Probabilistic Methods in Concurrency

Lecture 4

Problems in distributed systems for which only randomized solutions exist

Catuscia Palamidessi
catuscia@lix.polytechnique.fr
www.lix.polytechnique.fr/~catuscia

Page of the course:
www.lix.polytechnique.fr/~catuscia/teaching/Pisa/
(1) The dining philosophers

- Each philosopher needs exactly two forks
- Each fork is shared by exactly two philosophers
- A philosopher can access only one fork at the time
Intended properties of solution

• **Deadlock freedom** (aka progress): if there is a hungry philosopher, a philosopher will eventually eat

• **Starvation freedom**: every hungry philosopher will eventually eat (but we won't consider this property here)

• **Robustness wrt a large class of adversaries**: Adversaries decide who does the next move (schedulers)

• **Fully distributed**: no centralized control or memory

• **Symmetric**:
  - All philosophers run the same code and are in the same initial state
  - The same holds for the forks
Non-existence of a “deterministic” solution

• Lehman and Rabin have shown that there does not exist a “deterministic” (i.e. non-probabilistic) solution to the dining philosophers, satisfying all properties listed in previous slide.

• The proof proceeds by proving that for every possible program we can define an adversary (scheduler) which preserves the initial symmetry

• Note: Francez and Rodeh did propose a “deterministic” solution using CSP. The solution to this apparent contradiction is that CSP cannot be implemented in a fully distributed way
The algorithm of Lehmann and Rabin

1. Think
2. randomly choose fork in \{left, right\} \hspace{1em} \%commit
3. if taken(fork) then goto 3
4. else take(fork)
5. if taken(other(fork)) then \{release(fork); goto 2\}
6. else take(other(fork))
7. eat
8. release(other(fork))
9. release(fork)
10. goto 1
Correctness of the algorithm of Lehmann and Rabin

- **Theorem**: for every fair adversary, if a philosopher becomes hungry, then a philosopher (not necessarily the same) will eventually eat with probability 1.

- **Question**: why the fairness requirement? Can we write a variant of the algorithm which does not require fairness?
(2) The committee coordination problem

• **Description of the problem:** In a certain university, professors have organized themselves into committees. Each committee has a fixed membership roster of two or more professors. From time to time a professor may decide to attend a committee meeting. He then starts waiting and continues to wait until a meeting of a committee in which he is member is established.

• **Requirements:**
  - **Mutual exclusion:** No two committees meet simultaneously if they have a common member.
  - **Weak Interaction Fairness (WIF):** If all professors of a committee are waiting (i.e., the committee meeting is enabled), then eventually some professor will attend a committee meeting (not necessarily the same).
  or
  - **Strong Interaction Fairness (SIF):** A committee meeting that is enabled infinitely often will be established infinitely often.
The committee coordination problem

- **Question:** for which requirement among WIF and SIF do we have a correspondence with the synchronization mechanisms used in process calculi, like (the theory of) CSP and \( \pi \)?
  
  - General case equivalent to multiway synchronization, like the mechanism used in (the Theory of) CSP
  
  - Binary case equivalent to the synchronization among two partners, like in CCS and \( \pi \)
The algorithm of Joung and Smolka

1. while waiting do {
2. randomly choose a committee $M$ ;
3. if TEST&OP$(C_M, \text{inc}, \text{inc}) = n_M - 1$
4. then % a committee meeting is established
5. attend the meeting $M$
6. else { wait $\delta_M$ time ;
7. if TEST&OP$(C_M, \text{no-op}, \text{dec}) = 0$
8. then % a committee meeting is established
9. attend the meeting $M$
10. % else try another committee  }
Correctness of the algorithm

- **Assumption:**
  \[ \delta_M > \max_{\text{prof}}\{\text{time}_{2-3}(M,\text{prof})\} \]

- **Theorem:** if a committee is enabled then a professor will eventually attend a meeting with probability 1 (WIF)

- **Theorem:** if a professor’s transition from thinking to waiting does not depend on the random draws performed by other professors, then a committee meeting which is enabled infinitely often will eventually be established (SIF)
Importance of the assumption on $\delta$

- The assumption on $\delta_M$ is an assumption about the degree of synchronism (in the sense of cooperation) of the system. In Distributed Algorithms there are three models of cooperation:
  1. Partially synchronous
  2. Asynchronous
  3. Synchronous (lockstep)

This assumption corresponds to (2)

- Hence this algorithm would not be suitable for implementing the synchronization mechanism of CSP or CCS in a fully distributed setting, since we need an asynchronous cooperation model.
Algorithm of Joung and Smolka:
Example of a livelock in absence of the assumption on $\delta$

The states at the beginning of Lines 3, 5 and 6 are represented with a filled circle. The states at the beginning of Line 1, 2 and 8 are represented with a white circle. Lines 4 and 7 are never reached.