MPRI – Course on Concurrency

Lecture 12

Probabilistic process calculi

Catuscia Palamidessi
LIX, Ecole Polytechnique
catuscia@lix.polytechnique.fr
www.lix.polytechnique.fr/~catuscia

Page of the course:
http://mpri.master.univ-paris7.fr/C-2-3.html

Plan of the lecture

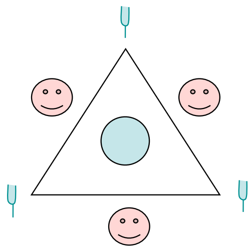
- Motivations
- The power of randomization
  - Problems in distributed systems that can only be solved with the use of randomization
    - Dining Philosophers
- Probabilistic automata
  - Examples
- Probabilistic CCS
  - Operational rules
- Semantics

Motivations

- Modeling: Often a system depends on complex phaenomena whose behavior we cannot know in detail, but for which we can give a quantitative extimation
  - probability as abstraction
  - examples: requests from users
- Expressiveness: Some problems in distributed computing can only be solved by using randomization
  - Dining Philosophers [LR81]
  - Consensus (in presence of components that may fail) [FLP85]
  - Leader Election [B88]

The power of randomization: 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

- Progress: if there is a hungry philosopher, a philosopher will eventually eat
  - Note that we could require something stronger, i.e. Starvation freedom: every hungry philosopher will eventually eat. However we won't consider this property here
- Works for all (fair) schedulers: A scheduler decides who does the next move
- Fully distributed: no centralized control or memory, and no guarded choice
- Symmetry:
  - Symmetry of code: All philosophers run the same code (modulo remaing of channels). Same for the forks.
  - Symmetry of state: All philosophers are in the same initial state. Same for the forks.

Proof that the DP does not have a solution satisfying the intended properties

- The proof is due to Lehmann and Rabin [LR81]
- The idea is the following:
  - Assume by contradicton that there exists a solution
  - Construct a non terminating computation in which Progress does not hold

  - In the "solution", let P1 be the first philosopher making a move
  - The scheduler selects all the other philosophers P2,P3,...,Pn, in turn, and forces them to make the same move. (This is possible by the symmetry of the codes). At the end, the system is in a new state, but still symmetric. So symmetry is preserved by this scheduler
  - The computation cannot pass by a situation in which one philosopher eats, because this would mean that at the end of the previous iteration one philosophers has 2 forks while another has 0 forks.

Proof that the DP does not have a solution satisfying the intended properties

- The proof is due to Lehmann and Rabin [LR81]
- The idea is the following:
  - Assume by contradicton that there exists a solution
  - Construct a non terminating computation in which Progress does not hold

  - In the "solution", let P1 be the first philosopher making a move
  - The scheduler selects all the other philosophers P2,P3,...,Pn, in turn, and forces them to make the same move. (This is possible by the symmetry of the codes). At the end, the system is in a new state, but still symmetric. So symmetry is preserved by this scheduler
  - The computation cannot pass by a situation in which one philosopher eats, because this would mean that at the end of the previous iteration one philosophers has 2 forks while another has 0 forks.

- Question: how does this result combine with the program seen at the beginning of the course? (cfr. first lecture)
- Note: the program seen at the beginning of the course is similar to the algorithm of Francez and Rodek [FR80]

The randomized algorithm of Lehmann and Rabin

1. Think
2. randomly choose fork in {left,right}   %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 randomized algorithm of Lehmann and Rabin

- Lehmann and Rabin proved that their algorithm satisfies Progress (when the scheduler is fair) with probability 1

Bibliography

- [LR81] D.J. Lehmann and M.O. Rabin. The advantages of free choice: a symmetric and fully distributed solution to the dining philosophers problem. In Proceedings of the 8th annual ACM SIGACT-SIGPLAN Symposium on principles of Programming Languages, pages 133--138. 1981.
- [FR80] N. Francez and M. Rodeh A Distributed Abstract Data Type Implemented by a Probabilistic Communication Scheme. Proc. 21st Ann. IEEE Symp. on Foundations of Computer Science. pp. 373-379, 1980.
- [FLP85] Michael J. Fischer, Nancy A. Lynch, Mike Paterson: Impossibility of Distributed Consensus with One Faulty Process J. ACM 32(2): 374-382 (1985)
- [B88] Luc Bougé: On the Existence of Symmetric Algorithms to Find Leaders in Networks of Communicating Sequential Processes. Acta Inf. 25 (2): 179-201 (1988)