Concurrent Object Oriented Languages

Semaphores

wiki.eecs.yorku.ca/course/6490A

Semaphores

A semaphore is a datatype. Its values are nonnegative integers. A semaphore, say s, supports two atomic operations:

- V(s): increment s by 1.
- P(s): decrement s by 1 as soon as the result is nonnegative.

Classical concurrency problems

Using semaphores, we solve the following classical concurrency problems:

- The Producer-Consumer Problem (last lecture)
- The Readers-Writers Problem (already started last lecture)
- The Dining Philosophers Problem
- The Cigarette Smokers Problem

The Readers-Writers Problem

The readers and writers problem, due to Courtois, Heymans and Parnas, is another classical concurrency problem. It models access to a database. There are many competing threads wishing to read from and write to the database. It is acceptable to have multiple threads reading at the same time, but if one thread is writing then no other thread may either read or write. The problem is how do you program the reader and writer threads?

David Parnas

- Canadian early pioneer of software engineering
- Ph.D. from Carnegie Mellon University
- Taught at the University of North Carolina at Chapel Hill, the Technische Universität Darmstadt, the University of Victoria, Queen's University, McMaster University, and University of Limerick
- Won numerous awards including ACM SIGSOFT's "Outstanding Research" award

David Parnas

source: Hubert Baumeister

Pierre-Jacques Courtois

- Professor emeritus at the Catholic University of Leuven

Pierre-Jacques Courtois

source: www.info.ucl.ac.be/~courtois

The Readers-Writers Problem

The readers and writers share the following variable.

```
semaphore mutex = 1;
```

Reader:

```
P(mutex);
read;
V(mutex);
```

Writer:

```
P(mutex);
write;
V(mutex);
```

The Readers-Writers Problem

Question

Does it solve the readers-writers problem?

The Readers-Writers Problem

Question

Does it solve the readers-writers problem?

Answer

Yes!

The Readers-Writers Problem

Question

Does it solve the readers-writers problem?

Answer

Yes!

Question

Is it a satisfactory solution?

The Readers-Writers Problem

Question

Does it solve the readers-writers problem?

Answer

Yes!

Question

Is it a satisfactory solution?

Answer

No!

The Readers-Writers Problem

Question

Why not?

The Readers-Writers Problem

Question

Why not?

Answer

It does not allow multiple readers to read at the same time.

The Readers-Writers Problem

Options

While a writer is writing, readers and writers arrive. Once the writer is done, can the readers start reading?

Options

While readers are reading, readers and writers arrive. Can the readers start reading?

The Readers-Writers Problem

Options

While a writer is writing, readers and writers arrive. Once the writer is done, can the readers start reading?

Yes

Options

While readers are reading, readers and writers arrive. Can the readers start reading?

Yes

No reader is kept waiting unless a writer is writing.

The Readers-Writers Problem

Options

While a writer is writing, readers and writers arrive. Once the writer is done, can the readers start reading?

Options

While readers are reading, readers and writers arrive. Can the readers start reading?

The Readers-Writers Problem

Options

While a writer is writing, readers and writers arrive. Once the writer is done, can the readers start reading?

No

Options

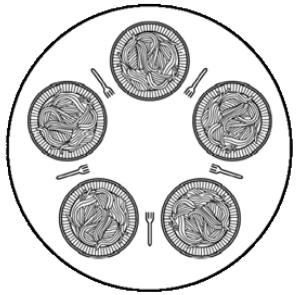
While readers are reading, readers and writers arrive. Can the readers start reading?

No

If a writer wants to write, it writes as soon as possible.

The Dining Philosophers Problem

In the dining philosophers problem, due to Dijkstra, five philosophers are seated around a round table. Each philosopher has a plate of spaghetti. The spaghetti is so slippery that a philosopher needs two forks to eat it. The layout of the table is as follows.



The life of a philosopher consists of alternative periods of eating and thinking. When philosophers get hungry, they try to pick up their left and right fork, one at a time, in either order. If successful in picking up both forks, the philosopher eats for a while, then puts down the forks and continues to think.

The Dining Philosophers Problem

```
int N = 5;

philosopher(i):
while (true)
   think;
   takeForks(i);
   eat;
   putForks(i);

takeForks(i):
   ...
putForks(i):
   ...
```

The Dining Philosophers Problem

```
int N = 5;
semaphore mutex = 1;

philosopher(i):
while (true)
   think;
   takeForks(i);
   eat;
   putForks(i);

takeForks(i):
P(mutex);

putForks(i):
V(mutex);
```

The Dining Philosophers Problem

Question

Is this solution correct?

The Dining Philosophers Problem

Question

Is this solution correct?

Answer

Yes!

The Dining Philosophers Problem

Question

Is this solution correct?

Answer

Yes!

Question

Does this solution allow two philosophers to eat at the same time?

The Dining Philosophers Problem

Question

Is this solution correct?

Answer

Yes!

Question

Does this solution allow two philosophers to eat at the same time?

Answer

No!

The Cigarette Smokers Problem

In the Cigarette Smokers Problem, due to Patil, there are an agent and three smokers. The smokers loop forever, first waiting for ingredients, then making and smoking cigarettes. The ingredients are tobacco, paper, and matches. We assume that the agent has an infinite supply of all three ingredients, and each smoker has an infinite supply of one of the ingredients; that is, one smoker has matches, another has paper, and the third has tobacco.

The agent repeatedly chooses two different ingredients at random and makes them available to the smokers. Depending on which ingredients are chosen, the smoker with the complementary ingredient should pick up both resources and proceed. For example, if the agent puts out tobacco and paper, the smoker with the matches should pick up both ingredients, make a cigarette, and then signal the agent.

The Cigarette Smokers Problem

Question

Can we implement the random choices made by the agent using concurrency?

The Cigarette Smokers Problem

Question

Can we implement the random choices made by the agent using concurrency?

Answer

Yes, we can introduce a thread for each choice and run them concurrently.

The Cigarette Smokers Problem

Question

Can we implement the random choices made by the agent using concurrency?

Answer

Yes, we can introduce a thread for each choice and run them concurrently.

Question

How can we ensure that at most one of those threads executes at any time?

The Cigarette Smokers Problem

Question

Can we implement the random choices made by the agent using concurrency?

Answer

Yes, we can introduce a thread for each choice and run them concurrently.

Question

How can we ensure that at most one of those threads executes at any time?

Answer

Introduce a binary semaphore.

The Cigarette Smokers Problem

Problem

Use semaphores paper and tabacco for the agent who puts out tobacco and paper, signal the smoker with the matches.

The Cigarette Smokers Problem

Shared variables

```
semaphore agent = 1
semaphore match = 0
semaphore paper = 0
semaphore tabacco = 0
```

Agent (paper and tabacco):

```
P(agent); V(tabacco); V(paper)
```

Agent (paper and matches):

```
P(agent); V(paper); V(match)
```

Agent (tabacco and matches):

```
P(agent); V(tabacco); V(match)
```

The Cigarette Smokers Problem

Smoker (paper):

P(tabacco); P(match); V(agent)

Smoker (matches):

P(tabacco); P(paper); V(agent)

Smoker (tabacco):

P(paper); P(match); V(agent)

The Cigarette Smokers Problem

Question

Is this solution correct?

The Cigarette Smokers Problem

Question

Is this solution correct?

Answer

No, because it can deadlock.