2015-03-13 | ../../

Classic Problems in Concurrency

Every student of distributed computing should know quite a bit about the fundamentals of concurrency. A good chunk of that is knowing why the following problems (not necessarily their solutions) are worthy of study.

The classic problems mentioned below are well defined; so do a web search. In this course, we study the Dining Philosophers Problem and Mutual Exclusion Problem. We study the others if time permits.

1 Dining Philosophers Problem

1. Illustrates deadlock
2. Illustrates livelock
3. Illustrates malicious cooperation
4. Assume or not: Communication among them
5. https://en.wikipedia.org/wiki/Dining_philosophers_problem

2 Mutual Exclusion Problem

1. Consider two or more processes. Each P_i has an area of code C_i "sensitive enough" that we call it a "critical section" CS.
2. Assume CS always terminates.
3. "Sensitive enough" == shared variable, usually
4. https://en.wikipedia.org/wiki/Mutual_exclusion

2.1 Mutual Exclusion Problem

1. Find a solution that satisfies four requirements:
    1. R1 Mutual Exclusion: Number of processes in the CS, at any time == 0 or 1.
    2. R2: Deadlock-Free + Live-lock-free
    3. R3: No Unnecessary Delay
    4. R4: Eventual Entry or Bounded Waiting: A process wishing to enter its CS, must be able to enter it in a finite amount of wait.

2.2 Mutual Exclusion Problem

1. Typically solved with semaphores.
    1. m : semaphore := 1

2. Entry to CS: P(m);
3. Exit from CS: V(m);

2.3 Mutual Exclusion Problem

1. Also, read about solutions such as Dekker's and Peterson's, using ordinary variables.

2.4 Mutual Exclusion Problem

1. Two processes can illustrate the essence of the issue, but we are also interested in "starvation-free" semaphore based solutions, especially in distributed systems.
    1. The above solution is not starvation free.
2. Starvation-free Mutex Solutions Using Split Binary Semaphores
    1. ../Semaphores/udding-morris-algs.html

3 Readers-Writers Problem

1. Read-Sharing a resource
2. Exclusive update of the resource
3. Andrews' Notes on Passing The Baton Technique
4. Parnas readers-writers
5. https://en.wikipedia.org/wiki/Readers%E2%80%93writers_problem

4 Producers-Consumers Problem

1. aka Bounded Buffer Problem
2. https://en.wikipedia.org/wiki/Producer%E2%80%93consumer_problem

5 Cigarette Smokers Problem

1. What problems of concurrency can semaphores solve?
2. Introduces an array of semaphores.
3. Cigarette Smokers Problem cannot be solved without arrays.
4. Cigarette Smokers Problem

6 The Drinking Philosophers Problem

1. The Drinking Philosophers Problem is a generalization of the Dining Philosophers Problem.
2. Captures the essence of conflict resolution problems in distributed systems.
3. http://tosummarize.blogspot.com/2008/10/drinking-philosophers-problem.html
4. K M Chandy, J Misra. The Drinking Philosophers Problem. In ACM Transactions on Programming

Languages and Systems, pp 632-646, v 6, No 4, October 1984.

7 Dining Cryptographers Problem

1. https://en.wikipedia.org/wiki/Dining_cryptographers_problem

8 Sleeping Barber Problem

1. https://en.wikipedia.org/wiki/Sleeping_barber_problem

9 End
