Universitetet i Oslo
Institutt for Informatikk
Research group for Reliable Systems (PSY)
Andrea Pferscher

INF 5170: Models of Concurrency

Fall 2025 **Group Session 4** 26.09.2025

Topic: Channels and Message Passing

Exercise 1 (Partition filter) ([1, Exercise 7.2a]) Consider a filter process `Partition` having the following specifications. `Partition` receives unsorted integer values from one input channel `in` and sends the values it receives to one of two output channels, `out1` or `out2`. `Partition` uses the first value `v` it receives to partition the input values into two sets. It sends all values less than or equal to `v` to `out1` and all values greater than `v` to `out2`. Finally, `Partition` sends `v` to `out1` and sends a sentinel `EOS` to both `out1` and `out2`. The end of the input stream is marked by a sentinel `EOS`.

Develop an implementation of `Partition`. First, give predicates specifying the contents of the channels, then develop the body of `Partition`.

Exercise 2 (Readers/writers & server with asynchronous message passing) ([1, Exercise 7.6]) Consider the readers/writers problem defined in the lecture: There are reader and writer processes that share access to a database/shared data structure. Readers only read from the database and writers write to the database. A writer needs exclusive access to the database, so no other reader or writer should access the database at the same time. On the other hand, many readers can access the database at the same time.

Develop a server process to implement access to the database that the readers and writers access. Show the reader and writer interfaces to the server. The processes should interact using asynchronous message passing.

Exercise 3 (Savings account) ([1, Exercise 7.8]) A savings account is shared by several people. Each person may deposit or withdraw funds from the account. The current balance in the account is the sum of all deposits to date minus the sum of all withdrawals to date. The balance must never become negative.

Develop a server to solve this problem and show the client interface to the server. Clients make two kinds of requests: one to deposit `amount` dollars and one to withdraw `amount` dollars. The withdraw operation must delay until there are sufficient funds. Assume that `amount` is positive. (Remember that we had the same problem for monitors.)

Exercise 4 (Printers) ([1, Exercise 7.10]) Suppose a computer center has two printers, `A` and `B`, that are similar but not identical. Three kinds of client processes use the printers: those that must use `A`, those that must use `B`, and those that can use either `A` or `B`.

Develop code that each kind of client executes to request and release a printer, and develop a server process to allocate the printers. Your solution should be fair assuming that a client using a printer eventually releases it.

References

[1] G. R. Andrews. *Foundations of Multithreaded, Parallel, and Distributed Programming.* Addison-Wesley, 2000.