Strong Interaction Fairness via Randomization

Yuh-Jzer Joung*
Department of Information Management
National Taiwan University
Taipei, Taiwan
joung@ccms.ntu.edu.tw

Scott A. Smolka†
Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11794–4400, U.S.A.
sas@cs.sunysb.edu

Abstract

We present Multi, a symmetric, fully distributed, randomized algorithm that, with probability 1, schedules multiparty interactions in a strongly fair manner. To our knowledge, Multi is the first algorithm for strong interaction fairness to appear in the literature. Moreover, the expected time taken by Multi to establish an interaction is a constant not depending on the total number of processes in the system. In this sense, Multi guarantees real-time *response.*

Multi makes no assumptions (other than boundedness) about the time it takes processes to communicate. It thus offers an appealing tonic to the impossibility results of Tsay&Bagrodia and Joung concerning strong interaction fairness in an environment, shared-memory or message-passing, in which processes are deterministic and the communication time is nonnegligible.

Because strong interaction fairness is as strong a fairness condition that one might actually want to impose in practice, our results indicate that randomization may also prove fruitful for other notions of fairness lacking deterministic realizations and requiring real-time response.

1 Introduction

A *multiparty interaction* is a set of I/O actions executed jointly by a number of processes, each of which must be ready to execute its own action for any of the actions in the set to occur. An attempt to participate in an interaction delays a process until all other participants are available. After the actions are executed, the participating processes continue their local computation. Although a relatively new concept, the multiparty interaction has found its way into various distributed programming languages and algebraic models of concurrency. See [10] for a taxonomy of programming languages offering linguistic support for multiparty interaction.

Although multiparty interactions are executed synchronously, the underlying model of communication is usually asynchronous and bipartied. The *multiparty interaction scheduling problem* then is concerned with synchronizing asynchronous processes to satisfy the following requirements: (1) an interaction can be established only if all of its participants are ready (i.e., the interaction is *enabled*), and (2) a process can participate in only one interaction at a time. Moreover, some notion of fairness is typically associated with the implementation to prevent "unfair" computations that favor a particular process or interaction.

Several important fairness notions have been proposed in the literature [1, 2, 3], including: *weak interaction fairness*, where if an interaction is continually enabled, then some of its participants will eventually engage in an interaction; and *strong interaction fairness*, where an interaction that is infinitely often enabled will be established infinitely often. A distinguishing characteristic of weak interaction fairness is that it is much weaker than most known fairness notions, while strong interaction fairness is much stronger.

In general, stronger fairness notions induce more liveness properties, but are also more difficult to implement. Therefore, it is not surprising to see that only weak interaction fairness has been widely implemented (e.g., [16, 14, 4, 13, 11, 17, 9]). It is also not surprising to see that all of these algorithms are asymmetric and deterministic, as weak interaction fairness (and thus strong interaction fairness) has been proven impossible by any symmetric, deterministic, and distributed algorithm [7, 12]. Given that a process decides autonomously when it will attempt an interaction, and at a time that cannot be predicted in advance, strong interaction fairness is still not possible even if the sym-

*Research supported by the National Science Council, Taipei, Taiwan, under Grant NSC 84-2213-E-002-005.

†Research supported in part by NSF Grants CCR-9120995 and CCR-9208585, and AFOSR Grant F49620-93-1-0250.

metry requirement is dropped [18, 8].

Note that these impossibility results do not depend on the type of communication primitives (e.g., message-passing or shared-memory) provided by the underlying execution model. They hold as long as one process's readiness for multiparty interaction can be known by another only through communications, and the time it takes two processes to communicate is non-negligible (but can be finitely bounded).

In the case of CSP communication guard scheduling, a special case of the multiparty interaction scheduling problem where each interaction involves exactly two processes, *randomization* has proven to be an effective technique for coping with the aforementioned impossibility phenomena. For example, the randomized algorithm of Reif and Spirakis [15] is symmetric, weak interaction fair with probability 1, and guarantees real time response: if two processes are continuously willing to interact with each other within a time interval Δ, then they establish an interaction within Δ time with high likelihood, and the expected time for establishment of interaction is constant.

The randomized algorithm of Francez and Rodeh [7] is simpler: a process p_i expresses its willingness to establish an interaction with a process p_j by setting a Boolean variable shared by the two processes; p_i may then need to wait a certain amount of time δ before re-accessing the variable to determine if p_j is likewise interested in the interaction. The authors show that, under the proviso that the time to access a shared variable is negligible compared to δ, the algorithm is weak interaction fair with probability 1. Note, however, that this assumption, combined with the fact that no lower bound on δ is provided, may significantly limit the algorithm's practicality. Furthermore, no strong interaction fairness is claimed by either algorithm.

In this paper, we present Multi, an extension of Francez and Rodeh's randomized algorithm to the multiparty case. We prove that Multi is weak interaction fair with probability 1. We also show that if the transition of a process to a state in which it is ready for interaction is independent of the random draws of the other processes, then, with probability 1, Multi is strong interaction fair. To our knowledge, Multi is the first algorithm for strong interaction fairness to appear in the literature.

We also present a detailed timing analysis of Multi and establish a lower bound on how long a process must wait before re-accessing a shared variable. Consequently, our algorithm can be fine-tuned for optimal performance. Moreover, we show that the expected time to establish an interaction is a constant not depending on the total number of processes in the system. Thus, Multi also guarantees *real-time* response.

Because strong interaction fairness is as strong a fairness condition that one might actually want to impose in practice, our results indicate that randomization may also prove fruitful for other notions of fairness lacking deterministic realizations and requiring real-time response.

The rest of the paper is organized as follows. Section 2 describes the multiparty interaction scheduling problem in a more anthropomorphic setting, known as Committee Coordination. Our randomized algorithm is presented in Section 3 and analyzed in Section 4. Section 5 concludes.

2 The Committee Coordination Problem

The problem of scheduling multiparty interactions in asynchronous systems has been elegantly characterized by Chandy and Misra as one of *Committee Coordination* [5]:

> Professors (cf. processes) in a certain university have organized themselves into committees (cf. interactions) and each committee has a fixed membership roster of one or more professors. From time to time, a professor may decide to attend a committee meeting; it starts waiting and continues to wait until a meeting of a committee of which it is a member is started.

Given that all meetings terminate in finite time, the problem is to devise an algorithm satisfying the following three requirements:

Synchronization: A committee meeting may be started only if *all* members of that committee are waiting.

Exclusion: No two committees meet simultaneously if they have a common member.

Weak Interaction Fairness: If all professors of a committee are waiting, then eventually some professor will attend a committee meeting.

We shall also consider **strong interaction fairness**, i.e., a committee that is infinitely often enabled will be started infinitely often. A committee is *enabled* if every member of the committee is waiting, and is *disabled* otherwise.

The overall behavior of a professor can be described by the state transition diagram of Figure 1, where state T corresponds to thinking, W corresponds to waiting for a committee meeting, and E means that the professor is actively engaged in a meeting.

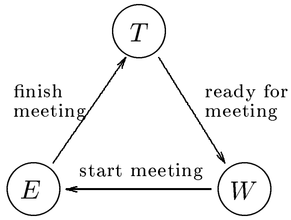

Figure 1: State transition diagram of a professor.

Note that any algorithm for the problem should only control the transition of a professor from state W to state E, but not the other two transitions. That is, the transitions from T to W and from E to T are autonomous to each professor. Moreover, we do not assume any upper bound on the time a professor can spend in thinking. Otherwise, an algorithm for the problem could simply wait long enough until all professors become waiting, and then schedule a committee meeting of its choosing. All three requirements for the problem and strong interaction fairness would then be easily satisfied.

3 The Algorithm

In this section, we present Multi, our randomized algorithm for committee coordination. In the algorithm, we associate with each committee M a counter C_M whose value ranges over $[0 \cdot\cdot\ |\mathit{prof.M}|\ -1]$, where *prof.M* is the set of professors involved in M. C_M can be accessed only by the professors in *prof.M* and only through the TEST&OP instruction as follows:

$$\mathit{result} := \texttt{TEST\&OP}(C_M,\ \mathit{zero\text{-}op},\ \mathit{nonzero\text{-}op})$$

The effect of this instruction is to apply to C_M the operation *zero-op* if its value was zero and the operation *nonzero-op* otherwise, and to assign to the variable *result* the old value (*i.e.*, the value before the operation) of C_M. The operations used here are *no-op*, *inc*, and *dec*, where

$$\mathit{no\text{-}op}(C_M) = C_M,$$
$$\mathit{inc}(C_M) = (C_M + 1) \bmod |\mathit{prof.M}|,$$
$$\mathit{dec}(C_M) = (C_M - 1) \bmod |\mathit{prof.M}|.$$

For example, if $C_M = 2$, then TEST&OP(C_M, *no-op*, *dec*) sets $C_M = 1$ and returns 2. If $C_M = 0$, TEST&OP(C_M, *no-op*, *dec*) keeps C_M unchanged and returns 0. We assume that the execution of the TEST&OP instruction is "atomic," although, in the full version of the paper, we show how this assumption can be removed.

Algorithm Multi can be informally described as follows. Initially, all the shared counters are set to zero. When a professor p_i decides to attend a committee meeting, it randomly chooses a committee M of which it is a member. It then attempts to start a meeting of M by increasing the value of C_M by 1 (all increments and decrements are to be interpreted modulo $|\mathit{prof.M}|$). If the new value of C_M is 0 (i.e., $C_M = |\mathit{prof.M}| - 1$ before the increment), then professor p_i concludes that each of the other members of M have increased C_M by one, and are waiting for p_i to convene M. So p_i goes to state E to start the meeting.

If the new value of C_M is not zero, however, then at least one of the professors in *prof.M* is not yet ready. So professor p_i waits for some period of time (hoping that its partners will become ready) and then re-accesses C_M. If C_M has now been set to 0, then all the professors which were not ready for M are now ready, and so p_i can attend the meeting. If C_M is still not zero, then some professor is still not ready for M. So p_i withdraws its attempt to start M by decreasing the value of C_M by 1 and tries another committee.

```
1.  while waiting do {
2.    randomly choose a committee M from
                                   {M | p_i ∈ prof.M } ;
3.    if TEST&OP(C_M, inc, inc) = |prof.M| − 1
4.      then /* a committee meeting is established */
5.        attend the meeting of M
6.      else { wait δ time ;
7.        if TEST&OP(C_M, no-op, dec) = 0
8.          then /* a committee meeting is established */
9.            attend the meeting of M ;
10.         /* else try another committee */ }
11.   }
```

Figure 2: Algorithm Multi for professor p_i.

The algorithm to be executed by each professor p_i is presented in Figure 2, where *waiting* (line 1) is a Boolean flag indicating whether or not p_i is waiting for a committee meeting. The constant δ (line 6) is the amount of time a professor waits before re-accessing a counter. We will later require (see Section 4) that δ be greater than $\eta_{\max} \times (|\mathrm{prof.}M| - 1)$, where $\eta_{\max}$ is the maximum amount of time a professor can spend in executing lines 2 and 3.[1] Note that the algorithm is *symmetric* in the sense that all professors execute the same code and make no use of their process ids.

[1] More precisely, $\eta_{\max}$ should also include the time it takes to execute line 1. To simplify the analysis, we assume that the Boolean flag *waiting* only serves to indicate the state of the executing professor, and so no explicit test of the flag is needed. Moreover, we assume that an action is instantaneously executed at some time instance. The *time* it takes to execute an action is the difference between the time the action is executed and the time the previous action (of the same professor) was executed.

4 Analysis of the Algorithm

In this section we prove that Multi satisfies the synchronization and exclusion requirements of the Committee Coordination problem, and, with probability 1, is weak and strong interaction fair. We also analyze the expected time Multi takes to schedule a committee meeting.

4.1 Definitions

We assume a discrete global time axis where, to an external observer, the events of the system are totally ordered. Internally, however, processors may execute instructions simultaneously at the same time instance. Simultaneous access to a shared counter will be arbitrated in the implementation of the `TEST&OP` instruction, which we assume is executed atomically.

Since the time axis is discrete, it is meaningless to say that "there are infinitely many time instances in some finite time interval such that ..." Therefore, throughout this paper, the phrase "there are infinitely many time instances" refers to the interval $[0, \infty)$.

For analysis purposes, we present in Figure 3 a refinement of the state transition diagram of Figure 1, where state W is refined into three sub-states $W_0 - W_2$. The actions taken by the professors from these sub-states are:

W_0**:** randomly choose a new committee.

W_1**:** access counter C_M by `TEST&OP`(C_M, *inc*, *inc*).

W_2**:** wait δ time before re-accessing C_M by `TEST&OP`(C_M,*no-op*, *dec*).

We say that a professor is *watching for committee* M if it is in state W_2 waiting for re-access to C_M.

According to the algorithm, if at time t a professor p accesses a counter C_M by `TEST&OP`(C_M, *inc*, *inc*) in state W_1, then it will be in state W_2 or E after t, depending on the value of C_M. Furthermore, if p enters state W_2 at time t to watch for a committee M, then at time $t + \delta$ it will re-access C_M by `TEST&OP`(C_M, *no-op*, *dec*). Depending on the value of C_M, after time $t + \delta$ the professor will either return to state W_0 to choose another committee, or enter state E to attend the meeting of M.

Figure 4 illustrates a possible scenario for four professors p_1, p_2, p_3, and p_4 executing the algorithm, where p_1 and p_4 are involved in committee M_{14}, p_1, p_2, and p_3 are involved in M_{123}, and p_2, p_3 and p_4 are involved in M_{234}. For each professor, we explicitly depict its state (on the Y-axis) at each global time instance (on the X-axis). For example, at time 1 professor p_1 starts waiting for a committee meeting and so it enters state W_0 from state T. At time 2, it randomly chooses

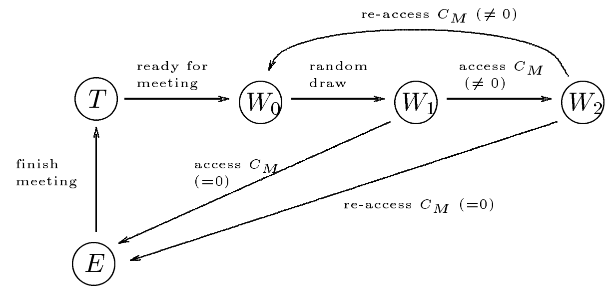

Figure 3: State transition diagram of a professor executing the algorithm.

M_{14} and then accesses $C_{M_{14}}$ at time 3. Since $C_{M_{14}} = 0$ before the access, p_1 enters state W_2 to watch for M_{14} for $\delta = 3$ time units and then re-accesses $C_{M_{14}}$ at time 6. Since p_4 will not access $C_{M_{14}}$ until time 9, p_1 returns to state W_0 to try another committee. Later at time 12, p_1 chooses committee M_{123} and then accesses $C_{M_{123}}$ at time 13. When p_1 re-accesses $C_{M_{123}}$ at time 16, it finds that both p_2 and p_3 are willing to start the meeting of M_{123}. So p_1 enters state E to attend the meeting. The meeting of M_{123} ends at time 19, after which the committee members can return to state T at a time of their own choosing. The shaded area between time 17 and 19 represents a synchronization interval for the three professors.

4.2 Properties of the Algorithm That Hold with Certainty

We now analyze the correctness of the algorithm. We begin with an invariant about the value of a shared counter C_M, which we will use in proving that Multi satisfies the synchronization condition of the Committee Coordination Problem.

Lemma 1 *If at time t there are k professors in state W_2 watching for committee M and no professor, since last entering state W_0, has entered state E to attend a meeting of M, then the value of C_M at time t is k. If, however, at time t some professor has entered state E to attend a meeting of M, then $C_M = 0$ and in $[t, t+\delta]$ all professors in prof.M will have entered state E to attend the meeting of M.*

Proof: By induction on t_i, the time at which the i th system event occurs. □

Theorem 1 (Synchronization) *If a professor in prof.M enters state E at time t to attend a meeting of M, then within δ time all professors in prof.M will have entered state E to attend the meeting of M.*

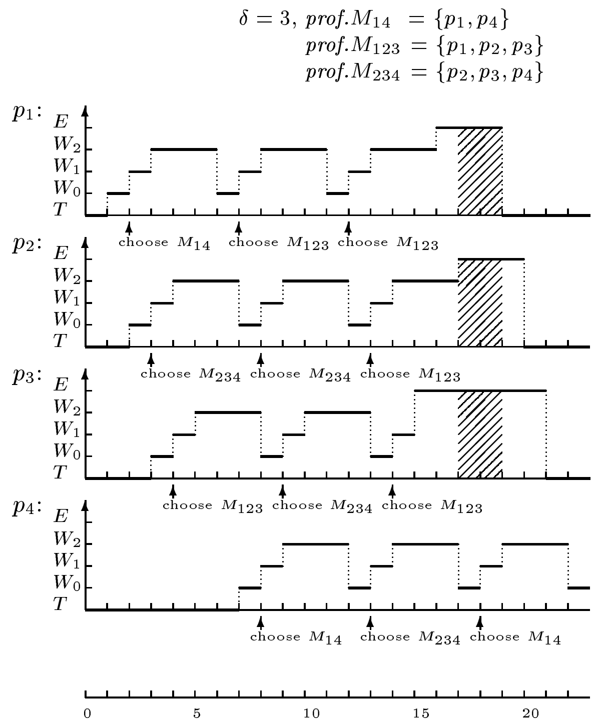

Figure 4: A partial computation of four professors.

Proof: The theorem follows immediately from Lemma 1. □

Theorem 2 (Exclusion) *No two committees convene simultaneously if they have a common member.*

Proof: The result follows from the fact that a professor watches for one committee at a time. □

4.3 Properties of the Algorithm That Hold with Probability 1

We move on to prove that Multi is weak and strong interaction fair, and analyze its time complexity. For this we will need some definitions about the "random draw" a professor performs in state W_0 when deciding which meeting to attempt.

We say that a professor *accesses* a counter C_M when it executes the instruction `TEST&OP`(C_M, *inc*, *inc*) (line 3 of the algorithm) and *re-accesses* C_M when it executes `TEST&OP`(C_M, *no-op*, *dec*) (line 7). Now suppose that professor p accesses some counter in the time interval $(t-\delta, t]$. If there is more than one such access, choose the most recent one. Then the choice of counter must be the result of the random draw performed before the access (line 2). Let $D_{t,p}$ denote this random draw, and let $D_{t,prof.M} = \{D_{t,p} \,|\, p \in prof.M$ and $D_{t,p}$ is defined $\}$.

Note that if p is in state W_2 at time t, then p must have accessed some counter in the interval $(t-\delta, t]$, and so $D_{t,p}$ must be defined. Note further that the interval $(t-\delta, t]$ is open at $t-\delta$ and closed at t. Therefore, $D_{t,p}$ is defined if p accesses C_M at t, but is not defined if the access occurs at $t-\delta$. The reason for choosing this semi-closed interval is to avoid the situation where an access to a counter occurs simultaneously with a re-access to the same counter by another professor.

For example, suppose that counter C_M, shared exclusively by p_1 and p_2, is accessed by p_1 at time $t-\delta$ and found to be zero. Then p_1 must wait δ time before re-accessing C_M at time t. Suppose further that p_2 also chooses M and accesses C_M at time t. We thus have an access/re-access conflict involving C_M at time t, and the meeting is established if and only if the conflict is resolved in favor of the access; i.e., p_2 gets to go first.

Note, however, that no dependency on the resolution order is manifest if p_1's access occurs any time *after* $t-\delta$ (but no later than t). In this case, p_2's access is certain to precede p_1's re-access and the meeting is once again established. Also noteworthy of this case is the fact that both D_{t,p_1} and D_{t,p_2} are defined, while in the case where p_1's access occurs at $t-\delta$, only D_{t,p_2} is defined. In general, if $D_{t,p}$ is defined for all $p \in M$, and these random draws yield the same outcome M, then M will be established (see Lemma 4).

In the rest of this section we shall use $\psi_{p,M}$ to denote the fixed non-zero probability that professor $p \in prof.M$ chooses committee M in a random draw. Thus,

$$\psi_M = \prod_{p \in prof.M} \psi_{p,M}$$

is the probability that a set of mutually independent random draws, one by each professor in *prof.M*, yields the same outcome M.

The following three lemmas are used in the fairness proofs.

Lemma 2 *$\forall t, t'$ such that $t' - t > \delta$, $D_{t,prof.M} \bigcap D_{t',prof.M} = \emptyset$.*

Proof: Follows directly from the definition of $D_{t,p}$. □

Lemma 3 *Suppose that committee M is continuously enabled in the interval $(t_i, t_i + \theta)$. Then there exists a time instance t, $t_i < t \leq t_i + \theta$, such that $|D_{t,prof.M}| = |prof.M|$ if:*

1. *$\theta \geq \eta_{\max} \times |prof.M|$, and*
2. *$\delta > \eta_{\max} \times (|prof.M| - 1)$.*

Proof: Since M is enabled in $(t_i, t_i+\theta)$, every professor of M is in a W-state in this interval. Clearly, either (i) there exists an interval (t_i, t_j), $t_i < t_j \le t_i + \theta$, during which every professor $p \in$ *prof.M* is in state W_2, or (ii) there exists some professor p such that p is not in state W_2 in (t_i, t_j). In case (i), $D_{t,p}$ must be defined for every $p \in$ *prof.M* and every t in (t_i, t_j), and so $|D_{t,prof.M}| = |prof.M|$.

For case (ii), suppose that some professor $p_1 \in$ *prof.M* stays in W_0 or W_1 in $(t_i, t_i + m_1)$ and then accesses some counter at $t_i + m_1$, where $m_1 > 0$. So $D_{t_i+m_1,p_1}$ is defined. Since $m_1 \le \eta_{\max}$ and $|prof.M| \ge 1$, by condition 1, $t_i < t_i + m_1 \le t_i + \theta$.

If $D_{t_i+m_1,p}$ is defined for all $p \in$ *prof.M* (i.e., $|D_{t_i+m_1,prof.M}| = |\mathrm{prof.}M|$), then we are done. Otherwise, $|prof.M|$ must be greater than one, and there exists another professor p_2 such that D_{t,p_2} is not defined for all t in $[t_i + m_1, t_i + m_1 + m_2)$, but D_{t,p_2} is defined for $t = t_i+m_1+m_2$ (similarly because p_2 stays in state W_0 or W_1 in $(t_i+m_1, t_i+m_1+m_2)$, and then accesses some counter at $t_i + m_1 + m_2$). Moreover, $0 < m_2 \le \eta_{\max}$.

By condition 1 and the fact that $|prof.M| \ge 2$, we have $m_1 + m_2 \le \theta$. So p_1 is still in a W-state in $(t_i + m_1, t_i + m_1 + m_2)$. However, since p_1 accesses some counter at t_i+m_1, p_1 must have entered state W_2 after the access, and stays in W_2 in $(t_i+m_1, t_i+m_1+\delta)$. So D_{t,p_1} must be defined for all t in $[t_i+m_1, t_i+m_1+\delta)$. Moreover, by condition 2, $t_i + m_1 < t_i + m_1 + m_2 < t_i + m_1 + \delta$. So $D_{t,p}$ must be defined for both $p = p_1, p_2$, and $t = t_i + m_1 + m_2$. Since $m_1 + m_2 \le \theta$, $t_i < t_i + m_1 + m_2 \le t_i + \theta$.

If $D_{t_i+m_1+m_2,p}$ is also defined for every other professor $p \in$ *prof.M*, then we are done. Otherwise, *prof.M* must contain at least three professors, and there exists a third professor p_3 such that D_{t,p_3} is not defined for all t in $[t_i + m_1 + m_2, t_i + m_1 + m_2 + m_3)$, where $0 < m_3 \le \eta_{\max}$, but D_{t,p_3} is defined for $t = t_i + m_1 + m_2 + m_3$. Moreover, we can use a similar argument to show that D_{t,p_1} and D_{t,p_2} must also be defined for $t = t_i + m_1 + m_2 + m_3$. In general, we can show that if $|prof.M| \ge n$, then D_{t,p_k} must be defined for $k = 1..n$ and $t_i < t = t_i+m_1+m_2+\ldots+m_n \le t_i+\theta$.

Since the number of professors in *prof.M* is finite, eventually we will reach a point where $D_{t,p}$ is defined for each $p \in$ *prof.M* and some $t_i < t \le t_i + \theta$ (see Figure 5). The lemma is then established. □

We shall henceforth refer to Condition 2 of Lemma 3 (the lower bound on δ) as Assumption A1. Note that different professors can choose different values for δ; these values need only satisfy the lower bound established by the lemma. As such, the clocks used by the professors to implement time-outs need not be adjusted to the same accuracy.

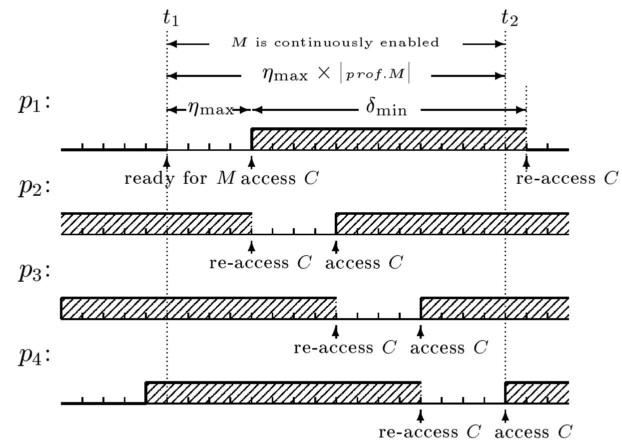

Figure 5: Illustration of Lemma 3. (t_1, t_2) is the maximum possible interval during which M is enabled but $|D_{t,prof.M}| \ne |prof.M|$. Here $|prof.M| = 4$. Then, $D_{t_2,p}$ must be defined for all $p \in$ *prof.M*. Note that if $\delta_{\min}$ would equal $\eta_{\max} \times (|\mathrm{prof.}M| - 1)$, then D_{t_2,p_1} would not be defined.

Lemma 3 says that if a committee M is continuously enabled sufficiently long, then there exists an interval of length θ within which every professor in *prof.M* performs a random draw. The following lemma ensures that if their random draws yield the same outcome, then they must establish M.

Lemma 4 *If $|D_{t,prof.M}| = |prof.M|$ and all the random draws in $D_{t,prof.M}$ yield the same outcome M, then by time t some professor must have already entered state E to start M, and by time $t+\delta$ all professors in prof.M will enter state E to start M.*

Proof: Assume the hypotheses described in the lemma. Let $p_i \in$ *prof.M* be the first professor which, after performing its random draw in $D_{t,prof.M}$, accesses C_M by `TEST&OP`(C_M, *inc*, *inc*), and let $p_j \in$ *prof.M* be the last professor to do so. Let t_i and t_j be the time at which p_i and p_j, respectively, accesses C_M. Then, $t_j - t_i < \delta$, and $t - \delta < t_i \le t_j \le t$.

Since p_i accesses C_M at time t_i, it will not re-access C_M until $t_i + \delta$. Since $t_i + \delta > t$, all professors which access C_M in $[t_i, t_j]$ will remain in state W_2 before p_j accesses C_M. By Lemma 1, $C_M = |prof.M| - 1$ just before p_j's access. So when p_j accesses C_M at t_j, it will set C_M to zero and enter state E to start M. Moreover, by time $t_j + \delta$, every other professor of M will learn that M has been started when it re-accesses C_M by `TEST&OP`(C_M, *no-op*, *dec*), and so will also enter state E to start M. Since $t_j + \delta \le t + \delta$, the lemma is thus established. □

Theorem 3 (Weak Interaction Fairness) *Assume A1 and that all members of a committee M are waiting for committee meetings. Then, the probability is 1 that eventually a meeting involving some member of M will be started.*

Proof: Suppose that M is enabled at t. If M is continuously enabled in $(t, t + \eta_{\max} \times |\text{prof}.M|\,)$, then by Lemma 3 under A1, there exists a time instance t_1, $t < t_1 \leq t + \eta_{\max} \times |\text{prof}.M|$, such that $|D_{t_1,prof.M}| = |prof.M|$. If the random draws in $D_{t_1,prof.M}$ yield the same outcome M, then, by Lemma 4, M must be disabled at t_1. Even if the random draws do not yield the same outcome, some professor of M may still establish another committee meeting M' if its random draw has the outcome M' and at the same time all other professors of M' are also interested in M'. So the probability that the random draws in $D_{t_1,prof.M}$ do not cause any committee involving a member of M to be started is no greater than $1 - \psi_M$.

Similarly, if M remains enabled for another $\eta_{\max} + \delta$ time, then each professor must perform another random draw and access some counter within this interval. So there must exist another time instance t_2 such that $t_1 < t_2 \leq t_1 + \eta_{\max} + \delta$, and $D_{t_2,prof.M}$ contains a completely new set of random draws of size $|prof.M|$. Again, the probability that M remains enabled after these random draws is no greater than $1 - \psi_M$, given that the random draws in $D_{t_1,prof.M}$ do not cause any member of M to attend a meeting. In general, the probability that M remains enabled after i sets of such random draws $D_{t_1,prof.M}$, $D_{t_2,prof.M}$, …, $D_{t_i,prof.M}$ is no greater than $(1 - \psi_M)^i$. As i tends to infinity, $(1 - \psi_M)^i$ tends to zero. So the probability is zero that M remains enabled forever. □

Intuitively, A1 requires that the δ parameter used in the algorithm is large enough so that a professor will not re-access a counter before the other professors get a chance to access the counter. If this assumption is removed from Theorem 3, then a set of professors could access and re-access a counter forever without ever establishing a committee meeting. To illustrate, consider Figure 6. Each professor re-accesses a counter before the other professor could access the same counter. So no matter what committees they choose in their random draws, there just does not exist any time instance at which a professor can see the result of a counter set by the other professor.

The strong interaction fairness property of the algorithm additionally requires the assumption that *a professor's transition from thinking to waiting (see Figure 1) does not depend on the random draws performed by other processes*. We refer to this assumption as A2.

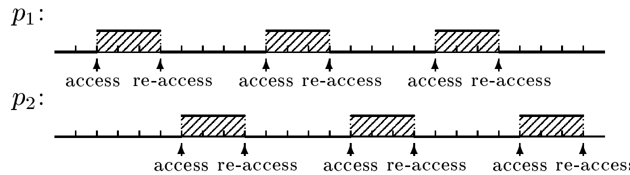

Figure 6: Two professors wait forever without establishing any meeting due to a bad choice of δ.

We also need the following lemma on the probabilistic behavior of a large number of random draws.

Lemma 5 *If there are infinitely many t's such that $|D_{t,prof.M}| = |prof.M|$, then the probability is 1 that all the random draws in $D_{t,prof.M}$ produce the same outcome for infinitely many t's.*

Proof: Let $t_1, t_2, \ldots$ be an infinite sequence of increasing time instances at each of which $|D_{t_i,prof.M}| = |prof.M|$. W.l.o.g. assume that $\forall i, t_{i+1} - t_i > \delta$. By Lemma 2, the sets $D_{t_i,prof.M}$ are pairwise disjoint.

Consider the random draws in set $D_{t_i,prof.M}$. Let E_M denote the event that the random draws in $D_{t_i,prof.M}$ produce the same outcome M. Clearly the probability of E_M's occurrence is independent of the time these random draws are made and is given by ψ_M. Define random variable A_i to be 1 if E_M occurs at t_i, and 0 otherwise. Then, $A_i = 1$ has probability ψ_M, and $A_i = 0$ has probability $1 - \psi_M$.

By the *Law of Large Numbers* (see, for example, [6]), for any ϵ we have

$$\lim_{n \to \infty} \mathrm{P}\left(\left| \frac{\sum_{1 \leq i \leq n} A_i}{n} - \psi_M \right| \leq \epsilon \right) = 1$$

That is, when n tends to infinity, the probability is 1 that $\frac{\sum_{1 \leq i \leq n} A_i}{n}$ tends to ψ_M. Therefore, with probability 1, the set $\{i \,|\, A_i = 1, i \geq 1\}$ is infinite. Hence, with probability 1, there are infinitely many i's such that the random draws in $D_{t_i,prof.M}$ produce the same outcome M. □

Theorem 4 (Strong Interaction Fairness) *Assume A1 and A2. Then, if a committee is enabled infinitely often, with probability 1 the committee will be convened infinitely often.*

Proof: Since the algorithm satisfies weak interaction fairness, we assume that there are infinitely many i's such that M becomes enabled at time instance t_i. By Lemma 3 and A1, either (1) there are infinitely many i's such that each interval $(t_i, t_i + \theta]$, $\theta \geq \eta_{\max} \times |prof.M|$, contains some time instance t

such that $|D_{t,prof.M}| = |prof.M|$, or (2) there are infinitely many i's such that when M becomes enabled at t_i, some professor $q \in prof.M$ attends some meeting in the interval $(t_i, t_i + \theta)$.

Consider Case (1). By Lemma 5, with probability 1 there are infinitely many t's such that all the random draws in $D_{t,prof.M}$ produce the same outcome. So by Lemma 4, with probability 1 M is convened infinitely often.

Consider Case (2). Let $D'_{t_i,q}$ be the last random draw performed by q before it attends some meeting in $(t_i, t_i + \theta)$. By A2, the random draw $D'_{t_i,q}$ is independent of the enabledness of M at time t_i, and so is independent of the other random draws $D'_{t_k,q}$. Since the event that $D'_{t_i,q}$ yields outcome M has a nonzero probability, by the *Law of Large Numbers* (see Lemma 5), the probability is 1 that there are infinitely many i's such that $D'_{t_i,q}$ yields outcome M. Clearly, the meeting that q will attend in $(t_i, t_i + \theta)$ is determined by $D'_{t_i,q}$. So, with probability 1, M is convened infinitely often. □

Note that if Assumption A2 is dropped from Theorem 4, then a conspiracy against strong interaction fairness can be devised. To illustrate, consider a system of two professors p_1 and p_2, and three committees M_1, which involves only p_1, M_2, which involves only p_2, and M_{12}, which involves both p_1 and p_2. Suppose that p_1 becomes waiting, and then tosses a coin to choose either M_1 or M_{12}. The malicious p_2 could stay in thinking until p_1 has selected M_1; then p_2 becomes waiting just before p_1 accesses C_{M_1}. Since p_1 can successfully start M_1 once it selects M_1, M_{12} will not be started if p_1 remains in its meeting while p_2 is waiting. However, M_{12} is enabled as soon as p_2 becomes waiting. So if this scenario is repeated ad infinitum, then the resulting computation would not be strong interaction fair.

The time complexity of the algorithm is analyzed in the following theorem.

Theorem 5 (Time Complexity) *Suppose that from some time onward committee M is enabled. Then, the expected time it takes for any member of M to start a meeting is no greater than*

$$\frac{\delta + \eta_{\max}}{\psi_M} + \eta_{\max} \times (|prof.M| - 1).$$

Proof: Suppose that M is enabled from time t onward. By Lemma 3, there exists a time instance t_1, $t < t_1 \leq t + \eta_{\max} \times |\text{prof.}M|$, such that $|D_{t_1,prof.M}| = |prof.M|$. By Lemma 4, if these random draws yield the same outcome M (an event that occurs with probability ψ_M), then all professors in *prof.M* will start a meeting of M by time $t_1 + \delta$. Otherwise, if some professor's random draw leads to the establishment of some other committee meeting involving the professor, then M will also be disabled by $t_1 + \delta$. If neither of these is the case, then each professor in *prof.M* must perform another random draw and access the selected counter within $\eta_{\max}$ time (from the time it re-accesses the previous selected counter). So there must exist another time instance t_2, $t_1 < t_2 \leq t_1 + \delta + \eta_{\max}$, such that $D_{t_2,prof.M}$ contains a completely new set of random draws, one by each professor in *prof.M*. Once again, if the new random draws yield the same outcome M or cause some other committee meeting to be started, then some professor will enter a meeting by time $t_2 + \delta \leq t_1 + 2\delta + \eta_{\max}$. Otherwise, each professor in *prof.M* will perform another random draw within $\eta_{\max}$ time, and so on.

Therefore, the expected time starting from t_1 until some member of M enters state E to start a meeting is no greater than

$$\sum_i (i \times (\delta + \eta_{\max}) \times (1 - \psi_M)^{i-1} \times \psi_M) \; - \; \eta_{\max}$$

$$= \frac{\delta + \eta_{\max}}{\psi_M} - \eta_{\max}$$

Since $t_1 \leq t + \eta_{\max} \times |\text{prof.}M|$, the expected time for any member of M to start a meeting when M is enabled is no greater than

$$\frac{\delta + \eta_{\max}}{\psi_M} - \eta_{\max} + \eta_{\max} \times |\text{prof.}M|$$

$$= \frac{\delta + \eta_{\max}}{\psi_M} + \eta_{\max} \times (|\text{prof.}M| - 1)$$

□

Note that $\eta_{\max}$ is a constant determined by the size (number of members) of the largest committee; call this value MaxCommSize. ψ_M is a constant determined by the maximum number of committees of which a professor can be a member; call this value MaxNumComm. Finally, δ is a constant determined by $\eta_{\max}$ and MaxCommSize. Therefore, the time complexity of the algorithm is a constant determined by MaxCommSize and MaxNumComm, and is independent of the total number of professors and committees in the system.

5 Conclusions

We have presented Multi, a new randomized algorithm for scheduling multiparty interactions. We have shown that by properly setting the value of δ (the amount of time a process is willing to wait for an interaction to be established), our algorithm is both weak and strong interaction fair with probability 1. Our results hold even if the time it takes to access a shared variable (the communication delay) is nonnegligible. To our knowledge, this makes Multi the first algorithm for strong interaction fairness to appear in the literature.

Strong interaction fairness has been proven impossible by any deterministic algorithm. Our results therefore indicate that randomization is a feasible and efficient countermeasure to such impossibility phenomena. Furthermore, since most known fairness notions are weaker than strong interaction fairness, they too can be implemented via randomization. For example, *strong process fairness* [1], where a process infinitely often ready for an enabled interaction will participate in an interaction infinitely often, is also realized by our algorithm.

Multi is an extension of Francez and Rodeh's randomized algorithm for CSP-like biparty interactions. Francez and Rodeh were able to claim only weak interaction fairness for their algorithm, and then only under the limiting assumption that the communication time is negligible compared to δ. In this case, strong interaction fairness would be possible even in a deterministic setting.

We have also analyzed the time complexity of our algorithm. Like Reif and Spirakis's real-time algorithm [15], the expected time taken by Multi to establish an interaction is a constant not depending on the total number of processes in the system.

Although Multi is presented in a shared-memory model, it can be easily converted to a message-passing algorithm by letting some processes maintain the shared variables, and other processes communicate with them by message passing to obtain the values of these variables. The time to read/write a shared variable then accounts for the time it takes to deliver a message. The δ parameter in Assumption A1 can be properly adjusted to reflect the new communication delay so that both weak and strong interaction fairness notions can still be guaranteed with probability 1.

References

[1] K. Apt, N. Francez, and S. Katz. Appraising fairness in languages for distributed programming. *Distributed Computing*, 2(4):226–241, 1988.

[2] P. Attie, I. Forman, and E. Levy. On fairness as an abstraction for the design of distributed systems. In *Proc. of the 10th Int'l Conf. on Distributed Computing Systems*, pages 150–157, 1990.

[3] P. Attie, N. Francez, and O. Grumberg. Fairness and hyperfairness in multi-party interactions. *Distributed Computing*, 6:245–254, 1993.

[4] R. Bagrodia. Process synchronization: Design and performance evaluation of distributed algorithms. *IEEE Trans. on Software Engineering*, SE-15(9):1053–1065, Sept. 1989.

[5] K. Chandy and J. Misra. *A Foundation of Parallel Program Design.* Addison-Wesley, 1988.

[6] K. Chung. *A Course in Probability Theory.* A Series of Monographs and Textbooks. Academic Press, New York, second edition, 1974.

[7] N. Francez and M. Rodeh. A distributed abstract data type implemented by a probabilistic communication scheme. Technical Report TR-80, IBM Israel Scientific Center, Apr. 1980. Also: *Proc. of the 21st Annual IEEE Symp. on Foundations of Computer Science*, pages 373-379, 1980.

[8] Y.-J. Joung. Characterizing fairness implementability for multiparty interaction. In *Proc. of the 23rd Int'l Colloquium on Automata, Languages and Programming*, 1996 (to appear).

[9] Y.-J. Joung and S.A. Smolka. Coordinating first-order multiparty interactions. *ACM Trans. on Programming Languages and Systems*, 16(3), May 1994.

[10] Y.-J. Joung and S.A. Smolka. A comprehensive study of the complexity of multiparty interaction. *Journal of the ACM.* To appear.

[11] D. Kumar. An implementation of N-party synchronization using tokens. In *Proc. of the 10th Int'l Conf. on Distributed Computing Systems*, pages 320–327, 1990.

[12] D. Lehman and M. Rabin. On the advantage of free choice: A symmetric and fully distributed solution to the dining philosophers problem (extended abstract). In *Proc. of the 8th ACM Symp. on Principles of Programming Languages*, pages 133–138, 1981.

[13] M. Park and M. Kim. A distributed synchronization scheme for fair multi-process handshakes. *Information Processing Letters*, 34:131–138, Apr. 1990.

[14] S. Ramesh. A new and efficient implementation of multiprocess synchronization. In *Proc. Conf. on PARLE, Lecture Notes in Computer Science 259*, pages 387–401, 1987.

[15] J. Reif and P. Spirakis. Real time synchronization of interprocess communications. *ACM Trans. on Programming Languages and Systems*, 6(2):215–238, Apr. 1984.

[16] P. Sistla. Distributed algorithms for ensuring fair interprocess communications. In *Proc. of the 3rd ACM Symp. on Principles of Distributed Computing*, pages 266–277, 1984.

[17] Y.-K. Tsay and R. Bagrodia. A real-time algorithm for fair interprocess synchronization. In *Proc. of the 12th Int'l Conf. on Distributed Computing Systems*, pages 716–723, 1992.

[18] Y.-K. Tsay and R. Bagrodia. Some impossibility results in interprocess synchronization. *Distributed Computing*, 6(4):221–231, 1993.