in N$ to access shared resources whenever there are at least r_i edges oriented toward itself among the e_{ij} edges it shares with each neighbor n_j. Upon completion of its access to shared resources, the orientation changes by the reversal of r_i edges toward each of its neighbors. We refer to this mechanism as Scheduling by Multiple Edge Reversal (SMER). Like SER, SMER admits an extremely simple, fully distributed implementation employing message passing to indicate reversal.

The crux of SMER is of course the value of each e_{ij} and how to orient each of those edges so that conditions (i) and (ii) are satisfied. Some bounds on the value of e_{ij} are pretty obvious, as for example $e_{ij} \geq \max\{r_i, r_j\}$ (otherwise, neither n_i nor n_j will ever be in an independent set). Similarly, it must be that $e_{ij} \leq r_i + r_j - 1$ (so n_i and n_j can never be granted access to shared resources at the same pulse).

Let $\mu_0, \mu_1, \ldots$ be the sequence of orientations of M that SMER produces from the initial orientation μ_0. Naturally, our interest lies in sequences that are infinite, although at this point the conditions under which this can be ensured are still unknown. For $s \geq 0$, let a_s^{ij} denote the greatest multiple of g_{ij} that does not exceed the number of edges oriented from n_i to n_j by μ_s. The sum $a_s^{ij} + a_s^{ji}$ is constant for $s \geq 0$, because its two terms vary with s by transferring a certain multiple of g_{ij} from one to the other. We denote such a sum by f_{ij}. Every orientation of M for which the sum of such quantities is f_{ij} is called *legal for* (n_i, n_j).

Now consider the sub-multigraph of M, call it M^{ij}, induced by the pair of neighbors n_i and n_j. Also, let $\mu_0^{ij}, \mu_1^{ij}, \ldots$ denote the sequence of orientations of M^{ij} produced by SMER from μ_0^{ij}. We then have the following, letting G^{ij} be the subgraph of G induced by n_i and n_j.

Theorem 5. *If $r_i \geq r_j$ and $\max\{r_i, r_j\} \leq e_{ij} \leq r_i + r_j - 1$, then the use of SMER from μ_0^{ij} on M^{ij} solves the instance of DPPr given by G^{ij}, r_i and r_j if and only if $f_{ij} = r_i + r_j - g_{ij}$. In this case, the sequence $\mu_0^{ij}, \mu_1^{ij}, \ldots$ includes all orientations of M^{ij} that are legal for (n_i, n_j).*

Proof: If $f_{ij} = r_i + r_j - g_{ij}$, then

$$(a_0^{ij} - r_j) + (a_0^{ji} - r_i) = -g_{ij},$$

where the terms in parentheses are both multiples of g_{ij}, and then the equality can only hold if one of them is nonnegative. Straightforward induction shows that the sequence $\mu_0^{ij}, \mu_1^{ij}, \ldots$ is infinite, and then there has to exist $s \geq 0$ for which $a_s^{ji} \geq r_i$.

From μ_s^{ij}, SMER generates edge reversals in the following two patterns. In the first pattern, n_i reverts r_i edges toward n_j, which in turn reverts r_j edges toward n_i a number of pulses given by $\lceil r_i/r_j \rceil$ for a total of $r_i + r_j - r_i \bmod r_j$ edges. In the second pattern, n_i reverts r_i edges toward n_j, and then r_j edges are reverted back toward n_i for $\lfloor r_i/r_j \rfloor$ pulses, totaling $r_i - r_i \bmod r_j$ edges.

The net effect of the first pattern of reversals is to revert $r_j - r_i \bmod r_j$ edges toward n_i. For the second pattern, the net effect is to revert $r_i \bmod r_j$ edges in the direction of n_j. Because the number of distinct orientations is finite, a periodic repetition of orientations must at some point happen, and during this repetition the net effect of all patterns of one type must equal the net effect of all patterns of the other type. Also, all edge reversals occur in groups whose sizes are multiples of g_{ij}, so it follows that the first pattern occurs $r_i \bmod r_j/g_{ij}$ times and the second $(r_j - r_i \bmod r_j)/g_{ij}$ times.

Conditions (i) and (ii) are then satisfied, the former as a direct consequence of the value of e_{ij}, the latter by the numbers of times each of the two patterns of reversals occurs. In addition, the number of distinct legal orientations for (n_i, n_j) is $f_{ij}/g_{ij} + 1$. By (1), the number of orientations it takes for an orientation to be repeated is

$$\begin{aligned} p_{ij} &= \frac{r_i + r_j}{g_{ij}} \\ &= \frac{f_{ij} + g_{ij}}{g_{ij}} \\ &= \frac{f_{ij}}{g_{ij}} + 1, \end{aligned}$$

and then the sequence $\mu_0^{ij}, \mu_1^{ij}, \ldots$ includes all orientations that are legal for (n_i, n_j).

If $f_{ij} \neq r_i + r_j - g_{ij}$, then either $f_{ij} \geq r_i + r_j$, which leads to the contradiction of $f_{ij} > e_{ij}$, or $f_{ij} \leq r_i + r_j - 2g_{ij}$. In the latter case, and contrary to the theorem's assertion, suppose that the use of SMER from μ_0^{ij} does solve the instance of DPPr under consideration. By (1), the number of distinct orientations of M^{ij} that are legal for (n_i, n_j) must be at least p_{ij}, which is strictly greater than the maximum number of such orientations yielded by f_{ij}, that is,

$$\begin{aligned} p_{ij} &= \frac{r_i + r_j}{g_{ij}} \\ &> \frac{r_i + r_j - g_{ij}}{g_{ij}} \\ &= \frac{r_i + r_j - 2g_{ij}}{g_{ij}} + 1, \end{aligned}$$

a contradiction. ∎

It is a consequence of Theorem 5 that $e_{ij} = r_i + r_j - g_{ij}$ is the necessary and sufficient number of edges in M^{ij}. Fewer edges (even respecting the lower bound of $\max\{r_i, r_j\}$) render the solution impossible by leading to $f_{ij} < r_i + r_j - g_{ij}$, while edges in excess (though respecting the upper bound of $r_i + r_j - 1$) leave the value of f_{ij} unchanged and are therefore useless. We then henceforth make the assumption that $e_{ij} = r_i + r_j - g_{ij}$ and that μ_0 (and hence $\mu_1, \mu_2, \ldots$ as well) is legal for (n_i, n_j). It then follows that, for all $s \geq 0$, a_s^{ji} and a_s^{ij} are the actual numbers of edges oriented toward n_i and n_j, respectively, between n_i and n_j.

We now set out to look for conditions under which the assertion of Theorem 5 can be generalized to hold for M as a whole. Our goal is to be able to display orientations of M such that the use of SMER from one such orientation solves the instance of DPPr given by G and $r_1, \ldots, r_n$.

Our first step is to consider the set K of undirected simple cycles in G. For $\kappa \in$ K, let $\ell(\kappa)$ be the number of nodes that constitute κ, and let $n_{i_1}, \ldots, n_{i_{\ell(\kappa)}}$ be those nodes in the order they appear in κ for arbitrary i_1 and arbitrary traversal direction (this notation does not mention κ explicitly, but lets it be clear from the context instead). Also, we use $(n_i, n_j) \in \kappa$ to denote that edge (n_i, n_j) is one of the edges of κ. We also use $(n_i \to n_j) \in \kappa$ to indicate this and that, in addition, κ is to be traversed, beginning at n_i, from n_i to n_j.

Now consider, for $s \geq 0$, the definition

$$\rho_s^{i_1 i_{\ell(\kappa)}}(\kappa) = \left\lfloor \frac{a_s^{i_1 i_{\ell(\kappa)}}}{r_{i_{\ell(\kappa)}}} \right\rfloor \tag{6}$$

and the recursive definition

$$\sigma_s^{i_1 i_z}(\kappa) = \begin{cases} \left\lfloor \dfrac{a_s^{i_{\ell(\kappa)} i_1}}{r_{i_1}} \right\rfloor, & \text{if } z = 1; \\ \left\lfloor \dfrac{\sigma_s^{i_1 i_{z-1}}(\kappa) r_{i_{z-1}} + a_s^{i_{z-1} i_z}}{r_{i_z}} \right\rfloor, & \text{if } 1 < z \leq \ell(\kappa). \end{cases} \tag{7}$$

Clearly, for $(n_i, n_j) \in \kappa$, $\rho_s^{ij}(\kappa) > 0$ if and only if $\rho_s^{ji}(\kappa) = 0$. Similarly, $\sigma_s^{ij}(\kappa) \neq \sigma_s^{ji}(\kappa)$ is a possibility.

An illustration is given in Figure 2 of how to compute the quantities in (6) and (7) for a simple cycle κ. In the figure, κ is the simple cycle depicted, with $\ell(\kappa) = 8$. What is illustrated are the paths taken on the simple cycle to compute, for some $s \geq 0$ and arbitrary (n_i, n_j) on the cycle, the pair $\rho_s^{ij}(\kappa)$ and $\sigma_s^{ij}(\kappa)$ and the pair $\rho_s^{ji}(\kappa)$ and $\sigma_s^{ji}(\kappa)$.

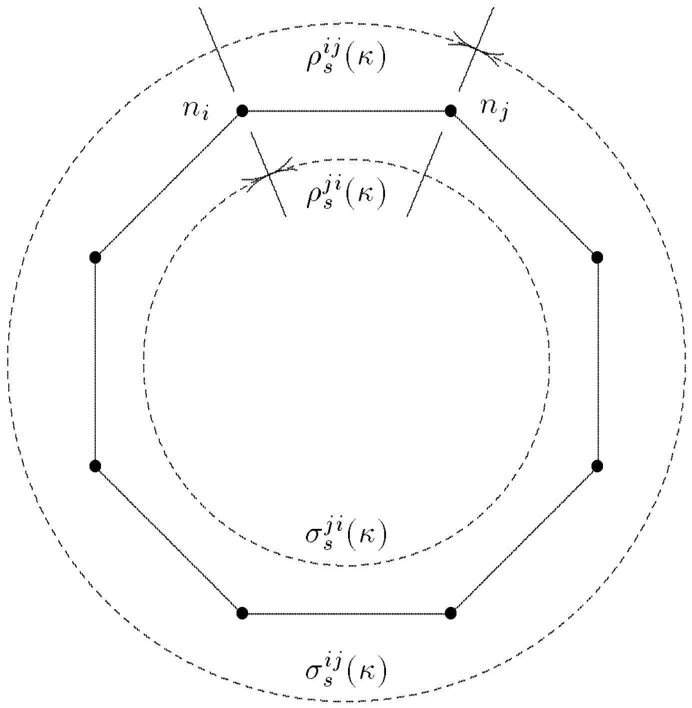

Figure 2. Computation of ρ's and σ's

These are paths from n_i to n_j, in the former case, from n_j to n_i in the latter. The ρ's are computed on one-hop paths, and the σ's on seven-hop paths.

The intuition behind these definitions is that $\rho_s^{i_1 i_{\ell(\kappa)}}(\kappa)$ and $\sigma_s^{i_1 i_{\ell(\kappa)}}(\kappa)$ indicate, respectively, the number of times $n_{i_{\ell(\kappa)}}$ would be able to access the shared resources if this only depended on the edge $(n_{i_1}, n_{i_{\ell(\kappa)}})$, and this same number if it depended exclusively on the $(\ell(\kappa)-1)$-edge path between n_{i_1} and $n_{i_{\ell(\kappa)}}$ on κ. It is on the interplay of these two quantities that the correctness of SMER depends, as we see following a few additional definitions.

For $s \geq 0$, every orientation μ_s of M induces an orientation of G as follows. For $(n_i, n_j) \in E$, we say that (n_i, n_j) is oriented toward n_j if and only if $a_s^{ij} \geq r_j$. We let $\omega(\mu_s)$ denote such an orientation. Sinks according to $\omega(\mu_s)$ are nodes that may access shared resources, just as in the case of SER for DPP as we discussed in Section 3. Unlike SER, however, under SMER it does not necessarily happen that a sink becomes a source in the next pulse, nor is it guaranteed that the acyclicity of $\omega(\mu_s)$ leads to an acyclic $\omega(\mu_{s+1})$ as well. Ultimately, what we seek is precisely a set of conditions under which such a guarantee exists.

Lemma 6. *For all $s \geq 0$, all $\kappa \in \mathrm{K}$, and all $(n_i \to n_j) \in \kappa$, $\sigma_{s+1}^{ij}(\kappa) \geq \rho_{s+1}^{ij}(\kappa)$ if and only if $\sigma_s^{ij}(\kappa) \geq \rho_s^{ij}(\kappa)$.*

Proof: By (6) and (7), first note that $\rho^{ij}_{s+1}(\kappa) = \rho^{ij}_s(\kappa)$ and $\sigma^{ij}_{s+1}(\kappa) = \sigma^{ij}_s(\kappa)$, unless n_j is a sink in $\omega(\mu_s)$. The former equality is immediate, and the latter follows from the fact that, if n_j is not a sink in $\omega(\mu_s)$, then any other node in κ that is a sink in $\omega(\mu_s)$, say n_z for $1 \leq z < \ell(\kappa)$ with $i = i_1$ and $j = i_{\ell(\kappa)}$, is such that

$$
\begin{aligned}
\sigma^{i_1 i_{z+1}}_{s+1}(\kappa) &= \left\lfloor \frac{\sigma^{i_1 i_z}_{s+1}(\kappa) r_{i_z} + a^{i_z i_{z+1}}_{s+1}}{r_{i_{z+1}}} \right\rfloor \\
&= \left\lfloor \frac{\sigma^{i_1 i_z}_{s}(\kappa) r_{i_z} - r_{i_z} + a^{i_z i_{z+1}}_{s} + r_{i_z}}{r_{i_{z+1}}} \right\rfloor \\
&= \sigma^{i_1 i_{z+1}}_s(\kappa).
\end{aligned}
$$

We then concentrate on the case in which n_j is a sink in $\omega(\mu_s)$. In this case,

$$
\begin{aligned}
\rho^{ij}_{s+1}(\kappa) &= \left\lfloor \frac{a^{ij}_{s+1}}{r_j} \right\rfloor \\
&= \left\lfloor \frac{a^{ij}_s - r_j}{r_j} \right\rfloor \\
&= \left\lfloor \frac{a^{ij}_s}{r_j} - 1 \right\rfloor \\
&= \rho^{ij}_s(\kappa) - 1
\end{aligned}
$$

and, letting n_k be the node that precedes n_j in the traversal of κ from n_i,

$$
\begin{aligned}
\sigma^{ij}_{s+1}(\kappa) &= \left\lfloor \frac{\sigma^{ik}_{s+1}(\kappa) r_k + a^{kj}_{s+1}}{r_j} \right\rfloor \\
&= \left\lfloor \frac{\sigma^{ik}_{s+1}(\kappa) r_k + a^{kj}_{s} - r_j}{r_j} \right\rfloor \\
&= \left\lfloor \frac{\sigma^{ik}_{s+1}(\kappa) r_k + a^{kj}_{s}}{r_j} - 1 \right\rfloor.
\end{aligned}
$$

If $\sigma^{ij}_s(\kappa) \geq \rho^{ij}_s(\kappa)$, then it follows that

$$
\begin{aligned}
\frac{\sigma^{ik}_s(\kappa) r_k + a^{kj}_s}{r_j} &\geq \left\lfloor \frac{\sigma^{ik}_s(\kappa) r_k + a^{kj}_s}{r_j} \right\rfloor \\
&\geq \left\lfloor \frac{a^{ij}_s}{r_j} \right\rfloor,
\end{aligned}
$$

and then

$$
\sigma^{ik}_s(\kappa) \geq \frac{\lfloor a^{ij}_s / r_j \rfloor r_j - a^{kj}_s}{r_k}.
$$

Because n_j is a sink in $\omega(\mu_s)$, $a_{s+1}^{ji} = a_s^{ji} + r_j$, and then by (7) we have $\sigma_{s+1}^{ik}(\kappa) \geq \sigma_s^{ik}(\kappa)$, which leads to

$$\begin{aligned}
\sigma_{s+1}^{ij}(\kappa) &\geq \left\lfloor \frac{\sigma_s^{ik}(\kappa) r_k + a_s^{kj}}{r_j} - 1 \right\rfloor \\
&\geq \left\lfloor \frac{\lfloor a_s^{ij}/r_j \rfloor r_j}{r_j} - 1 \right\rfloor \\
&= \rho_s^{ij}(\kappa) - 1 \\
&= \rho_{s+1}^{ij}(\kappa).
\end{aligned}$$

Conversely, if $\sigma_s^{ij}(\kappa) < \rho_s^{ij}(\kappa)$, then

$$\frac{\sigma_s^{ik}(\kappa) r_k + a_s^{kj}}{r_j} < \frac{a_s^{ij}}{r_j},$$

and consequently

$$\sigma_s^{ik}(\kappa) < \frac{a_s^{ij} - a_s^{kj}}{r_k}.$$

Although n_j is a sink in $\omega(\mu_s)$, the fact that $\sigma_s^{ij}(\kappa) < \rho_s^{ij}(\kappa)$ implies that s is not the last pulse at which n_j is a sink before n_i becomes a sink. As a consequence, even though $a_{s+1}^{ji} = a_s^{ji} + r_j$, (7) implies that $\sigma_{s+1}^{ik}(\kappa) = \sigma_s^{ik}(\kappa)$, and then

$$\begin{aligned}
\sigma_{s+1}^{ij}(\kappa) &= \left\lfloor \frac{\sigma_s^{ik}(\kappa) r_k + a_s^{kj}}{r_j} - 1 \right\rfloor \\
&< \left\lfloor \frac{a_s^{ij}}{r_j} - 1 \right\rfloor \\
&= \rho_s^{ij}(\kappa) - 1 \\
&= \rho_{s+1}^{ij}(\kappa).
\end{aligned}$$

■

Lemma 6 is of fundamental importance in our next theorem, which is the counterpart of Theorem 5 for the entirety of M. Before presenting that theorem, however, let us pause briefly to discuss the significance of Lemma 6 in a little more detail by means of two observations.

First suppose that for some $s \geq 0$ orientation μ_s of M is such that $\omega(\mu_s)$ contains a directed cycle. It must then contain a directed simple cycle, of which we let κ be the underlying undirected simple cycle in G. Now suppose $(n_i, n_j) \in \kappa$ is oriented by $\omega(\mu_s)$ from n_i to n_j. By (6), $\rho_s^{ij}(\kappa) > 0$, and by (7) we have $\sigma_s^{ij}(\kappa) = 0$. In this case, Lemma 6 implies that every $s' \geq 0$ must be such that $\sigma_{s'}^{ij}(\kappa) < \rho_{s'}^{ij}(\kappa)$, not just $s' = s$.

For the second observation, let us consider the following orientation $\bar{\mu}_0$ of M, i.e., an initial assignment of directions to the edges in the multigraph. For all $(n_i, n_j) \in E$, $\bar{\mu}_0$

orients exactly r_i edges in the direction of n_i if and only if $r_i \geq r_j$. If G happens to contain an undirected simple cycle κ such that $r_i = r_j$ for all $(n_i, n_j) \in \kappa$, then in addition $\bar{\mu}_0$ must be such that no directed cycle is created in $\omega(\bar{\mu}_0)$.

By (6), clearly $\rho_0^{ij}(\kappa) \leq 1$ for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$. By (7), and by construction of $\bar{\mu}_0$, it must be that $\sigma_0^{ij}(\kappa) \geq 1$, also for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$. To see the latter inequality more clearly, let n_k be the other neighbor of n_j on κ and n_l be the node on κ such that, when traversed beginning at n_j and going through n_k first, the sequence $r_j, r_k, \ldots, r_l$ is maximal nondecreasing. Clearly, by (7) it must hold that $\sigma_0^{lj}(\kappa) \geq 1$, thence $\sigma_0^{ij}(\kappa) \geq 1$ as well. What this second observation amounts to is that, for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$, $\sigma_0^{ij}(\kappa) \geq \rho_0^{ij}(\kappa)$.

Theorem 7. *The use of SMER from μ_0 solves the instance of DPPr given by G and $r_1, \ldots, r_n$ if and only if $\sigma_0^{ij}(\kappa) \geq \rho_0^{ij}(\kappa)$ for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$.*

Proof: If $\sigma_0^{ij}(\kappa) \geq \rho_0^{ij}(\kappa)$ for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$, then suppose, contrary to the theorem's assertion, that the instance of DPPr given by G and $r_1, \ldots, r_n$ is not solved by SMER. By Theorem 5, there has to exist a node that stops becoming sink at some pulse s, and from this node a directed path can be traced on G oriented by $\omega(\mu_s)$ along the directions of the edges until a directed simple cycle is found. As we remarked before, this directed cycle, whose underlying undirected simple cycle we let be κ, implies that $\sigma_s^{ij}(\kappa) < \rho_s^{ij}(\kappa)$ for all $(n_i \to n_j) \in \kappa$ such that (n_i, n_j) is oriented by $\omega(\mu_s)$ from n_i to n_j. By Lemma 6, $\sigma_0^{ij}(\kappa) < \rho_0^{ij}(\kappa)$, a contradiction.

Conversely, if there exists $\kappa \in \mathrm{K}$ and $(n_i \to n_j) \in \kappa$ such that $\sigma_0^{ij}(\kappa) < \rho_0^{ij}(\kappa)$, then by Lemma 6 the same holds for all $s \geq 0$, that is, $\sigma_s^{ij}(\kappa) < \rho_s^{ij}(\kappa)$, meaning that every node on κ (and consequently every node in G) eventually reaches a pulse from which it no longer becomes a sink. The instance of DPPr given by G and $r_1, \ldots, r_n$ is then not solved by the use of SMER starting at μ_0. ■

By Theorem 7, the orientation $\bar{\mu}_0$ introduced earlier is a proper starting point for the solution of DPPr by SMER.

To finalize, let us now return to the question raised in Section 4 on the conditions for ω'_0 to be acyclic. As we recall, the acyclicity of this orientation of G' depends, for all $(n_i, n_j) \in E$ such that $r_i \geq r_j$, on how to select, during step 2 of the construction of G', node groups of size $\lfloor q_j/q_i \rfloor$ or $\lceil q_j/q_i \rceil$. The answer follows from Theorem 7, as we discuss next. First orient M by some orientation μ_0 such that $\sigma_0^{ij}(\kappa) \geq \rho_0^{ij}(\kappa)$ for all $\kappa \in \mathrm{K}$ and all $(n_i \to n_j) \in \kappa$. Then, for all $(n_i, n_j) \in E$ such that $r_i \geq r_j$, perform step 2 so that the two possibilities for group sizes appear following the same pattern according to which n_i and n_j become sinks from μ_0. The resulting ω'_0 is guaranteed to be acyclic because, by construction of ω_0' and by Lemma 1 applied to each of the cycles obtained in step 2 of that construction, a directed cycle in $\omega(\mu_0)$ would also have to exist otherwise, which is precluded by Theorem 7.

6. Fully asynchronous operation

The fully synchronous model of distributed computation we have been employing so far is very useful because its restricted timing properties allow us to refer unambiguously to global time-related properties, as for example the existence of sequences of orientations. In this section we argue, very briefly, that all our solutions to both DPP and DPPr are still correct if we no longer assume full synchronism.

A model of timing characteristics during a distributed computation more general than the fully synchronous model is the fully asynchronous model. In this model, nodes are driven by independent, local time bases, and messages sent between neighbors, although guaranteed to be delivered in finite time, may be subject to unpredictable delays before delivery occurs.

All of our solutions are based on the notion of edge reversal, either full reversal on graphs, or partial reversal on multigraphs. In either case, a reversal that depends on the arrival of $x > 0$ reversal messages is only triggered by the arrival of the last such message, and therefore the order in which the previous messages arrived is totally immaterial. The effect of asynchronism on our solutions is then irrelevant as far as their correctness is concerned.

7. Concluding remarks

In this paper we have introduced a new resource sharing paradigm, DPPr, as an extension of DPP under heavy loads. Our extension has the same semantics as heavy-load DPP, and in addition requires nodes to access shared resources according to rates specified previously relatively to one another.

We have described two solutions to DPPr and demonstrated their correctness. The first solution is based on a reduction of DPPr to heavy-load DPP by increasing the multiplicity of nodes so to achieve the desired relative frequencies of access to shared resources. The second solution, by contrast, employs edge multiplicity to this end. Both solutions admit fully distributed, asynchronous implementations.

One question that we have not addressed in the paper, although pertinent, is the question of efficiently deciding, in the context of Section 5, whether an orientation μ_0 of M meets the necessary and sufficient condition of Theorem 7. In the same vein, it may also be legitimate to inquire on how such an orientation would relate, concurrency-wise (cf. Theorem 4), to the orientation $\bar{\mu}_0$ and the ones that descend from it under SMER.

Acknowledgments

The authors have received partial support from the Brazilian agencies CNPq and CAPES, and wish to thank Carlos Ribeiro and Antonio Setubal for discussions in the early stages of the work.

References

1. Barbosa, V. C. (1986), "Concurrency in Systems with Neighborhood Constraints," Ph.D. Dissertation, Computer Science Department, University of California, Los Angeles.

2. Barbosa, V. C. and Gafni, E. (1989), Concurrency in heavily loaded neighborhood-constrained systems, *ACM Trans. on Programming Languages and Systems* **11**, 562–584.

3. Chandy, K. M. and Misra, J. (1984), The drinking philosophers problem, *ACM Trans. on Programming Languages and Systems* **6**, 632–646.

4. Dijkstra, E. W. (1968), Cooperating sequential processes, *in* "Programming Languages" (F. Genuys, Ed.), pp. 43–112, Academic Press, New York.

5. França, F. M. G. (1993), Scheduling weightless systems with self-timed Boolean networks, *in* "Proceedings, Weightless Neural Networks Workshop," pp. 87–92.

6. França, F. M. G. (1994), "Neural Networks as Neighbourhood-Constrained Systems," Ph.D. Thesis, Imperial College London.

7. França, F. M. G., Alves, V. C., and Granja, E. P. (1995), Asynchronous digital circuits as neighbourhood-constrained systems, Technical Report ES-357/95, COPPE/UFRJ, Rio de Janeiro.