Rev. Econ. Design 3, 93–127 (1998)

Review of Economic Design

The scheduling and organization of periodic associative computation: Efficient networks

Timothy Van Zandt*

Department of Economics, Princeton University, Princeton, NJ 08544-1021, USA
(e-mail: tvz@princeton.edu)

Received: 15 October 1994 / Accepted: 6 March 1997

Abstract. This paper characterizes the efficient decentralized networks for calculating the associative aggregate of cohorts of data of a fixed size that arrive periodically. Radner (1993) proposed this problem of periodic parallel associative computation as a model of the ongoing information processing and communication by the administrative staff of a large organization. For a simpler model in which the organization processes a single cohort of data – which is equivalent to the periodic model when the agents are paid only when busy – he found that the efficient networks are hierarchical but quite irregular, even though the computation problem and technology are each symmetric. In the periodic model in which managers are paid even when idle, it becomes important to minimize idle time when scheduling managers to processing tasks. Such scheduling appears more difficult when each problem is processed by an irregular hierarchy, which suggest that hierarchies might be more regular in the periodic model. However, we show that in a class of efficient networks for periodic computation that spans the efficiency frontier, the processing of each cohort is similar to the efficient processing of a single cohort, and the overall organizational structure is not even hierarchical.

JEL classification: D83, D23

Key words: Organizations, decentralization, parallel processing

* This is part 2 of a revision of a paper entitled "Periodic Parallel Addition," which I wrote while I was a Post-Doctoral MTS at AT&T Bell Laboratories (1988–1989). The support of AT&T Bell Laboratories is gratefully acknowledged. This revision has been supported in part by grant SBR-9223971 from the National Science Foundation and a CORE Research Fellowship. I benefited from discussions with Roy Radner and the extensive comments of two anonymous referees.

1 Motivation

The agents in organizations such as firms jointly process information about their environment that is needed for decision-making. This is called decentralized information processing, and it is analogous to parallel processing within a multiprocessor computer and to distributed processing in a network of computers. By studying optimal decentralized networks for various computation and decision problems, we obtain insight into how information processing activities affect the structure, performance and returns to scale of organizations.[1]

A simple but prevalent and important example of the information processing tasks in organizations is associative computation. The characterization of decentralized associative computation as a means to understand information processing in organizations was proposed and developed by Radner (1993), who considered two models. The *one-shot* model considers the aggregation of a single list of data, without considering possible interaction with other computation tasks. The *periodic* (or systolic) model considers the aggregation of each of multiple independent cohorts of data, which arrive at fixed intervals and are of the same size. This model is meant to capture the ongoing nature of computation in organizations.[2] Radner (1993) characterized a class of efficient networks for the one-shot model, but not for the periodic model. The purpose of this paper is to provide a full characterization of a set of efficient periodic networks that can attain any efficient performance for the periodic model (but with efficiency defined with respect to a slightly restricted class of networks that satisfy a weak stationarity condition).

The appropriate measure of managerial resources in the one-shot model is the number of *operations*.[3] In the periodic model or in any model in which the organization must handle multiple information processing problems, if managers were paid on an hourly basis and hence only for the number of operations performed (and if the organization can use a distinct network to process each problem, as is allowed in Radner's periodic model), then the organization should

[1] Such an exercise presumes that administrators have limited computational ability, and hence are boundedly rational. This explicitly-modeled bounded rationality determines the feasible set of organizations. The process by which one of the feasible organizations is selected or comes into existence is not modeled. The reasons for characterizing organizations that are constrained-optimal within this feasible set are the same as for studying optimal behavior in other economic models, and are not based on a presumption that there is an unboundedly rational organization designer. Instead, (i) this characterization helps delimit the set of feasible organizations and thereby understand the effect of information processing constraints, (ii) efficient organizations are a benchmark that may approximate actual organizations that have operated in a stationary environment for a long time (in particular, on a longer time scale than the daily computational activities that are explicitly modeled), and (iii) efficient organizations are a natural objective for the economist as organization designer. See Van Zandt (1997a) for further discussion.

[2] The ongoing nature of computation is captured more realistically in real-time processing models. See for example, Van Zandt (1997a,b).

[3] See Meagher and Van Zandt (1997) for a discussion of managerial costs in the one-shot model. Radner (1993) and Keren and Levhari (1979, 1983) measured resources by the number of managers, but, as shown in Meagher and Van Zandt (1997), their results on efficient hierarchies also hold when resources are measured by the number of operations. For this reason, their papers are treated here as if they measured managerial costs in the one-shot model by the number of operations, as we do.

use a separate efficient one-shot network for each problem. However, here, as in Radner (1993), we measure managerial costs in the periodic model by the number of managers in the organization. This presumes that the organization must have a fixed number of managers who are paid salaries even when idle.[4] Therefore, the actual distinction between the one-shot and the periodic models is that in the former, managers are paid an hourly wage only for the time they are busy, and in the latter managers are paid salaries whether busy or idle. Comparing these models thus allows one to study how the need to schedule managers to information processing activities in order to reduce idle time affects the structure and costs of decentralized associative computation.

The organizational structure that we infer from a decentralized information processing network is the graph that depicts the flow of information between managers. This graph is naturally a hierarchy (tree) in the efficient one-shot networks (see Radner 1993; Van Zandt 1997c; Meagher and Van Zandt 1997). Whereas models of hierarchies such as in Keren and Levhari (1979, 1983, 1989) restricted hierarchies to be completely balanced – meaning that the distance from each leaf to the root is the same and all managers in the same tier have the same number of subordinates – Radner (1993) found that the efficient one-shot hierarchies are highly irregular. This is not counterfactual, but it is of interest to know whether the observed irregularities arise because of asymmetries among managers, data or computation problems, or whether such irregularities can arise even with homogeneous managers, data and operations. Furthermore, given that balanced hierarchies have simple structures that make them convenient for modeling organizations, it is useful to know under what conditions this regularity property arises endogenously.

Therefore, one question raised of the periodic model is whether it leads to more regular hierarchies than the one-shot model. One reason for making such a conjecture is that, in the periodic model, it appears more difficult to schedule managers to tasks without idle time when the hierarchies that process the individual cohorts are irregular, because in irregular hierarchies managers are busy for different lengths of time. In contrast, suppose that every cohort is processed by a network that is completely uniform, i.e., in which every manager has the same number of subordinates and hence is busy for the same amount of time. If the time that each manager is busy is equal to the time between problems, then such a hierarchy has no idle time. Because the class of uniform hierarchies is too limited to allow a trade-off between delay and managerial costs, Radner (1993) proposed a slight generalization, called "preprocessing/overhead" (PPO) trees; these hierarchies are also fairly regular and have little or no idle time. Radner (1993) found that managerial costs in the PPO trees were not too far from a theoretical lower bound, but he did not show that these networks are efficient, and these networks can only attain a limited range of performance values.

[4] The managers should be thought of as roles, which may be occupied by different managers at different times. The assumption is that the turnover of managers is slow and managers cannot be quickly fired when finished with one and rehired when needed for a new task.

One of the main themes of this paper is that, in contrast to the conjecture above, the processing of each cohort in efficient network can be very similar to that of the efficient one-shot networks. *This is because idle time can be reduced without equalizing the workloads of the managers. Instead, it suffices to adjust the workloads so that they are multiples of the amount of time between cohorts.*

In the efficient periodic networks, the workloads are of different lengths, and the teams that process the cohorts shift over time. Hence a manager may not report to the same superior or receive messages from the same subordinates during the processing of each cohort, and the overall organizational structure is neither hierarchical nor even a forest of trees. However, we note that this shifting of communication channels is costly if communication costs depend on the number of links in the network instead of on just the number of messages exchanged, and suggest that modifying the communication costs in this way may change the structure of efficient networks, perhaps towards more regularity.

Associative computation is just one of the activities of the administrative staff of organizations, and the value of this literature is not primarily specific predictions or recommendations about the shape of hierarchies. Instead, this literature has been a vehicle for exploring many issues in information processing in organizations that will also arise in other information processing tasks and models, such as communication costs, the trade-off between processing costs and delay, and the distinctions between one-shot versus repeated computation, salaried versus non-salaried managers, and stationary versus non-stationary networks. All of these themes are explored here.

2 Model

This paper draws extensively on the definition of the model and the preliminary results contained in a companion paper (Van Zandt 1997c). The current paper is not intended to be self-contained, and the companion paper will not be cited each time notation, definitions and results from that paper are used.

As in Van Zandt (1997c), there is a fixed set $\mathcal{N} = \{1, \ldots, N\}$ of data sources of finite size $N \geq 2$ and a set $\mathcal{T} \subset \mathbb{N}$ of arrival times. At the beginning of each cycle $\tau \in \mathcal{T}$, an organization receives a cohort $\{X_{1\tau}, \ldots, X_{N\tau}\}$ of data containing one item from each data source, and must calculate $X_{1\tau} \oplus \cdots \oplus X_{N\tau}$ for an associative and commutative operation $\oplus$. The cohort that arrives in cycle τ is called cohort τ.

In this paper, we consider two models. In the *one-shot* model, a single cohort arrives in cycle 0, so that $\mathcal{T} = \{0\}$ (and the cohort subscript will be often suppressed). In the *periodic* model, a new cohort arrives every T cycles, starting in cycle 0, so that $\mathcal{T} = \{\tau \in \mathbb{N} | \tau \bmod T = 0\}$.

The cohorts are processed by a network of managers, according to the computation model in Radner (1993). The current paper is based on the more formal statement of the capabilities of individual managers and the definition of a network $\langle \mathcal{M}, \mathcal{I} \rangle$ given in Van Zandt (1997c). The model does not account for costs

of memory (data storage) and of the transmission of messages. Hence, the only computation or managerial costs are the managerial wages. We say that a manager in a network $\langle \mathcal{M}, \mathcal{I} \rangle$ is *busy* in cycle t if she performs an operation; otherwise, she is *idle*. As discussed in the introduction and in Meagher and Van Zandt (1997), in the one-shot model managers are paid only when busy, whereas in the periodic model we presume that the organization has a fixed size and managers are paid even when idle.

Definition 2.1 *The* **size** *of a network $\langle \mathcal{M}, \mathcal{I} \rangle$ is the number M of managers in $\mathcal{M}$. The* **managerial costs** *of a periodic network $\langle \mathcal{M}, \mathcal{I} \rangle$ are equal to M. The* **managerial costs** *of a one-shot network $\langle \mathcal{M}, \mathcal{I} \rangle$ are equal to the number W of operations in $\mathcal{I}$.*

A network is functional if the aggregate of each cohort is sent to the output device. If a network is functional, the delay for each cohort is denoted $D(\tau)$, and the function $D : \mathcal{T} \to \mathbb{N}$ is called the delay of the network. Dominance, weak dominance and efficiency (within the class of functional networks) are defined in the standard way with respect to delay and managerial costs (for each, less is better).

As stated in Van Zandt (1997c, Sect. 5), every functional network is weakly dominated by an essential network, which is a functional network such that it is not possible to eliminate a manager or instruction from the network and still obtain a functional network. Hence, a network is functional if it is essential, and it is efficient if it is also not dominated by any essential network. These facts are very useful to us because of the characterization of essential networks given in Corollary 5.1 in Van Zandt (1997c). According to this result, each manager's instructions in an essential network can be partitioned into *tasks*, which is a set $\mathcal{H}$ of instructions performed consecutively by the same manager that begins with a LOAD, followed by zero or more ADD's, and then concludes with a message to a manager or to the output device.

Definition 2.2 *Let $\langle \mathcal{M}, \mathcal{I} \rangle$ be a network and let $\mathcal{H} \subset \mathcal{I}$ be a task. If the LOAD in $\mathcal{H}$ is executed in cycle t_1 and its message is executed in cycle t_2, then we say that $\mathcal{H}$ is* **active** *in cycles $t_1, \ldots, t_2 - 1$, and the* **duration** *of $\mathcal{H}$ is $t_2 - t_1$. The manager who executes the instructions in $\mathcal{H}$ is said to* **be assigned** *or to* **perform** *task $\mathcal{H}$.*

Remark 2.1 Let $\langle \mathcal{M}, \mathcal{I} \rangle$ be an essential network. Note that the manager who performs a task $\mathcal{H} \subset \mathcal{I}$ cannot execute any other instructions while $\mathcal{H}$ is active. As long as we respect this restriction, we can construct a new network by reassigning one or more tasks to different managers. ("Reassigning" a task $\mathcal{H}$ to a manager m means changing the manager who executes the instructions in $\mathcal{H}$ to m.) The resulting network is essential and has the same delay and the same number of operations in any cycle as does the original network.

3 Maximum-slack one-shot networks

The purpose of this section is to characterize a class of one-shot networks, called MS networks. The main motivation for this exercise is that the MS networks will be the building blocks for efficient periodic networks in Sect. 4. Theorem 3.1 is also of independent interest as a characterization of efficient information processing in the one-shot model, because we show that any one-shot network is weakly dominated by an MS network.

Henceforth, we need to distinguish more frequently between one-shot and periodic networks. Therefore, we denote a one-shot network by $\langle \mathcal{Q}, \mathcal{J} \rangle$, we let Q be the number of managers in $\mathcal{Q}$, and we index managers in $\mathcal{Q}$ by q. The symbols $\langle \mathcal{M}, \mathcal{I} \rangle$, M and m are reserved for periodic networks. Furthermore, we denote the delay $D(0)$ of a one-shot network simply by D. Recall that the measure of managerial costs for one-shot networks is the number W of operations.

In the rest of this section, a network (without a qualifier) means an *essential one-shot network*.

We begin by defining some properties for networks and stating the main result:

Definition 3.1 *For each manager $q \in \mathcal{Q}$ in a network $\langle \mathcal{Q}, \mathcal{J} \rangle$, let $b(q)$ be the first and let $c(q)$ be the last cycle in which manager q performs an instruction in $\mathcal{J}$.*[5] *Manager q is said to* **begin** *in cycle $b(q)$ and to* **finish** *in cycle $c(q)$. The total number of cycles managers are idle before they begin, $\sum_{q \in \mathcal{Q}} b(q)$, is called the* **slack** *of $\langle \mathcal{Q}, \mathcal{J} \rangle$.*

Definition 3.2 *Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a network.*

P0. $\langle \mathcal{Q}, \mathcal{J} \rangle$ *is a* **maximum-slack** *network if and only if it has as much slack as any other network with the same size and delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$.*

P1. $\langle \mathcal{Q}, \mathcal{J} \rangle$ *is* **simple** *if and only if each manager performs one task.*

P2. $\langle \mathcal{Q}, \mathcal{J} \rangle$ *is* **continuous** *if and only if, for each manager $q \in \mathcal{Q}$ and each cycle t such that $b(q) \leq t < c(q)$, manager q performs an operation in cycle t.*

P3. $\langle \mathcal{Q}, \mathcal{J} \rangle$ *is* **just-in-time** *if and only if each SEND is executed in the same cycle in which it is processed.*

P4. $\langle \mathcal{Q}, \mathcal{J} \rangle$ **has preprocessing first** *if and only if there are no preprocessing operations after the first postprocessing operation.*

P5. $\langle \mathcal{Q}, \mathcal{J} \rangle$ *is* **strongly overlapping** *if and only if each manager performs at least 2 operations before any other manager finishes.*

Definition 3.3 *A network $\langle \mathcal{Q}, \mathcal{J} \rangle$ is called an* **MS** *network if and only if it is strongly overlapping and has maximum slack.*

Theorem 3.1 *A network $\langle \mathcal{Q}, \mathcal{J} \rangle$ is an MS network if and only if it has Properties P1–P5.*

Proof. See Appendix A. □

[5] Note that the first and last instructions are a LOAD and a message, respectively.

In the rest of this section, we will illustrate the properties defined in Definition 3.2 and discuss the intuition behind and significance of Theorem 3.1. We defer until Sect. 4 (see especially Remark 4.3) the explanation of why MS networks are useful for constructing periodic networks (in brief, this is because slack allows the workloads of managers to be adjusted in order to reduce idle time).

A message of this paper is that the MS networks that process each cohort in the efficient periodic networks are similar to efficient one-shot networks. In particular, it is *not* true that the hierarchies that process each cohort are more regular in the periodic model than in the one-shot model because of the cost of idle time. The following is an indication of the similarity between MS networks and efficient one-shot networks, and of the irregularity of MS networks:

Proposition 3.1 (Meagher and Van Zandt 1997)

1. *An efficient network is simple and strongly overlapping.*
2. *A simple network is hierarchical.*
3. *In a simple and strongly overlapping network, each manager processes at least two raw data.*

Thus, both MS networks and efficient networks are simple and hierarchical, but also irregular, because each manager processes raw data and hence has immediate subordinates that are data sources. We now briefly examine why Proposition 3.1 holds, as this will also help us understand Theorem 3.1.

That efficient networks are simple and hence hierarchical is illustrated in Fig. 1. The network on the left is not simple and is not hierarchical. Manager q_1 sends a message to q_2 at the end of her first task and then processes a message from q_2 during her second task. The simple network on the right is obtained by concatenating q_1's tasks. This eliminates a message and hence a postprocessing operation.

Note that an equivalent definition of a simple network (given the restriction that networks be essential) is that each manager sends one message. Since each message except for the OUTPUT is processed, we obtain the following proposition:

Proposition 3.2 (Meagher and Van Zandt 1997) *If a network $\langle \mathcal{Q}, \mathcal{J} \rangle$ is simple, it has $N+Q-1$ operations; otherwise, it has more than $N+Q-1$ operations.*

This means that the managerial costs of simple networks can be parameterized by the size of the networks.

The balanced hierarchies in Keren and Levhari (1979) are simple but are not strongly overlapping. This is because managers are arranged in tiers and the managers in each tier do not begin processing until all managers in the subordinate tier have finished. The messages from the subordinate tier are then distributed evenly among the managers in the current tier, who aggregate these messages and send their partial results to the next tier. The computation begins when the raw data are processed by the managers in the lowest tier.

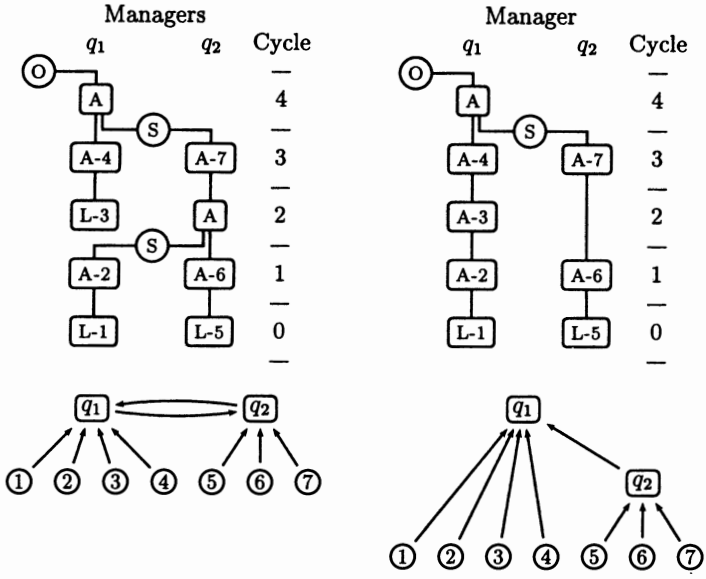


Fig. 1. Two networks for $N = 7$. Both networks have 2 managers, but the l.h.s. has 8 operations whereas the r.h.s. has 7. The l.h.s. is not simple, because manager q_1 has two tasks. The l.h.s. is also not hierarchical. The r.h.s. is simple and hierarchical.

Radner (1993) showed that such balanced hierarchies, although appealing because of their regular structure, are typically not efficient. They can be "reduced" by combining the operations of a manager q_2 in one tier with the operations of a manager q_1 in the subordinate tier, because the operations of the two managers are not concurrent. This reduction is similar to the concatenation of two tasks performed by the same manager in a network that is not simple, which was illustrated in Fig. 1. Meagher and Van Zandt (1997) modifies this reduction, by transferring one of q_1's operations to the manager who processes q_1's message, so that the reduction also works when q_1 and q_2 perform one operation concurrently. This is illustrated in Fig. 2. Observe that the reduction not only eliminates a manager, but it also eliminates one of the messages this manager sends along with the operation that processes the message. Hence, as stated in Proposition 3.1, efficient networks are strongly overlapping.

In contrast, Properties P2–P4 are not necessary for efficiency. However, they are clearly necessary properties of MS networks. The maximum-slack condition first of all restricts the postprocessing, because managers should perform their operations, and hence send their messages, as late as possible. This means that each message should be sent just before it is processed (P3) and also that the postprocessing operations should be performed last (P4). For example, the network $\langle \mathcal{Q}, \mathcal{J} \rangle$ in Fig. 3 is not just-in-time. The network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is obtained by delaying the instructions of manager D. $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is just-in-time and has more slack than $\langle \mathcal{Q}, \mathcal{J} \rangle$. However, $\langle \mathcal{Q}, \mathcal{J}' \rangle$ does not have preprocessing first. By interchanging manager D's last preprocessing operation and manager A's first postprocessing

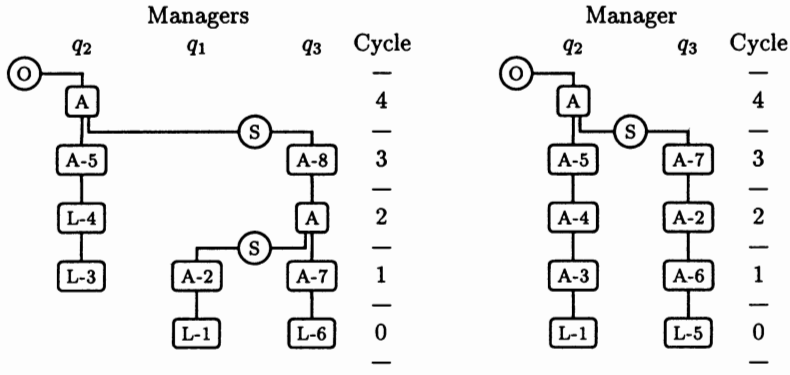


Fig. 2. The network on the left does not have strongly overlapping managers. The network on the right is obtained by combining the operations of managers q_1 and q_2. It also processes 7 items with a delay of 5, but it has 1 less manager and 1 less operation.

operation, we obtain the network $\langle \mathcal{Q}, \mathcal{J}'' \rangle$. This network does not have more slack than $\langle \mathcal{Q}, \mathcal{J}' \rangle$, but it is not just-in-time. By delaying manager B's operations by one cycle, we obtain the network $\langle \mathcal{Q}, \mathcal{J}''' \rangle$, which has more slack than $\langle \mathcal{Q}, \mathcal{J}' \rangle$.

Once the time each manager finishes is fixed by Properties P3 and P4, the slack is determined by the total number of operations and the total amount of time managers are idle between when they start and when they finish. The former is reduced by making a network simple if it is not already, and the latter is reduced by making a network continuous if it is not already. For example, after concatenating tasks in the network on the left side of Fig. 1 to obtain the simple network on the right side, and then delaying operations so that the network is continuous, we obtain the network in Fig. 4, which has more slack than the ones in Fig. 1. Hence, P1–P4 are necessary conditions for a network to have maximum slack. Theorem 3.1 states also that the converse holds if the network is strongly overlapping.

Properties P2–P4 are not necessary conditions for efficiency because, as long as an efficient network can have some slack, which is true if and only if $Q + N \bmod Q$ is a power of 2, the network can be perturbed so that P2, P3 or P4 is not satisfied, without affecting the number of operations and delay. For example, the efficient one-shot networks in Radner (1993) satisfy P1, P2, P4 and P5, but not necessarily P3 (just-in-time); in these networks, all managers begin processing in cycle 0 and hence there is no slack. However, we have the following result, which further supports the claim that MS networks resemble efficient one-shot networks:

Proposition 3.3 *Every network is weakly dominated by an MS network. If* $Q + N \bmod Q$ *is a power of* 2, *then every efficient network of size* Q *is an MS network.*

Proof. See Appendix A. □

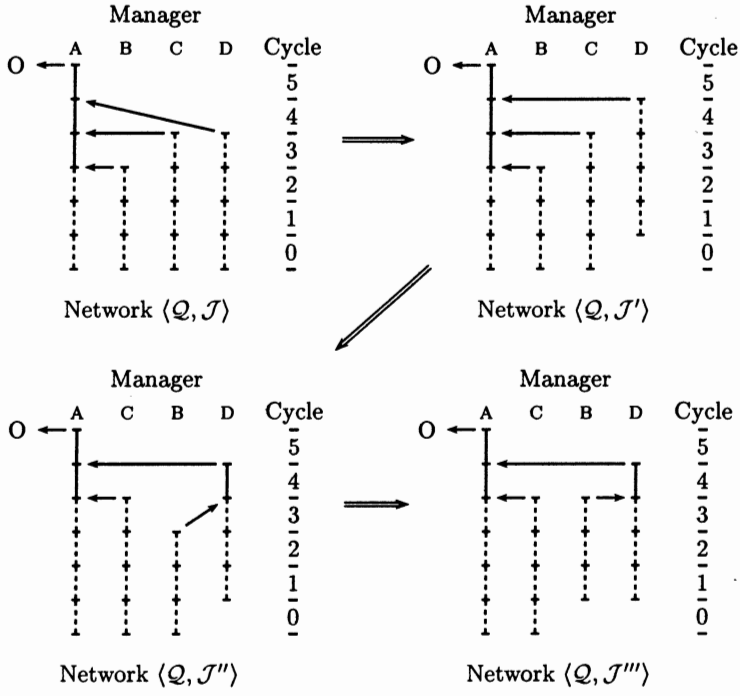


Fig. 3. Four one-shot networks with $N = 14$, $Q = 4$ and $D = 6$. $\langle \mathcal{Q}, \mathcal{J} \rangle$ is not just-in-time. $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is obtained by delaying the instructions of manager D by one cycle, which increases the slack. $\langle \mathcal{Q}, \mathcal{J}' \rangle$ does not have preprocessing first. $\langle \mathcal{Q}, \mathcal{J}'' \rangle$ is obtained by interchanging A's first postprocessing operation and D's last preprocessing operation. This does not increase slack, but the resulting network is not just-in-time. By delaying B's instructions, we obtain $\langle \mathcal{Q}, \mathcal{J}''' \rangle$, which has more slack than $\langle \mathcal{Q}, \mathcal{J}' \rangle$.

Remark 3.1 We have not required that MS network be efficient because we will use *inefficient MS networks* to construct *efficient periodic networks*. (Figure 5 shows an example of an MS network that is not efficient.) This will be explained in Sect. 4. We note here that the postprocessing in MS networks that are not efficient resembles the postprocessing in efficient MS networks. Specifically, if we take a particular MS network of size Q, we can obtain a class of MS networks of size Q (i) by adjusting the delay (i.e., increasing or decreasing the execution time of all instructions) to create more or less slack and (ii) by redistributing the raw data among the managers subject to the constraint that each manager processes two items before any manager finishes. Starting two cycles before the first manager finishes, the processing in different networks in the class differs only by the execution times of the instructions. Hence, all members of the class have the same communication subgraphs when the nodes of the graphs are restricted to the set of managers. Like the efficient networks in Radner (1993), the communication graph of each network is irregular, in that if each node is assigned a tier equal to the length of the longest path in the graph from the node to a data source, then each manager has subordinates in all the tiers below the

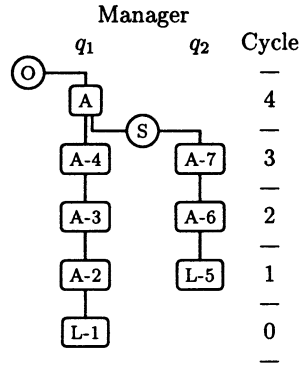


Fig. 4. A simple network with more slack than the network on the left side of Fig. 1, which is not simple.

manager's tier, including at least two that are data sources and hence are in the bottom tier. See, for example, the communication graph in Fig. 5.

Remark 3.2 The construction of the MS networks with Q managers and delay D can be described recursively starting from cycle D, when manager q_1 finishes with an OUTPUT instruction. In cycle $D-1$, only this manager can be busy, and so she processes a single message, sent by manager q_2 at the beginning of that cycle. Then in cycle $D-2$, these two managers are postprocessing messages sent by managers q_3 and q_4 at the beginning of that cycle, as seen in Fig. 5. Let $s \in \{1, \ldots, \lceil \log_2 Q \rceil - 1\}$. In cycle $D-s$, there are 2^{s-1} managers who are busy postprocessing messages from 2^{s-1} other managers, and so 2^s managers finish in or after cycle $D-s$ and hence are busy in cycle $D-s-1$. In cycle $D-\lceil \log_2 Q \rceil$, $2^{\lceil \log_2 Q \rceil - 1}$ managers are busy. $Q - 2^{\lceil \log_2 Q \rceil - 1} > 0$ of these managers process messages from the remaining managers. The rest, $2^{\lceil \log_2 Q \rceil} - Q \geq 0$, process raw data. Since the first managers to finish do so in this cycle, each manager must then have two raw data to process in cycles $D - \lceil \log_2 Q \rceil - 1$ and $D - \lceil \log_2 Q \rceil - 2$, so that the network is strongly overlapping. The remaining data are then distributed arbitrarily among the managers, and a manager with k additional items processes these in cycles $D - \lceil \log_2 Q \rceil - 2 - k, \ldots, D - \lceil \log_2 Q \rceil - 3$. This construction is well-defined as long as:

- There is enough data so that all managers have two raw data to process in cycles $D - \lceil \log_2 Q \rceil - 2$ and $D - \lceil \log_2 Q \rceil - 1$. This is true if and only if $N \geq 2Q + (2^{\lceil \log_2 Q \rceil} - Q)$, i.e., if and only if $Q \leq \bar{Q}(N)$, where $\bar{Q}(N)$ is the unique solution to $N = \bar{Q} + 2^{\lceil \log_2 \bar{Q} \rceil}$.
- It is possible to distribute the data so that each manager begins in or after cycle 0. As shown in the proof of Lemma A.6 in Appendix A, this is true as long as

$$D \geq D^*(Q,N) \equiv \lfloor N/Q \rfloor + \lceil \log_2(Q + N \bmod Q) \rceil \ .$$

$D^*(Q,N)$ is the minimum delay for processing N items with Q managers, as derived in Radner (1993), for $1 \leq Q \leq \bar{Q}(N)$.

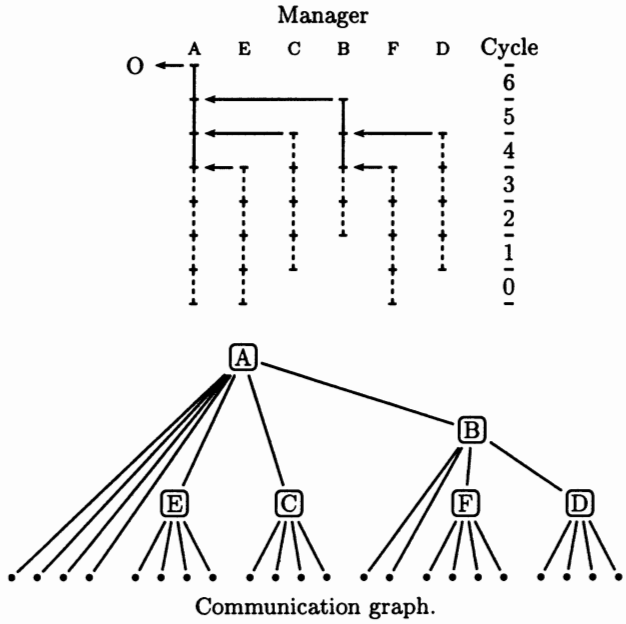


Fig. 5. An MS network with 6 managers for processing 21 data with a delay of 7. The slack is 4. In the communication graph, the data sources are shown simply as dots and all edges point upwards.

4 Efficient periodic networks

Theorem 5.1 in Van Zandt (1997c) tells us that the processing of each cohort by an essential, periodic network is separate. Therefore, we can represent the processing of each cohort by a one-shot network, in which time is measured relative to the time the cohort arrives. To define such representations, we need the following notation. Let $\langle \mathcal{M}, \mathcal{I} \rangle$ be an essential periodic network. For $\tau \in \mathcal{T}$, let $\mathcal{M}(\tau)$ be the managers who execute the instructions in $\mathcal{I}(\tau)$ and let $\mathcal{I}^0(\tau)$ be the set of instructions obtained from $\mathcal{I}(\tau)$ by subtracting τ from the execution time for each instruction in $\mathcal{I}(\tau)$ and by replacing cohort τ by 0 in each INPUT and OUTPUT instruction in $\mathcal{I}(\tau)$. Then $\langle \mathcal{M}(\tau), \mathcal{I}^0(\tau) \rangle$ is an essential one-shot network.

Two cohorts τ_1 and τ_2 may be processed in the same way but perhaps with different managers, in the sense that $\mathcal{I}^0(\tau_2)$ can be obtained from $\mathcal{I}^0(\tau_1)$ by reassigning tasks in $\mathcal{I}^0(\tau_1)$ to managers in $\mathcal{M}(\tau_2)$ and by relabeling message ID's. We would like to define a one-shot representation so that the processing of cohorts τ_1 and τ_2 can be represented by the same one-shot network. If we do not require that the mappings from managers in the one-shot representations to managers in the periodic network be one-to-one, then, without loss of generality, we can use simple networks for the one-shot representations.

Definition 4.1 *Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a one-shot network. A simple network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ is a* **simple representation** *of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with* **task assignment** *$\alpha : \mathcal{Q}' \to \mathcal{Q}$ if there is*

$f_a : \mathbb{A} \to \mathbb{A}$ such that $\mathcal{J}$ can be obtained from $\mathcal{J}'$ by replacing manager $q' \in \mathcal{Q}'$ and message ID $a' \in \mathbb{A}$ in the instructions in $\mathcal{J}'$ by $\alpha(q')$ and $f_a(a)$, respectively.

That is, $\langle \mathcal{Q}, \mathcal{J} \rangle$ can be obtained from $\langle \mathcal{Q}', \mathcal{J}' \rangle$ by reassigning the tasks and renaming the message ID's. There are obvious isomorphisms between the tasks in $\langle \mathcal{Q}, \mathcal{J} \rangle$ and the tasks in $\langle \mathcal{Q}', \mathcal{J}' \rangle$ and between the execution graphs of $\langle \mathcal{Q}, \mathcal{J} \rangle$ and $\langle \mathcal{Q}', \mathcal{J}' \rangle$.

If $\langle \mathcal{Q}, \mathcal{J} \rangle$ is a simple representation of $\langle \mathcal{M}(\tau), \mathcal{I}^0(\tau) \rangle$, then we call it a one-shot representation of the processing of cohort τ in $\langle \mathcal{M}, \mathcal{I} \rangle$. We will be interested in periodic networks in which the processing of each cohort can be represented by the same one-shot network, but perhaps with different task assignments.

Definition 4.2 *Let $\langle \mathcal{M}, \mathcal{I} \rangle$ be an essential periodic network, let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a simple network and let $\alpha_\tau : \mathcal{Q} \to \mathcal{M}(\tau)$ for $\tau \in \mathcal{T}$. $\langle \mathcal{M}, \mathcal{I} \rangle$ is a* **replication** *of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with* **task assignments** *$\{\alpha_\tau\}_{\tau \in \mathcal{T}}$ if and only if, for each $\tau \in \mathcal{T}$, $\langle \mathcal{Q}, \mathcal{J} \rangle$ is a simple representation of $\langle \mathcal{M}(\tau), \mathcal{I}^0(\tau) \rangle$ with task assignment α_τ. $\langle \mathcal{M}, \mathcal{I} \rangle$ is then said to be a* **replication network**.

We henceforth restrict our attention to replication networks and define efficiency to mean "undominated within this class". We conjecture that efficient networks are actually not dominated by any network, and in particular that any constant-delay network is weakly dominated by a replication network. However, even if this conjecture is true, there are networks without constant delay that are not weakly dominated by any replication network.

Each cohort has the same delay in a replication network, and so we denote this common value of $\{D(\tau) | \tau \in \mathcal{T}\}$ simply by D. We refer to the managers in the simple representation as tasks, in order to distinguish them from the managers in the periodic network. As defined in Remark 2.1, the duration of a task $q \in \mathcal{Q}$ in a simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$ is $d(q) = c(q) - b(q)$.

As defined in Van Zandt (1996), a *stationary* network is a replication network with the same task assignment for each cohort. That is, each cohort is processed in the same way by the same managers. In a stationary network, each manager must perform each assigned task every T cycles, and hence the duration of each task can be no more than T. This is a severe limitation on networks whose implications are explored in Bolton and Dewatripont (1994) and Van Zandt (1996).

Weaker than stationarity is the requirement that there be a disjoint collection of teams of managers, each of which periodically processes cohorts in the same way. There are multiple teams if the maximum duration of the tasks in the one-shot network exceeds T. Examples of such replication networks are the ROSE ("replications of one-shot efficient") networks in Radner (1993).

It is typically possible to improve upon such a network if we drop the requirement that the teams stay together from one cohort to another. Then managers whose tasks have short durations can process cohorts more frequently than those whose tasks have long durations. This suggests the following weaker form of time invariance called semi-stationarity, which means that each manager (i) al-

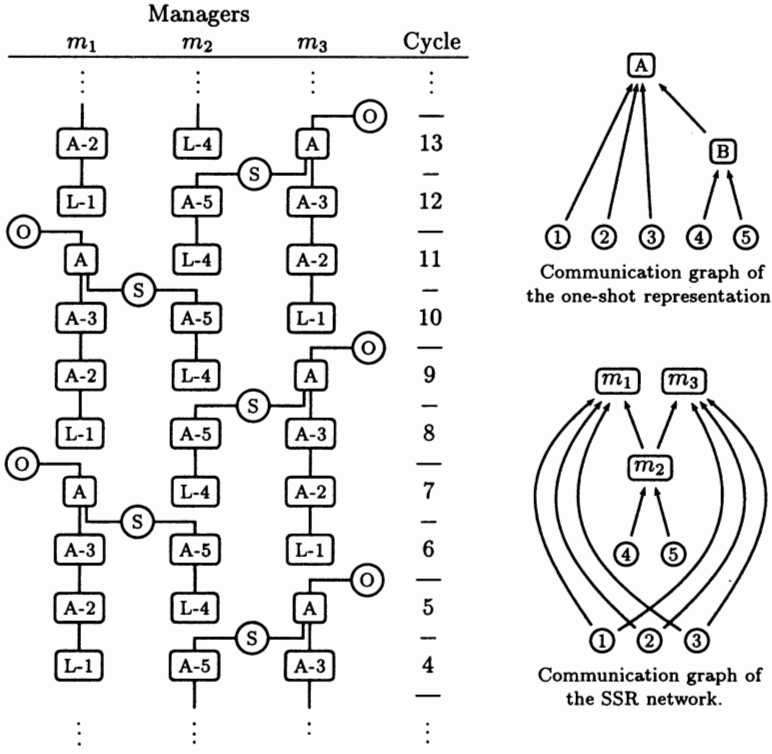


Fig. 6. A semi-stationary network for $N = 5$ and $T = 2$. There are 3 managers in the network, and each cohort is processed by 2 managers with a delay of 4. Manager m_2 repeats task B every 2 cycles and managers m_1 and m_3 repeat task A every 4 cycles. This network has no idle time.

ways performs the same task in the one-shot representation and (ii) switches to new cohorts as quickly as possible:

Definition 4.3 *$\langle \mathcal{M}, \mathcal{I} \rangle$ is a* **semi-stationary replication** *of $\langle \mathcal{Q}, \mathcal{J} \rangle$ if $\langle \mathcal{M}, \mathcal{I} \rangle$ is a replication of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with task assignments $\{\alpha_\tau | \tau \in \mathcal{T}\}$ such that $\forall m \in \mathcal{M}$, $\forall \tau_1, \tau_2 \in \mathcal{T}$ and $\forall q_1, q_2 \in \mathcal{Q}$: If $\alpha_{\tau_1}(q_1) = m$, then $\alpha_{\tau_2}(q_2) = m$ if and only if $q_1 = q_2$ and $|(\tau_2 - \tau_1)/T|$ is a multiple of $\lceil d(q)/T \rceil$. $\langle \mathcal{M}, \mathcal{I} \rangle$ is then said to be* **semi-stationary**.

Figure 6 shows part of the execution graph of a semi-stationary network (which is a replication of an MS network). Observe that manager m_2 alternates between processing a cohort with manager m_1 and processing a cohort with manager m_3, but performs the same task in the processing of each cohort.

For any simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$, every semi-stationary replication of $\langle \mathcal{Q}, \mathcal{J} \rangle$ has the same delay and number of managers. In fact, any two such networks $\langle \mathcal{M}, \mathcal{I} \rangle$ and $\langle \mathcal{M}', \mathcal{I}' \rangle$ are equivalent, in the sense that there are bijections $f_m : \mathcal{M} \to \mathcal{M}'$ and $f_a : \mathbb{A} \to \mathbb{A}$ such that $\mathcal{I}'$ is the set of instructions obtained by replacing manager $m \in \mathcal{M}$ and address $a \in \mathbb{A}$ in the instructions in $\mathcal{I}$ by $f_m(m)$

and $f_a(a)$, respectively. We denote the class of such networks by SSR($\mathcal{Q}, \mathcal{J}$), and also treat SSR($\mathcal{Q}, \mathcal{J}$) as a representative member of this class.

We state our main result:

Definition 4.4 *An* **RMS network** *is a semi-stationary replication of an MS network.*

Theorem 4.1 *Every replication network is weakly dominated by an RMS network.*

The first step in proving Theorem 4.1, which is also the most difficult, is the following proposition:

Proposition 4.1 *Every replication network is weakly dominated by a semi-stationary network.*

Proof. See Appendix B. □

The second and final step is easier and more intuitive, and the proof is included below:

Proposition 4.2 *Every semi-stationary network is weakly dominated by an RMS network.*

Remark 4.1 The proof of Proposition 4.2 makes use of the following fact. In SSR($\mathcal{Q}, \mathcal{J}$), a manager with task $q \in \mathcal{Q}$ repeats this task every $\lceil d(q)/T \rceil$ cohorts. Hence, $\lceil d(q)/T \rceil$ managers are assigned task q, and the number of managers in SSR($\mathcal{Q}, \mathcal{J}$) is equal to $\sum_{q \in \mathcal{Q}} \lceil d(q)/T \rceil$. Therefore, to find the semi-stationary replication network with given delay that has the fewest managers, we should search for the simple network with the given delay for which $\sum_{q \in \mathcal{Q}} \lceil d(q)/T \rceil$ is the smallest.

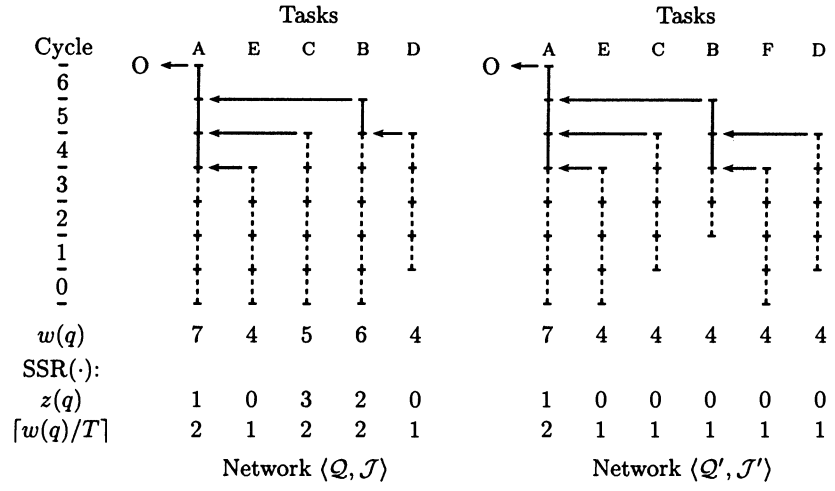
Remark 4.2 Furthermore, we can the decompose the managerial costs of SSR ($\mathcal{Q}, \mathcal{J}$) into operations and idle time, as follows. Let $w(q)$ be the number of operations in task q. According to Proposition 3.2, $\sum_{q \in \mathcal{Q}} w(q) = N + Q - 1$. Furthermore, $w(q) \leq d(q)$, and $d(q) - w(q)$ is the number of cycles that task q is idle while the task is active, i.e., after the task's LOAD and before the task's message. The number of cycles a manager with task q is idle between cohorts is

$$z(q) \equiv T\lceil d(q)/T \rceil - d(q) \ .$$

Therefore,

$$\begin{aligned} MT = \sum_{q \in \mathcal{Q}} T\lceil d(q)/T \rceil &= \sum_{q \in \mathcal{Q}} (w(q) + (d(q) - w(q)) + z(q)) \\ &= N + Q - 1 + \sum_{q \in \mathcal{Q}} (d(q) - w(q)) + \sum_{q \in \mathcal{Q}} z(q). \end{aligned}$$

$N + Q - 1$ is the number of operations per cohort, $\sum_{q \in \mathcal{Q}} (d(q) - w(q))$ is the per-cohort idle time *within* tasks, and $\sum_{q \in \mathcal{Q}} z(q)$ is the per-cohort idle time *between* tasks.



TOTALS	SSR$(\mathcal{Q},\mathcal{J})$	SSR$(\mathcal{Q}',\mathcal{J}')$
Tasks	5	6
Operations	26	27
Idle time	6	1
Managers	8	7

Fig. 7. These two MS networks process 22 items in 7 cycles. Also shown for each task are the idle time $z(q)$ and number of managers $\lceil w(q)/T \rceil$ who perform the task in the semi-stationary replication of each network when $T = 4$. Network $\langle \mathcal{Q}, \mathcal{J} \rangle$ on the left is one-shot efficient, whereas network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ on the right is not because it has one extra manager. However, when $T = 4$, SSR$(\mathcal{Q}', \mathcal{J}')$ has one less manager than SSR$(\mathcal{Q}, \mathcal{J})$ because it has less idle time. The communication graphs of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ and SSR$(\mathcal{Q}', \mathcal{J}')$ are shown in Figs. 5 and 8, respectively.

Remark 4.3 We can now see why MS networks are good networks to replicate. First, because they are continuous, there is no idle time within tasks. Second, because the tasks finish as late as possible in MS networks, there is slack that provides flexibility in distributing the raw data among the tasks in order to minimize the total idle time $\sum_{q \in \mathcal{Q}} z(q)$ (e.g., by setting each task's duration to a multiple of T, if possible). We can also see why it may be good to replicate *inefficient* MS networks. If we compare the MS networks with delay D and size Q_1 with the MS networks with delay D and size $Q_2 > Q_1$, the former have $Q_2 - Q_1$ fewer operations than the latter. However, the latter may have more slack and this slack can be used, as described above, to adjust the durations of the task in order to reduce idle time. This is illustrated in Fig. 7.

Proof of Proposition 4.2. According to Lemma A.2 in Appendix A, if $\langle \mathcal{Q}, \mathcal{J} \rangle$ does not satisfy P2, P3 or P4, then we can construct a network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ with the same delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$ by increasing the execution time of instructions and interchanging preprocessing and postprocessing operations, such that $\langle \mathcal{Q}, \mathcal{J}' \rangle$ has *more slack* than $\langle \mathcal{Q}, \mathcal{J} \rangle$ and the duration of each task in $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is as short as in $\langle \mathcal{Q}, \mathcal{J} \rangle$. (This is also discussed in Sect. 3 and is illustrated in Figs. 3

and 4.) Because $\lceil d(q)/T \rceil$ is weakly decreasing in $d(q)$, SSR$(\mathcal{Q}, \mathcal{J}')$ has as few managers as SSR$(\mathcal{Q}, \mathcal{J})$. Since the slack of essential networks with a given size and delay is bounded above, this construction can be iteratively applied only finitely many times, until $\langle \mathcal{Q}, \mathcal{J}' \rangle$ satisfies P2, P3 and P4. Hence:

Lemma 4.1 *Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a simple network. There is a simple network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ that satisfies P2, P3 and P4 such that* SSR$(\mathcal{Q}, \mathcal{J}')$ *weakly dominates* SSR$(\mathcal{Q}, \mathcal{J})$.

Proof. See above and Lemma A.2. □

Now suppose that a simple network, such as those replicated by Radner's PPO networks, is not strongly overlapping. Then there are two tasks q_1 and q_2 such that q_2 performs fewer than 2 operations before q_1 finishes. If q_2 has a single operation, then according to Meagher and Van Zandt (1997, Lemma 1) it is possible to delete the task and its instructions without changing the durations of other tasks. Otherwise, according to Meagher and Van Zandt (1997, Lemma 2), it is possible to combine tasks q_1 and q_2 into a single task q_2 such that the number of operations in the combined task is one less than the total number of operations in the two original tasks ($w'(q_2) = w(q_1) + w(q_2) - 1$). This is illustrated in Fig. 2. By increasing execution times of operations in the combined task to squeeze out idle time, as in Lemma A.2 in Appendix A, the duration of the combined task is equal to the number of operations in the task, which is less than $d(q_1) + d(q_2)$. The durations of the remaining tasks either do not change or decrease. Since

$$\lceil (d(q_1) + d(q_2))/T \rceil \leq \lceil d(q_1)/T \rceil + \lceil d(q_2)/T \rceil \ , \tag{4.1}$$

the number of managers who perform the combined task in the semi-stationary replication of the modified network is no greater than (and may be less than) the total number of managers who perform tasks q_1 and q_2 in the semi-stationary replication of the original network.[6] Hence, we have shown:

Lemma 4.2 *Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a simple network that is not strongly overlapping. There is a simple and strongly overlapping network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ that has fewer tasks than $\langle \mathcal{Q}, \mathcal{J} \rangle$ and whose semi-stationary replication has the same delay as and as few managers as* SSR$(\mathcal{Q}, \mathcal{J})$.

Proof. See above, Meager and Van Zandt (1997, Lemmas 1 and 2) and Lemma A.2. □

Now we combine Lemmas 4.1 and 4.2. Let SSR$(\mathcal{Q}, \mathcal{J})$ be a semi-stationary network. Starting with $\langle \mathcal{Q}, \mathcal{J} \rangle$, we obtain a sequence of simple networks by applying the construction in Lemma 4.2 if the network is not strongly overlapping, and then the construction in Lemma 4.1 if the resulting network does not satisfy P2, P3 or P4. The former decreases the number of tasks and the latter leaves it unchanged. Therefore, after finitely many iterations, we obtain an MS network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ such that SSR$(\mathcal{Q}', \mathcal{J}')$ weakly dominates SSR$(\mathcal{Q}, \mathcal{J})$.

This concludes the proof of Proposition 4.2. □

[6] It may be strictly less because (4.1) can hold with strict inequality and because the elimination of an operation from combining the tasks means that the duration of the combined task is strictly less than $d(q_1) + d(q_2)$.

5 Efficient performance

Radner (1993) shows that the minimum delay in the one-shot model is $1 + \lceil \log_2 N \rceil$. For D such that $1 + \lceil \log_2 N \rceil \leq D \leq N$, let $Q^*(D,N)$ be the size of the smallest one-shot network with a delay of D.

Consider the periodic model with N data and T cycles between cohorts. For D such that $1 + \lceil \log_2 N \rceil \leq D \leq N$, let $M^*(D,N,T)$ be the minimum number of managers in a replication network with delay D. We can construct the RMS network with delay D and $M^*(D,N,T)$ managers, and thereby calculate $M^*(D,N,T)$, using an algorithm such as the following: Start with an RMS network with delay D and $Q^*(D,N)$ tasks, distributing the data to tasks so as to minimize idle time. Now replicate an MS network with one extra task and again distribute the data so as to minimize the idle time. Continue to do this until adding a task does not decrease the idle time and thus does not improve the performance. The serial run-time of a crude implementation of this algorithm is $O(N^2)$; for D and N such that $Q^*(D,N) < \sqrt{N}$, the run-time is $O(\sqrt{N})$.[7] An efficiency frontier calculated this way can be found in Radner and Van Zandt (1992).

If it were possible to replicate an efficient one-shot network with delay D without any idle time, then the replication would have

$$M_L^*(D,N,T) \equiv \frac{N + Q^*(D,N) - 1}{T}$$

managers. This is a lower bound on $M^*(D,N,T)$ that is easier to calculate. Furthermore, $M_L^*(D,N,T)$ is the average number of operations per cycle when managers are not paid when idle (in which case it is optimal to replicate efficient one-shot networks). Hence, $M^*(D,N,T) - M_L^*(D,N,T)$ measures the increase in managerial costs due to paying idle time. Proposition 5.1 shows that $M_L^*(D,N,T)$ can be a good approximation of $M^*(D,N,T)$ when the delay is far enough above the minimum delay:

Proposition 5.1

1. *For all D and N such that* $1 + \lceil \log_2 N \rceil \leq D \leq N$,

(5.1)
$$M^*(D,N,T) \leq \left(\frac{T + 2 - 2/N}{3 - 2/N} \right) M_L^*(D,N,T) < T \cdot M_L^*(D,N,T) .$$

2. *If $N \to \infty$, $D \to \infty$ and $Q^*(D,N)/N \to 0$, then*
$$\frac{M^*(D,N,T) - M_L^*(D,N,T)}{M_L^*(D,N,T)} \to 0$$
at a rate $O((Q^*(D,N)/N)^2)$.
3. *If $Q^*(D,N) < \sqrt{N}$, then $M^*(D,N,T) \leq \lceil M_L^*(D,N,T) \rceil + 1$.*

[7] See the proof of part 3 of Proposition 5.1.

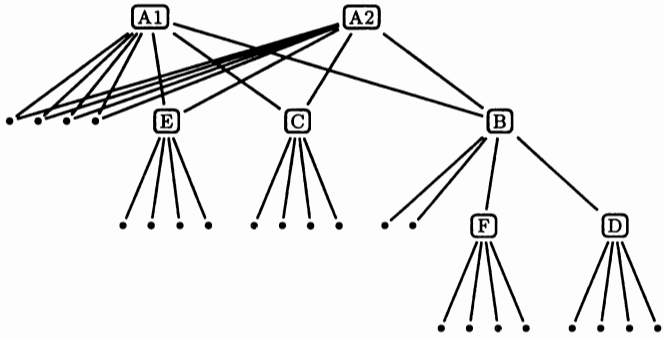


Fig. 8. The communication graph of the RMS network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ shown in Fig. 7. Managers A1 and A2 repeat task A every 8 cycles. The remaining managers are labeled by the task that they repeat every 4 cycles. All edges point upwards.

Proof. See Appendix B. □

A plot showing the difference between $M^*(D,N,T)$ and $M_L^*(D,N,T)$ can be found in Radner and Van Zandt (1992).

6 Robustness of irregularity of hierarchies

As discussed in Sect. 1, one motivation for studying this problem of periodic computation with salaried managers was to see whether the communication graphs of the efficient periodic networks would more closely resemble balanced hierarchies than do those of the efficient one-shot networks. However, we find that this is not the case. The RMS networks are replications of MS networks, which as discussed in Sect. 3 are irregular and resemble efficient one-shot networks. For example, Fig. 5 shows the communication graph of the MS network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ in Fig. 7. Furthermore, the RMS networks are not even hierarchical, even though they are semi-stationary replications of hierarchical one-shot networks. For example, Fig. 8 shows the communication graph of the RMS network SSR$(\mathcal{Q}', \mathcal{J}')$ in Fig. 7 (see also Fig. 6). Although the communication graphs of the RMS networks have no cycles – like any semi-stationary replication of a simple network whose communication graph has no cycles – each manager may communicate to more than one manager. Also, the communication graphs are not rooted since each manager whose task executes an OUTPUT is a maximal element in its communication graph, and there is more than one such manager as long as the delay is much longer than the time between cycles. In this section, we consider how changes in the computation model could change these results.

Our measure of managerial costs does not take into account the cost of memory and communication links. Recall, however, that a simple network with Q managers uses exactly $Q - 1$ communication links between managers, and any other essential one-shot network with Q managers uses at least this many. Furthermore, the MS networks have the property that each manager processes a

message in the same cycle in which the message is sent (if the execution times of the INPUT statements are set properly) and hence each manager in an MS network needs memory for at most for one message. Therefore, efficient one-shot networks economize in their use of communication links and efficient MS networks also economize in their use of memory.

Like the one-shot MS networks, the RMS networks use a minimal amount of memory. However, unlike the MS networks and unlike periodic networks whose communication graphs are trees or forests of trees, the RMS networks may use more than one communication link per manager. For example, suppose that 999 data arrive every 2 cycles. The RMS network with a delay of 252 has 4 tasks per cohort, and the postprocessing is like that of network $\langle \mathcal{Q}, \mathcal{T}''' \rangle$ in Fig. 3. Tasks A, B and C have 250 preprocessing operations and Task D has 249. (Task D begins 1 cycle later than the others.) Hence, the duration of tasks B, C and D is 250 cycles, while the duration of task A is 252 cycles. There are no idle cycles. 126 managers are assigned task A and 125 managers are assigned to each of tasks B, C and D. Thus, there are a total of 501 managers in the network.

Three managers who are assigned tasks B, C and D for one cohort finish at the same time and move on together to the next cohort. However, the manager who is assigned task A in the new cohort is different. The three managers are teamed up with a total of 126 managers who perform task A. Although a manager with task B always sends his partial sum to the same manager who has task D, managers with tasks C and D communicate with 126 different managers who have task A. Therefore, the number of communication links between managers is $125 + 126 \cdot (125 + 125)$.

Suppose that instead we keep the managers together in teams. Then three managers who are assigned tasks B, C and D for one cohort must be idle for two cycles aftering finishing the cohort, while they wait for the manager with task A to finish. Therefore, the number of managers increases to $4 \cdot 126 = 504$. However, each manager with task B, C or D now always sends his partial sum to the same manager, and so the number of communication links between managers is $126 \cdot 3$. Thus, although this new network has 3 more managers than the RMS network, it has nearly $2 \cdot 125^2$ fewer communication links. It may well be that if the cost of communication links is taken into account, this network is preferred to the RMS network.

An indirect way to study how costly communication links may affect the set of efficient networks is to require that networks be stationary, meaning that each problem is processed in the same way by the same managers. If such a network is hierarchical, then each manager sends messages to at most one manager, and the number of communication links is one less than the number of managers. The maximum time any manager is busy processing each problem determines the throughput, i.e., the rate at which the organization can process problems. In the efficient one-shot networks, managers higher in the hierarchy work longer, and hence there would be a bottleneck when processing periodically arriving problems. Redistributing tasks from these managers to managers lower in the hierarchy can increase throughput, at the cost of additional delay. Bolton and

Dewatripont (1994) show that if throughput is much more important than delay, then this stationarity assumption may lead to more regular hierarchies, and if not, at least to networks that are quite different from those in Radner (1993). Further results in Van Zandt (1996) also indicate that stationarity can make hierarchies more regular.

7 Related problems in computer science

The organization design problem that we outlined, like the ones in Keren and Levhari (1979, 1983, 1989), Radner (1993) and Bolton and Dewatripont (1994) and in the more general models of Mount and Reiter (1990) and Reiter (1996), could also be interpreted as algorithm design problems for a parallel processing machine or for a distributed network of computers. This is not because these authors have casually reinterpreted models of machines as models of humans, but rather because very simple models of joint processing by humans and very simple models of joint processing by machines naturally resemble each other.

Because economists who study decentralized information processing in organizations have different objectives from computer scientists who study parallel processing by machines, it is not easy to find the answers to questions that economists pose about organizations just by sifting through the existing literature on parallel and distributed processing. (See, for example, Zomaya (1996) for an overview of this literature.) It would be beyond the scope of this paper to summarize the insights that such an exercise would yield, but we can briefly relate this paper to some research in computer science.

To the knowledge of this author, which is based on a selective reading of the parallel and distributed processing literature and on private communication with active researchers in the field, the particular model and problem posed here have not been studied or solved before. A crude explanation for this is that the problem is a special one and a random draw of all such problems is unlikely to yield one that has already been analyzed. There may also be more systematic reasons why this model, which was put forth non-randomly to address questions about organizations, has not been studied by computer scientists or operations researchers.

The design of parallel algorithms for a multiprocessor system is often divided into (i) the writing of an algorithm in a high-level language that abstracts from the implementation on a particular machine and hence from certain resource constraints such as the number of processors and communication costs, and (ii) the assignment of the tasks in the algorithm to the processor elements of a particular machine. The latter step is called scheduling, which may be either static or dynamic, depending on whether the scheduling is done by the compiler or is done at run time. (See the papers in Shirazi et al. (1995) for an overview of static and dynamic scheduling.) In this paper, we perform both the design of the high-level algorithm and the *static* scheduling. In general, these two steps cannot be completely separated because the optimal way to decompose computation problems in step (i) typically depends on step (ii).

To be effective, static scheduling requires good estimates of the execution times of the tasks and the communication delays, and these should not be data dependent. Furthermore, this information must be provided to the compiler. While these conditions might be satisfied by a single computation job, they would rarely be satisfied by a flow of jobs that are submitted to run on a multi-purpose system. Such calculations are likely to be random both in the type and size of each calculation and also in the arrival time. Therefore, the scheduling of independent but overlapping jobs is usually done by dynamic scheduling, which is also referred to as (dynamic) load balancing. Whereas static scheduling can be quite fine-grained, meaning that the tasks are highly decomposed and the scheduling involves assigning each operation to a processor, fine-grained dynamic scheduling typically is not practical, and dynamic scheduling may simply involve the assignment of entire jobs to servers in a network.

Our model, which has a flow of calculations, is amenable to static scheduling because the jobs are homogeneous in type and size and because the intervals between jobs are deterministic and constant over time. Those who study static scheduling for the sake of designing compilers would find this special case uninteresting, because the compiler should be able to handle a wide range of jobs. However, an organization typically is not a multi-purpose machine ready to process information for outside clients. Instead, like the networks in this paper, the organization's administrative structure and the procedures the agents in that structure follow are designed to handle the anticipated recurring information processing tasks that are part of controlling the organization. While real-time control, as in Radner and Van Zandt (1992), Van Zandt and Radner (1997) and Van Zandt (1997a,b), is an even more accurate model of an organization's information processing activity, and while a real organization's activities are far from deterministic and perfectly anticipated, this model of periodic computation is simple but still captures the ongoing nature of information processing in organizations.

The simplicity of associative computation along with the periodicity of the arrival of jobs is also what allows us to obtain exact results about fully optimal networks; these results include a characterization of a class of networks that span the entire efficiency frontier with respect to delay and processing costs and a characterization of the interprocessor communication in these networks. Finding exact solutions to simple problems is also done in computer science, but it is more common to pursue approximate results for more complex problems. Approximate results might take the form of asymptotic bounds on parallel complexity or the design of parallel algorithms or scheduling procedures that are shown either analytically or through simulations to have performance that is within a fixed ratio of the optimum. This is particularly true for the static scheduling of a single job, for which finding the optimum is known to be NP-complete. As an indication of how special and simple the scheduling problem we study is (in spite of the fact that there is a flow of jobs, there are some communication costs, and the registers represent local memory), we find that optimal schedules can be calculated in polynomial time (see Sect. 5).

Appendices: Proofs

A Characterization of MS networks

It is recommended that all of Sect. 3 be read before proceeding with the proofs in this appendix.

In this appendix, all networks are *essential one-shot networks*. For such a network $\langle \mathcal{Q}, \mathcal{J} \rangle$ and for manager $q \in \mathcal{Q}$, we use the following notation:

$b(q)$≡cycle in which q begins (execution time of q's first LOAD).
$c(q)$≡cycle in which q finishes (execution time of q's last message).
$w(q)$≡number of operations performed by q.
$d(q)$≡duration of q; $d(q) = c(q) - b(q)$.

We will also denote some one-shot networks with primes or subscripts (e.g., $\langle \mathcal{Q}', \mathcal{J}' \rangle$), in which case primes or subscripts are added to the above notation (e.g., b', c', d' and w', and also D', Q' and W').

Remark A.1 Since the last instruction a manager q executes in an essential network is a message, she performs no operation in cycle $c(q)$. Therefore, $w(q) \leq c(q) - b(q)$ and $w(q) = c(q) - b(q)$ if and only if q is not idle between cycles $b(q)$ and $c(q)$.

Proof of Theorem 3.1. Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be an essential network with size Q and delay D. We prove (Step 1) that P0 $\Longrightarrow$ (P1–P4) and then (Step 2) that (P1–P5) $\Longrightarrow$ P0. The proofs of all lemmas are given after the proof of this theorem.

Step 1. We first show $\neg$P1 $\Longrightarrow$ $\neg$P0: Suppose that $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$ is not simple. To show that $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$ is not a maximum-slack network, we should construct a network *with the same number of managers* as $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$ that has more slack. The concatenation of tasks shown in Fig. 1 and that is the basis of Lemma 3 in Meagher and Van Zandt (1997) can result in a network with fewer managers, because when one manager's SEND is eliminated, another manager's operation is also eliminated. Therefore, we instead use the concatenation that is shown in Fig. 2 and that is the basis of Lemma 2 in Meagher and Van Zandt (1997). In this case, when a manager's SEND is eliminated, the manager's previous operation is transferred to the manager who processes the SEND.

Lemma 2 in Meagher and Van Zandt (1997) is stated in terms of concatenating tasks in a simple network. To apply this lemma, we first construct a simple network $\langle \mathcal{Q}_1, \mathcal{J}_1 \rangle$ by reassigning all but the last of each manager's tasks in $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$. Reassignment of tasks is described in Remark 2.1; there is an obvious isomorphism between the tasks in $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$ and $\langle \mathcal{Q}_1, \mathcal{J}_1 \rangle$ such that the instructions in corresponding tasks differ only by the name of the manager who executes the instructions. This particular construction of a simple network is given in Definition 4.1, and is such that $\mathcal{Q}_0 \subset \mathcal{Q}_1$ and for $q \in \mathcal{Q}_0$, q's unique task in $\langle \mathcal{Q}_1, \mathcal{J}_1 \rangle$ corresponds to q's *last* task in $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$.

According to Lemma 2 in Meagher and Van Zandt (1997), we can iteratively concatenate all tasks in $\langle \mathcal{Q}_1, \mathcal{J}_1 \rangle$ that are performed by the same manager in

$\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$, as no two of these tasks can be active at the same time. As constructed in that lemma, we thereby obtain a simple network $\langle \mathcal{Q}_0, \mathcal{J}_2 \rangle$ such that, for all $q \in \mathcal{Q}_0$, $c_2(q) = c_0(q)$ and $w_2(q) \leq w_0(q)$, with a strict inequality if q has multiple tasks in $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$. Apply Lemma A.1 below to $\langle \mathcal{Q}_0, \mathcal{J}_2 \rangle$ to obtain a continuous, simple network $\langle \mathcal{Q}_0, \mathcal{J}_3 \rangle$:

Lemma A.1 *Suppose $\langle \mathcal{Q}, \mathcal{J} \rangle$ is simple. Then there is a continuous simple network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ the same size and delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$, such that $c'(q) = c(q)$ and $w'(q) = w(q)$ for all $q \in \mathcal{Q}$.*

Then, for $q \in \mathcal{Q}_0$, $c_3(q) = c_0(q)$ and $b_3(q) = c_0(q) - w_2(q) \geq c_0(q) - w_0(q) \geq b_0(q)$. As the first strict inequality is strict if q has multiple tasks in $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$, the network $\langle \mathcal{Q}_0, \mathcal{J}_3 \rangle$ has more slack than $\langle \mathcal{Q}_0, \mathcal{J}_0 \rangle$.

To complete this step, we invoke Lemma A.2, which shows that P1 $\wedge$ ($\neg$P2 $\vee$ $\neg$P3 $\vee$ $\neg$P4) $\Longrightarrow$ $\neg$P0:

Lemma A.2 *Suppose $\langle \mathcal{Q}, \mathcal{J} \rangle$ is simple but does not have Property P2, P3 or P4. Then there is a simple network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ with the same size and delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$ such that $d'(q) \leq d(q)$ $\forall q \in \mathcal{Q}$, $\sum_{q \in \mathcal{Q}} b'(q) > \sum_{q \in \mathcal{Q}} b(q)$ and $\sum_{q \in \mathcal{Q}} w'(q) = \sum_{q \in \mathcal{Q}} w(q)$.*

Step 2. The key to this step is to show that networks that satisfy Properties P1–P5 attain an upper bound on how late managers finish. Since also simple networks have the fewest operations of networks of the same size and networks with continuous tasks have the least idle time within tasks, networks that satisfy P1–P5 have maximum slack. To begin the proof, define the following property for a network $\langle \mathcal{Q}, \mathcal{J} \rangle$ with Q managers and delay D:

P6. For $s \in \{0, 1, \ldots, D\}$, exactly $\min\{Q, 2^s\}$ managers finish in or after cycle $D - s$.

Remark A.2 An equivalent statement of P6 is that one manager finishes in cycle D, 2^{s-1} managers finish in cycle $D - s$ for $1 \leq s < \lceil \log_2 Q \rceil$, $Q - 2^{\lceil \log_2 Q \rceil - 1}$ managers finish in cycle $D - \lceil \log_2 Q \rceil$, and no managers finish in earlier cycles.

Lemma A.3 *For $s \in \{0, 1, \ldots, D\}$, at most $\min\{Q, 2^s\}$ managers finish in or after cycle $D - s$.*

Lemma A.4 *If $\langle \mathcal{Q}, \mathcal{J} \rangle$ satisfies P1–P5, then it satisfies P6.*

Lemma A.5 *If $\langle \mathcal{Q}, \mathcal{J} \rangle$ satisfies P1, P2 and P6, then it satisfies P0.*

(P1–P5) $\Longrightarrow$ P0 then follows from Lemmas A.4 and A.5. □

Proof of Lemma A.1. We can weakly increase the execution times of each manager's operations, without changing the order of the manager's instructions, so that the manager is not idle between when the manager begins and when he finishes. Each manager's instructions still constitute a task. Because we have not changed the execution time of any message, each INPUT or SEND is executed

before the operation that processes it. Hence, we have obtain a well-defined, essential network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ that is simple and continuous, such that, $\forall q \in \mathcal{Q}$, $c'(q) = c(q)$ and $w'(q) = w(q)$. □

Proof of Lemma A.2. P2. Suppose that $\langle \mathcal{Q}, \mathcal{J} \rangle$ is not continuous. The network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ constructed in Lemma A.1 has the stated properties.

P3. Suppose that $\langle \mathcal{Q}, \mathcal{J} \rangle$ is not just-in-time. Then there is a manager $q \in \mathcal{Q}$ who has a SEND in cycle t_1 that is processed in cycle $t_2 > t_1$. We construct $\mathcal{J}'$ from $\mathcal{J}$ by increasing the execution time of each of q's instructions by $t_2 - t_1$ cycles. Because q's message is then sent in cycle t_2, which is when the message is processed, it follows from Corollary 5.1 in Van Zandt (1997c) that $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is essential and has the same delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$. Observe that (i) $d'(q) = d(q)$ and $b'(q) = b(q) + (t_2 - t_1)$, and (ii) $d'(q) = d(q)$ and $b'(q') = b(q')$ for $q' \in \mathcal{Q} \setminus \{q\}$. For example, compare the networks on the left side of Fig. 3 with those on the right side.

P4. Suppose that there are managers $q_1, q_2 \in \mathcal{Q}$ such that manager q_1 performs a postprocessing operation i_1 in cycle t_1 and manager q_2 performs a preprocessing operation i_2 in cycle $t_2 > t_1$. We first construct $\mathcal{J}''$ from $\mathcal{J}$ by interchanging the message ID's a_1 and a_2 of i_1 and i_2, respectively. Since a_2 is an INPUT instruction, we can change its time of execution to t_1. Message a_1 is a SEND that is executed in or before cycle $t_1 < t_2$. Hence, each message is still sent before it is processed and $\langle \mathcal{Q}, \mathcal{J}'' \rangle$ is well-defined and essential. (For example, compare the networks $\langle \mathcal{Q}, \mathcal{J}' \rangle$ and $\langle \mathcal{Q}, \mathcal{J}'' \rangle$ in Fig. 3.) $\langle \mathcal{Q}, \mathcal{J}'' \rangle$ does not have more slack than $\langle \mathcal{Q}, \mathcal{J} \rangle$, but it is not just-in-time. Since also the beginning times and durations of the tasks are the same in $\langle \mathcal{Q}, \mathcal{J}'' \rangle$ and $\langle \mathcal{Q}, \mathcal{J} \rangle$, it follows from the preceding paragraph that there is a network $\langle \mathcal{Q}, \mathcal{J}' \rangle$ with the stated properties. (For example, compare the networks $\langle \mathcal{Q}, \mathcal{J}' \rangle$ and $\langle \mathcal{Q}, \mathcal{J}''' \rangle$ in Fig. 3.) □

Proof of Lemma A.3. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is essential, each manager sends a message in the cycle in which she finishes. One of these can be an OUTPUT instruction, and the remainder are SEND instructions with parents that are operations. Hence, (∗) if K managers finish in cycle $D - s$, then at least $K - 1$ operations are performed during or after cycles $D - s$.

Since no operations are performed during or after cycle D, at most 1 manager finishes in cycle D (the manager who performs the OUTPUT instruction) and hence, the proposition holds for $s = 0$.

We conclude with the following inductive step. Suppose that $0 \leq s < \log_2 Q$ and for $r = 0, \ldots, s$, no more than 2^r managers finish during or after cycle $D - r$, and hence no more than 2^r managers are busy during cycle $D - r - 1$. Then no more than $2^0 + \cdots + 2^s = 2^{s+1} - 1$ operations are performed during or after cycle $D - (s + 1)$, and by (∗), at most 2^{s+1} managers finish in or after cycle $D - (s + 1)$. (The proposition holds for $s \geq \log_2 Q$, because then $\min\{Q, 2^s\} = Q$.) □

Proof of Lemma A.4. Suppose that $\langle \mathcal{Q}, \mathcal{J} \rangle$ satisfies P1–P5.

Since only the manager who executes the OUTPUT instruction finishes in cycle D and no manager finishers after cycle D, $1 = \min\{Q, 2^0\}$ manager finishes in or after cycle D.

Inductive step: Now suppose that $0 \leq s < \log_2 Q$ and exactly 2^s managers finish in or after cycle $D - s$. Consider two cases:

Case 1. Suppose that some manager preprocesses in cycle $D-(s+1)$. P4 implies that no manager postprocesses before cycle $D - (s + 1)$. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is just-in-time (P3), no manager finishes before cycle $D - (s + 1)$. Hence, all Q managers finish in or after cycle $D - (s + 1)$. By Lemma A.3, this implies that $(s + 1) \geq \lceil \log_2 Q \rceil$, and hence $Q = \min\{Q, 2^{s+1}\}$.

Case 2. No manager preprocesses in cycle $D-(s+1)$. Since $s < \log_2 Q$, it follows from Lemma A.3 that some managers finish before cycle $D - s$. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is strongly overlapping (P5), no managers begin after cycle $D - (s + 1)$. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is continuous (P2), the 2^s managers who finish in or after cycle $D - s$ are busy in cycle $D - (s + 1)$. By assumption, they are postprocessing. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is just-in-time (P3), 2^s managers finish in cycle $D - (s + 1)$, and so 2^{s+1} managers finish in or after cycle $D - (s + 1)$, and it must be that $2^{s+1} = \min\{Q, 2^{s+1}\}$. □

Proof of Lemma A.5. Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ and $\langle \mathcal{Q}', \mathcal{J}' \rangle$ be networks with Q managers and delay D. Let W and W' be the number of operations in $\mathcal{J}$ and $\mathcal{J}'$, respectively. Assume that $\langle \mathcal{Q}, \mathcal{J} \rangle$ satisfies P1, P2 and P6.

It follows from Remark A.1 and the assumption that $\langle \mathcal{Q}, \mathcal{J} \rangle$ is continuous that

$$\text{(A.1)} \qquad W = \sum_{q \in Q} c(q) - b(q) \qquad \text{and} \qquad W' \leq \sum_{q \in Q'} c'(q) - b'(q) \ .$$

Since in an essential network every manager sends a message and the number of postprocessing operations equals the number of SEND instructions, and since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is simple,

$$\text{(A.2)} \qquad W = N + Q - 1 \leq W' \ .$$

Combining (A.1) and (A.2):

$$\text{(A.3)} \qquad \sum_{q \in Q} c(q) - b(q) \leq \sum_{q \in Q'} c'(q) - b'(q) \ .$$

Lemma A.3 and the assumption that $\langle \mathcal{Q}, \mathcal{J} \rangle$ satisfies P6 imply that $\sum_{q \in Q} c(q) \geq \sum_{q \in Q'} c'(q)$; combining this with (A.3) we have

$$\sum_{q \in Q} b(q) \geq \sum_{q \in Q'} b'(q) \ .$$

Hence, $\langle \mathcal{Q}, \mathcal{J} \rangle$ has as much slack as $\langle \mathcal{Q}', \mathcal{J}' \rangle$. □

Proof of Proposition 3.3. Any network is weakly dominated by an efficient network $\langle \mathcal{Q}, \mathcal{J} \rangle$, which is simple. Let Q be its size and D be its delay. Let $\langle \mathcal{Q}, \mathcal{J}' \rangle$

be a network with the most slack of those with size Q and delay D. $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is thus a maximum-slack network and must be simple (as shown, e.g., in Step 1 of the proof of Theorem 3.1). Hence, it has the same number of operations as $\langle \mathcal{Q}, \mathcal{J} \rangle$ and is also efficient. As shown in Meagher and Van Zandt (1997), efficient networks are strongly overlapping. Hence, $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is an MS network.

Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be an efficient network and let $\langle \mathcal{Q}, \mathcal{J}' \rangle$ be an MS network that weakly dominates $\langle \mathcal{Q}, \mathcal{J} \rangle$. Then both networks have the same number Q of managers and the same delay $D = D^*(Q,N)$. If also $Q + N \bmod Q$, then it follows from Lemma A.6 below that $\langle \mathcal{Q}, \mathcal{J}' \rangle$ has zero slack. Hence, $\langle \mathcal{Q}, \mathcal{J} \rangle$ must also have zero slack, and it is a maximum-slack network. Since $\langle \mathcal{Q}, \mathcal{J} \rangle$ is efficient, it is strongly overlapping, and hence is an MS network. □

Lemma A.6 *There is an MS network with Q managers and delay D if and only if $Q + 2^{\lceil \log_2 Q \rceil} \leq N$ and $D \geq D^*(Q,N)$. An MS network with Q managers and delay D has zero slack if and only if $D = D^*(Q,N)$ and $Q+N \bmod Q$ is a power of 2.*

Proof. First we show that if there is an MS network with size Q, then $Q + 2^{\lceil \log_2 Q \rceil} \leq N$. From Theorem 3.1 and Lemma A.4, the first manager to finish in an MS network of size Q does so in cycle $D - \lceil \log_2 Q \rceil$. Since the MS network is strongly overlapping, each manager has at least 2 operations before this cycle for a total of $2Q$ operations. Furthermore, since the network satisfies P6 and is continuous, 2^{s-1} managers are busy in cycles $D - s$ for $s = 1, \ldots, \lceil \log_2 Q \rceil$, for a total of $2^{\lceil \log_2 Q \rceil} - 1$ operations. Hence, the network has at least $2Q + 2^{\lceil \log_2 Q \rceil} - 1$ operations. Since the network is simple, there are $N + Q - 1$ operations, and so $N + Q - 1 \geq 2Q + 2^{\lceil \log_2 Q \rceil} - 1$, or $N \geq Q + 2^{\lceil \log_2 Q \rceil}$.

If an MS network has Q managers and delay D, then $D \geq D^*(Q,N)$, since $D^*(Q,N)$ is the minimum delay of any network with Q managers.

For the converse, suppose that $Q+2^{\lceil \log_2 Q \rceil} \leq N$ and $D \geq D^*(Q,N)$. Consider the construction of an MS network with size Q and delay D in Remark 3.2. It is well-defined as long as it is possible to divide the $N - (2^{\lceil \log_2 Q \rceil} - Q)$ items that must be preprocessed in the first $D - \lceil \log_2 Q \rceil$ cycles among the Q managers so that each manager has (i) at least 2 items and (ii) no more than $D - \lceil \log_2 Q \rceil$ items. Since $Q+2^{\lceil \log_2 Q \rceil} \leq N$, $N-(2^{\lceil \log_2 Q \rceil}-Q) \geq 2Q$, and so (i) is possible. For (ii), it is necessary and sufficient that the number $Q(D - \lceil \log_2 Q \rceil)$ of available manager-cycles be as large as the number of items to be processed; i.e., that

$$S \equiv Q(D - \lceil \log_2 Q \rceil) - (N - (2^{\lceil \log_2 Q \rceil} - Q)) \geq 0 .$$

Note that if this holds ($S \geq 0$), then S is the slack of the MS network. We conclude by showing that $S \geq 0$ if $D \geq D^*(Q,N)$ and $S = 0$ if and only if $D = D^*(Q,N)$ and $Q + N \bmod Q$ is a power of 2.

Recall that $D^*(Q,N) = \lfloor N/Q \rfloor + \lceil \log_2(Q + N \bmod Q) \rceil$, and note that $N - Q\lfloor N/Q \rfloor = N \bmod Q$. Then $S = Q(D - D^*(Q,N)) + S^*$, where

$$S^* \equiv Q(\lceil \log_2(Q + N \bmod Q) \rceil - \lceil \log_2 Q \rceil) + 2^{\lceil \log_2 Q \rceil} - (Q + N \bmod Q) .$$

We need to show that $S^* = 0$ if $Q + N \bmod Q$ is a power of 2 and $S^* > 0$ otherwise. If $Q+N \bmod Q$ is a power of 2, then, since $N \bmod Q < Q$, $\lceil \log_2 Q \rceil = \log_2(Q+N \bmod Q)$ and $2^{\lceil \log_2 Q \rceil} = Q+N \bmod Q$. Hence, $S^* = 0$. Suppose instead that $Q+N \bmod Q$ is not a power of 2. If also $\lceil \log_2(Q+N \bmod Q) \rceil = \lceil \log_2 Q \rceil$, then $2^{\lceil \log_2 Q \rceil} > Q+N \bmod Q$. Hence, $S^* > 0$. Otherwise, $\lceil \log_2(Q+N \bmod Q) \rceil - \lceil \log_2 Q \rceil = 1$, and

$$S^* = Q + 2^{\lceil \log_2 Q \rceil} - (Q + N \bmod Q) \geq Q - N \bmod Q > 0 . \qquad \square$$

B Characterization of efficient replication networks

See the notation for one-shot networks at the beginning of Appendix A.

Definition B.1 *Let $\langle \mathcal{M}, \mathcal{I} \rangle$ be a replication of a simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$, with task assignments $\{\alpha_\tau\}_{\tau \in \mathcal{T}}$. $\langle \mathcal{M}, \mathcal{I} \rangle$ is said to be a* **cyclic replication** *of $\langle \mathcal{Q}, \mathcal{J} \rangle$ if there is a permutation $r : \mathcal{Q} \to \mathcal{Q}$, called the* **task rotation**, *such that for each task $q \in \mathcal{Q}$, after a manager m finishes task q for a cohort $\tau \in \mathcal{T}$, manager m begins task $r(q)$ for the next cohort in which task $r(q)$ begins, i.e., for cohort $\tau + \lceil (c(q) - b(r(q)))/T \rceil T$.*

Lemma B.1 *If $\langle \mathcal{M}, \mathcal{I} \rangle$ is a replication of a simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$, then there is cyclic replication of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with the same delay and as few managers as $\langle \mathcal{M}, \mathcal{I} \rangle$.*

Proof. We avoid heavy notation, as it would not clarify the simple argument.

Recall from Remark 2.1 that we can reassign tasks in an essential network as long as in the resulting network no manager performs two tasks that are active during the same cycle. Therefore, the minimum number of managers in a replication of a simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$ is equal to the maximum number M of tasks that are active in any cycle.

We can imagine that each manager in a replication network stands in a queue when he is not performing an active task. If there are M managers, then in each cycle t there are always enough managers in the queue to be assigned to the tasks that begin in cycle t. By putting managers in the queue and assigning them tasks in an order that depends only on the tasks, we can ensure that the replication is cyclic.

For example, let the queue be FIFO. Initially, all managers are in this queue in arbitrary order. Order the set $\mathcal{Q}$ of tasks in the one-shot network linearly. In each cycle t, managers who complete tasks in cycle t are added to the queue in the order of the tasks that they complete. Then tasks that begin in cycle t are assigned managers from the queue in the order of the tasks. Once the first cohort has been processed, a steady state in terms of managers who are busy in each cycle, modulo T, is reached. Each time a task $q \in \mathcal{Q}$ is completed, the size of the queue is the same, the tasks that finish in that cycle are the same, and the tasks that begin in subsequent cycles are the same. Hence, after completing task q, the time a manager is idle and the next task to which the manager is assigned is invariant to the identity of the manager and to the cohort for which the task

is completed. Hence, the network is cyclic as long as managers begin their next assigned task for the first cohort in which that task begins. The latter condition holds because M equals the maximum number of tasks that are active in any cycle. For if there were a task q such that each time a manager completed q, he would be idle for T more cycles, then there would never be a cycle in which all managers are performing active tasks. □

Remark B.1 Let $\langle \mathcal{Q}, \mathcal{J} \rangle$ be a simple network and let $\langle \mathcal{M}, \mathcal{I} \rangle$ be a cyclic replication of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with task rotation r. For $q \in \mathcal{Q}$, let $z(q)$ be the number of cycles a manager who finishes task q is idle before beginning task $r(q)$. Then $z(q) = (b(r(q)) - c(q)) \bmod T$. Furthermore, $MT = \sum_{q \in \mathcal{Q}} d(q) + z(q)$.

Proof of Proposition 4.1. By Lemma B.1, we can assume that $\langle \mathcal{M}, \mathcal{I} \rangle$ is a cyclic replication of a simple network $\langle \mathcal{Q}, \mathcal{J} \rangle$, with task rotation r. To simplify notation, write $\mathcal{Q} = \{1, \ldots, Q\}$; these names conflict with those of the data sources, but no confusion should arise during this proof.

According to Lemma 1 in Meagher and Van Zandt (1997), if $\langle \mathcal{Q}, \mathcal{J} \rangle$ contains a task q_1 with a single operation, then task q_1 can be eliminated. That is, there is a network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ such that $\mathcal{Q}' = \mathcal{Q} \setminus \{q_1\}$ and $c'(q) = c(q)$ and $b'(q) = b(q)$ for $q \in \mathcal{Q}'$. If $r(q_1) = q_1$, then the cyclic replication of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with the task rotation $r' \equiv r_{|\mathcal{Q}'}$ has fewer managers than $\langle \mathcal{M}, \mathcal{I} \rangle$ because all the managers who perform task q_1 are eliminated. Otherwise, let q_2 be the task such that $r(q_2) = q_1$. Define the task rotation $r' : \mathcal{Q} \to \mathcal{Q}$ by $r'(q_2) = r(q_1)$ and $r'(q) = r(q)$ for $q \in \mathcal{Q} \setminus \{q_1, q_2\}$, and let $\langle \mathcal{M}', \mathcal{I}' \rangle$ be the cyclic replication of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with task rotation r'. The remaining tasks have the same duration in $\langle \mathcal{M}', \mathcal{I}' \rangle$ as in $\langle \mathcal{M}, \mathcal{I} \rangle$ and managers who are assigned task q_2 pass directly to $r(q_1)$ rather than from q_2 to q_1 to $r(q_1)$. Hence, $\langle \mathcal{M}', \mathcal{I}' \rangle$ has no more managers than $\langle \mathcal{M}, \mathcal{I} \rangle$. Specifically, by Remark B.1, we have

$$MT - M'T = z(q_1) + z(q_2) + d(q_1) - z'(q_2)$$

and

$$\begin{aligned} z'(q_2) &= (b(r(q_1)) - c(q_2)) \bmod T \\ &\leq (b(r(q_1)) - c(q_1)) \bmod T \; + \; (b(q_1) - c(q_2)) \bmod T \\ &\qquad + (c(q_1) - b(q_1)) \bmod T \\ &\leq z(q_1) + z(q_2) + d(q_1) \; . \end{aligned}$$

Therefore, in the rest of this proof we assume that all tasks in $\langle \mathcal{Q}, \mathcal{J} \rangle$ have at least two operations.

$\langle \mathcal{M}, \mathcal{I} \rangle$ is not semi-stationary if and only if the permutation r has a cycle of length greater than 1. Suppose this is true. Then we can renumber the tasks, if necessary, so that there is $K \in \{2, \ldots, Q\}$ such that

$$r(1) = 2, \; r(2) = 3, \; \ldots, \; r(K) = 1$$

and $b(1) \leq b(2)$. Consider three exhaustive cases:

1. $b(2) \geq c(1)$.
2. $(b(2) - c(1)) \bmod T \geq (b(1) - c(1)) \bmod T$.
3. $b(2) < c(1)$ and $(b(2) - c(1)) \bmod T < (b(1) - c(1)) \bmod T$.

In each case, we find a new simple network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with delay D, and a cyclic replication $\langle \mathcal{M}', \mathcal{I}' \rangle$ of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with task rotation r' that has as few managers as $\langle \mathcal{M}, \mathcal{I} \rangle$, such that

$$\#\{q' \in \mathcal{Q}' | r'(q') \neq q'\} < \#\{q \in \mathcal{Q} | r(q) \neq q\} . \tag{B.1}$$

$\langle \mathcal{M}', \mathcal{I}' \rangle$ is semi-stationary if and only if $\#\{q' \in \mathcal{Q}' | r'(q') \neq q'\} = 0$, which must be true after finitely many such transformation.

Case 1. We can apply Lemma 2 in Meagher and Van Zandt (1997), as illustrated in Fig. 2, to concatenate tasks 1 and 2 and thereby construct a network $\langle \mathcal{Q}', \mathcal{J}' \rangle$ that has the same delay and one less operation than $\langle \mathcal{Q}, \mathcal{J} \rangle$, such that $\mathcal{Q}' = \{2, \ldots, Q\}$ and such that:

1. $\forall q \in \mathcal{Q} \setminus \{1,2\}$: $b'(q) = b(q)$, $c'(q) = c(q)$ and $w'(q) = w(q)$.
2. $b'(2) = b(1)$, $c'(2) = c(2)$ and $w'(2) = w(1) + w(2) - 1$.

Let $\langle \mathcal{M}', \mathcal{I}' \rangle$ be a cyclic replication of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with the task rotation r', where r' is like r except that after task K, managers just perform the combined task 2 instead of task 1 and then task 2. That is, $r' : \mathcal{Q}' \to \mathcal{Q}'$ is defined by $r'(q) = r(q)$ for $q \in \mathcal{Q}' \setminus \{K\}$ and $r'(K) = 2$.

It is possible that $b(2) - c(1) \geq T$, in which case a manager who completes task 1 in $\langle \mathcal{M}, \mathcal{I} \rangle$ starts task 2 for an *earlier* cohort, so that he is still idle less than T cycles between tasks 1 and 2. In $\langle \mathcal{Q}', \mathcal{J}' \rangle$, this gap $b(2) - c(1)$ may appear as idle time *within* the combined task q_2. However, by Lemma A.1 we can increase the execution time of some of the instructions in task 2 if necessary so that $c'(2) = c(2)$ and $d'(2) = w'(2) < d(1) + d(2)$. To simplify some calculations below, assume that $d'(2) = d(1) + d(2)$. (If $\langle \mathcal{M}', \mathcal{I}' \rangle$ has no more than M managers under this assumption, then it has no more than M managers when $d'(2) < d(1) + d(2)$.)

Given that $b(q) = b'(q)$ and $c(q) = c'(q)$ for $q' \in \mathcal{Q} \setminus \{1,2\}$ and given the definition of r', we have $d(q) = d'(q)$ for $q \in \mathcal{Q} \setminus \{1,2\}$ and $z(q) = z'(q)$ for $q \in \mathcal{Q} \setminus \{1, K\}$. Since also $d'(2) = d(1) + d(2)$, by Remark B.1 we have

$$MT - M'T = z(1) + z(K) - z'(K) .$$

It follows from $d'(2) = d(1) + d(2)$ that $b'(2) = b(2) - d(1) = b(2) - c(1) + b(1)$. Hence,

$$\begin{aligned} z'(K) &= (b'(2) - c'(K)) \bmod T \\ &= (b(2) - c(1) + b(1) - c(K)) \bmod T \\ &\leq (b(2) - c(1)) \bmod T \; + \; (b(1) - c(K)) \bmod T \\ &= z(1) + z(K) . \end{aligned}$$

Therefore, $MT \geq M'T$ and $\langle \mathcal{M}', \mathcal{I}' \rangle$ has as few managers as $\langle \mathcal{M}, \mathcal{I} \rangle$. Furthermore, $\langle \mathcal{Q}', \mathcal{J}' \rangle$ has one less task than $\langle \mathcal{M}, \mathcal{I} \rangle$, and so (B.1) is satisfied.

Case 2. The time $(b(2) - c(1)) \bmod T$ a manager is idle when passing from task 1 to task 2 is greater than the time $(b(1) - c(1)) \bmod T$ a manager would be idle passing from task 1 to task 1 instead. We can thus weakly improve the network by having managers rotate from task 1 to task 1 and from task K to task 2. This is illustrated in Fig. 9.

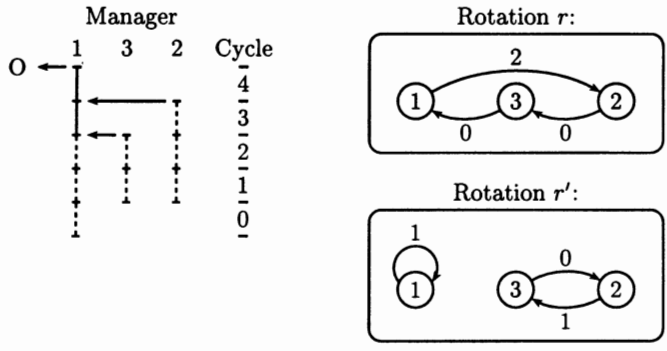


Fig. 9. Separating a task from a rotation cycle, as in case 2 of the proof of Proposition 4.1. The network on the left is the one-shot network that is replicated. The graphs on the right show two task rotations, with the idle between tasks shown on each arc. Task rotation r' has the same idle time as r, but the length of the cycle is less.

Explicitly, let $r'(1) = 1$, $r'(K) = 2$, and $r'(q) = r(q)$ for $q \in \mathcal{Q} \setminus \{1, K\}$. Let $\langle \mathcal{M}', \mathcal{I}' \rangle$ be a cyclic replication of $\langle \mathcal{Q}, \mathcal{J} \rangle$ with task rotation r'. The length of each task is unchanged and $z'(q) = z(q)$ for $q \in \mathcal{Q} \setminus \{1, K\}$. Therefore,

$$\begin{aligned} MT - M'T &= z(1) - z'(1) + z(K) - z'(K) \\ &= (b(2) - c(1)) \bmod T \; - \; (b(1) - c(1)) \bmod T \\ &\quad + (b(1) - c(K)) \bmod T \; - \; (b(2) - c(K)) \bmod T \; . \end{aligned} \tag{B.2}$$

Recall that for arbitrary integers x, y and z, with $z > 0$,

$$((x \bmod z) - (y \bmod z)) \bmod z = (x - y) \bmod z \; .$$

Thus, if

$$(x \bmod z) - (y \bmod z) \geq 0 \; ,$$

then

$$(x \bmod z) - (y \bmod z) = (x - y) \bmod z \; . \tag{B.3}$$

Otherwise,

$$(x \bmod z) - (y \bmod z) \leq 0 \leq (x - y) \bmod z \; . \tag{B.4}$$

From (B.3) and the assumption in this case that $(b(2) - c(1)) \bmod T \geq (b(1) - c(1)) \bmod T$, we have

(B.5)

$$(b(2) - c(1)) \bmod T \; - \; (b(1) - c(1)) \bmod T = (b(2) - b(1)) \bmod T \; .$$

From (B.3) and (B.4), we have

(B.6)

$$(b(2) - b(1)) \bmod T \geq (b(2) - c(K)) \bmod T \; - \; (b(1) - c(K)) \bmod T \; .$$

Therefore, by (B.2), (B.5) and (B.6), $MT - M'T \geq 0$, and $\langle \mathcal{M}', \mathcal{I}' \rangle$ has as few managers as $\langle \mathcal{M}, \mathcal{I} \rangle$. Note also that (B.1) holds.

Case 3. We shift some of task 1's operations to task 2, so that the beginning times of tasks 1 and 2 are interchanged. Then, when we have managers rotate from task 1 to task 1 and from task K to task 2, the idle time following each task is the same as in the original network. This is illustrated in Fig. 10.

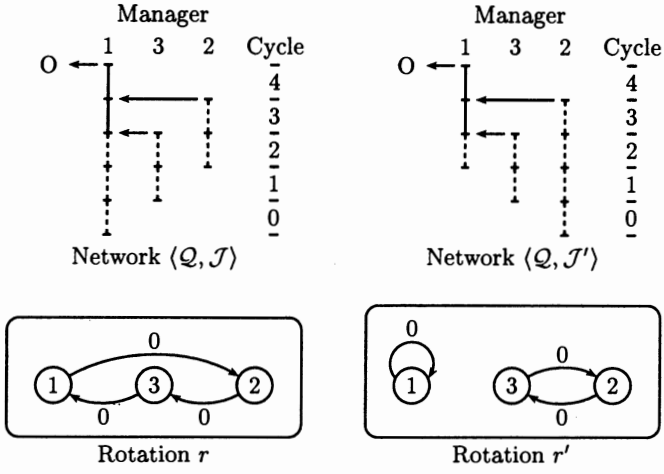


Fig. 10. Shifting operations to reduce a reassignment cycle, as in Case 3 of the proof of Proposition 4.1.

Explicitly, let $\langle \mathcal{Q}, \mathcal{J}' \rangle$ be the network like $\langle \mathcal{Q}, \mathcal{J} \rangle$ but in which task 2 performs the operations performed by 1 in $\langle \mathcal{Q}, \mathcal{J} \rangle$ in cycles $b(1), \ldots, b(2) - 1$, and 2's LOAD in cycle $b(2)$ is changed to ADD and 1's ADD in cycle $b(2)$ is changed to LOAD. This is well-defined because we numbered the tasks so that $b(1) \leq b(2)$ and because in this case $b(1) \neq b(2)$. Task 1 still performs an operation since $c(1) > b(2)$. Thus by Corollary 5.1 in Van Zandt (1997c), $\langle \mathcal{Q}, \mathcal{J}' \rangle$ is essential and has the same delay as $\langle \mathcal{Q}, \mathcal{J} \rangle$. Furthermore, each task begins and finishes at the same time in $\langle \mathcal{Q}, \mathcal{J}' \rangle$ and $\langle \mathcal{Q}, \mathcal{J} \rangle$, except that $b'(1) = b(2)$ and $b'(2) = b(1)$. Let $r'(1) = 1$, $r'(K) = 2$, and $r'(q) = r(q)$ for $q \in \mathcal{Q} \setminus \{1, K\}$. Let $\langle \mathcal{M}', \mathcal{I}' \rangle$ be a cyclic replication of $\langle \mathcal{Q}', \mathcal{J}' \rangle$ with task rotation r'. Now

$$b'(r'(K)) = b'(2) = b(1) = b(r(K)) \quad \text{and} \quad b'(r(1)) = b'(1) = b(2) = b(r(1)) \; ,$$

and thus $z'(q) = z(q)$ for $q \in \mathcal{Q}$. Also, $d'(1) = d(2)$, $d'(2) = d(1)$ and $d'(q) = d(q)$ for $q \in \mathcal{Q} \setminus \{1, 2\}$. Therefore, by Remark B.1, $MT = M'T$, and $\langle \mathcal{M}', \mathcal{I}' \rangle$ has the same number of managers as $\langle \mathcal{M}, \mathcal{I} \rangle$. Note also that (B.1) is satisfied. □

Proof of Proposition 5.1. 1. The idle time for each task in a semi-stationary replication is at most $T-1$ cycles. Therefore, the semi-stationary replication of an efficient one-shot network with delay D has at most

$$\frac{N+Q^*(D,N)-1+Q^*(D,N)(T-1)}{T},$$

managers, and so

$$\text{(B.7)}\quad \frac{M^*(D,N,T)-M_L^*(D,N,T)}{M_L^*(D,N,T)} \le \frac{(T-1)Q^*(D,N)}{N+Q^*(D,N)-1} \le \frac{T-1}{3-2/N}.$$

The second inequality follows from the fact that $Q^*(D,N) \le N/2$. Rearranging (B.7) yields (5.1).

2. Suppose that N, D and T are such that $N/Q^*(D,N) \ge T+1$. Then, compared to the replication of an MS network with $Q^*(D,N)$ tasks, we can add up to $Q^*(D,N)$ tasks, having each one preprocess a multiple of T items, so that no manager with one of the new tasks is idle. Each new task creates at least $T-1$ cycles of slack, because it preprocesses at least T items but creates an extra partial sum to be added by the other managers. The idle time of one of the original tasks can be eliminated by creating at most $T-1$ cycles of slack, and hence it is possible (in fact, optimal) to eliminate all the idle time except perhaps up to $T-1$ cycles by adding at most

$$\frac{Q^*(D,N)(T-1)}{(N/Q^*(D,N))-T}$$

additional tasks. The resulting network has at most $T-1$ idle cycles per cohort, so that

$$\frac{M^*(D,N,T)-M_L^*(D,N,T)}{M_L^*(D,N,T)} \le \frac{\left(\frac{Q^*(D,N)(T-1)}{(N/Q^*(D,N))-T}+T-1\right)}{(N+Q^*(D,N)-1)},$$

which converges to 0 at a rate $O((Q^*(D,N)/N)^2)$ if $N \to \infty$, $D \to \infty$ and $Q^*(D,N)/N \to 0$.

3. Suppose that

$$(N/Q^*(D,N)) - ((N/Q^*(D,N)) \bmod T) \ge \sqrt{N}-1$$

(roughly, $Q^*(D,N) \ge \sqrt{N}$). Then with $(T-1)$ additional tasks we can create at least $(T-1)\sqrt{N}$ manager-cycles of slack, and only $Q^*(D,N)(T-1) \le (T-1)\sqrt{N}$ manager-cycles of slack are needed to eliminate all the idle time of the original tasks. Thus, by adding at most $T-1$ tasks we can eliminate all the idle time except at most $T-1$ manager-cycles, so that

$$\begin{aligned} &M^*(D,N,T) - \lceil M_L^*(D,N,T)\rceil \\ &\quad\le \left\lfloor \frac{N+(Q^*(D,N)+(T-1)-1)+(T-1)}{T} \right\rfloor - \frac{N+Q^*(D,N)-1}{T} \\ &\quad\le \left\lfloor \frac{2(T-1)}{T} \right\rfloor \\ &\quad\le 1. \end{aligned}$$

□

List of symbols

Cohorts

t	Index for cycles ($t = 0, 1, \ldots$).
$\mathcal{T}$	Set of cycles in which cohorts arrive.
τ	Index for cohorts ($\tau \in \mathcal{T}$).
T	Number of cycles between cohorts in periodic mode.
N	Number of items per cohort.
$\mathcal{N}$	Set of data sources ($\mathcal{N} = \{1, \ldots, N\}$).
n	Index for items in a cohort ($n \in \mathcal{N}$).

Networks

$\mathcal{M}/\mathcal{Q}$	Set of managers in a periodic/one-shot network.
m/q	A manager in a periodic/one-shot network.
$\mathcal{I}/\mathcal{J}$	Set of instructions in a periodic/one-shot network.
i	An instruction in a network.
$\mathcal{H}$	Task.

One-shot networks

$b(q)$	Cycle in which manager q begins.
$c(q)$	Cycle in which manager q finishes.
$d(q)$	Duration of q : $d(q) = c(q) - b(q)$.
$w(q)$	Number of operations performed by q.

Replication networks

α_τ	Task assignment for cohort τ in a replication network.
$z(q)$	Idle time after task q in a cycle replication network.
$r(q)$	Task rotation in a cyclic replication network.

Performance

D	Delay of a network ($D \in \mathbb{N}$ or $D : \mathcal{T} \to \mathbb{N}$).
M/Q	Number of managers in a periodic/one-shot network.
W	Number of operations in a one-shot network.
$D^*(Q,N)$	Minimum delay for one-shot networks with Q managers.
$Q^*(D,N)$	Minimum size of one-shot networks with delay D.
$M^*(D,N,T)$	Minimum size of replication networks with delay D.
$M_L^*(D,N,T)$	A lower bound on $M^*(D,N,T)$.

References

Bolton, P., Dewatripont, M. (1994) The firm as a communication network. Qu. J. Econ. 109: 809–839

Keren, M., Levhari, D. (1979) The optimum span of control in a pure hierarchy. Manag. Sci. 11: 1162–1172

Keren, M., Levhari, D. (1983) The internal organization of the firm and the shape of average costs. Bell J. Econ. 14: 474–486

Keren, M., Levhari, D. (1989) Decentralization, aggregation, control loss and costs in a hierarchical model of the firm. J. Econ. Behav. Organ. 11: 213–236

Meagher, K., Van Zandt, T. (1997) Managerial costs for one-shot decentralized information processing. Australian National University and Princeton University

Mount, K., Reiter, S. (1990) A model of computing with human agents. The Center for Mathematical Studies in Economics and Management Science, Discussion Paper No. 890, Northwestern University, Evanston, Illinois

Radner, R. (1993) The organization of decentralized information processing. Econometrica 62: 1109–1146

Radner, R., Van Zandt, T. (1992) Information processing in firms and returns to scale. Ann. Econ. de Statist. 25/26: 265–298

Reiter, S. (1996) Coordination and the structure of firms. Northwestern University

Shirazi, B. A., Hurson, A. R., Kavi, K. M. (eds) (1995) Scheduling and Load Balancing in Parallel and Distributed Systems. IEEE Computer Society Press, Los Alamitos

Van Zandt, T. (1996) Organizations with an endogenous number of information processing agents. In: M. Majumdar (ed.) Organizations with incomplete information. Cambridge University Press, Cambridge

Van Zandt, T. (1997a) Real-time decentralized information processing as a model of organizations with boundedly rational agents. Review of Economic Studies (Forthcoming)

Van Zandt, T. (1997b) Real-time hierarchical resource allocation. Princeton University

Van Zandt, T. (1997c) The scheduling and organization of periodic associative computation: Essential networks, I. Rev. Econ. Design 3: 15–27

Van Zandt, T., Radner, R. (1997) Real-time decentralized information processing and returns to scale. Princeton University and New York University

Zomaya, A. Y. (Ed.). (1996) Parallel and distributed computing handbook. McGraw-Hill, New York