Technische Universität München
Institut für Informatik
Dr. Ralf-Peter Mundani
Michael Lieb, M. Sc.

SS 2013

Parallel Programming and HPC

Exercise Sheet 3: Dependency Analysis, Synchronisation, Parallel Structures

1 Dependency Analysis

For the successful parallel execution of processes a dependency analysis is necessary. Here, BERNSTEIN's conditions, for instance, help to identify instruction–level parallelism as to be observed in loops. It is required that an iteration of a loop is independent from all other iterations, i.e. read–write dependencies do not exist. Examine the following code fragment and decide whether instruction–level parallelism might occur or not.

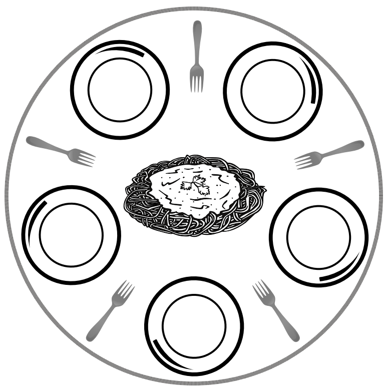
```
for (i = 1; i ≤ n; ++i) {
    a[i] = 2*a[ i + m ] + b[i];
}
```

For which values of m ($-1 \leq m \leq 2$) is a parallel execution of the loop iterations possible?

2 Dining Philosophers Problem

Five philosophers are sitting around a table, doing one of two things: thinking or eating. While thinking, they are not eating, and while eating, they are not thinking. In the centre of the table is placed a big bowl with spaghetti. Furthermore, in front of each philosopher lies a plate and between each plate lies one fork. Thus, each philosopher has one fork to his left side and one fork to his right side. For eating spaghetti, a philosopher is assumed to use two forks, but he can only use the two forks on his immediate left and right side (see figure).

As the philosophers never speak to each other, a synchronisation problem arises due to the shared usage of resources (i.e. the forks). Obviously, this scenario could

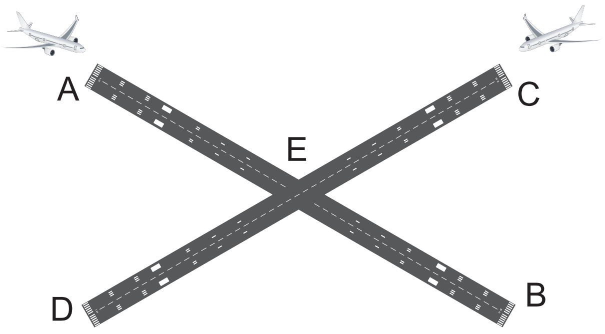


easily lead to a *deadlock* where no philosopher is able to eat or to a *lifelock* where at least some philosophers have to starve (i.e. never manage to get two forks at the same time while others do).

Give a solution for the above synchronisation problem by using lock variables, semaphores, or monitors and discuss your solution!

3 Semaphores

The Hamburg airport has two runways that cross in some point E (see figure below). Arriving airplanes either choose runway AB or runway CD for landing.



In order to prevent two planes from crashing, synchronisation is inevitable. Therefore, the airport administration designs the following rules:

I. No two airplanes can use the same runway at the same time.
II. If an airplane passed point E on its runway, the other runway is open for landing.

You are allowed to use as many binary semaphores as you like, in any point A, B, C, D, E as you like. Give a valid solution for both AB airplanes and CD airplanes with the correct usage of P(σ) and V(σ) operations for your semaphores σ! Discuss if your solution is a fair solution!

4 Deadlocks

In a multicore system 4 processes (threads) P_i are competing for 2 resources R_1 and R_2. Both resources are available two times. In order to finish their computations, the processes have the following request (in that order).

$P_1 \rightarrow R_1,\ P_3 \rightarrow R_2,\ P_4 \rightarrow R_2,\ P_2 \rightarrow R_1,\ P_1 \rightarrow R_2,\ P_3 \rightarrow R_1,\ P_2 \rightarrow R_2$

Assuming that all resources are only exclusively useable, resources cannot be withdrawn from a process, and processes do not release assigned resources while waiting for the allocation of other resources, the above situation might lead to a deadlock.

Draw an allocation/request graph for this scenario and discuss if the above situation is deadlock-free and in what ordering of execution – if possible – processes P_1 to P_4 might finish their computations!