

Excercise Sheet 6

(To be processed until 04.12.)

Lecture Distributed Systems Winter Term 2025/26

Exercise 1: Programmierung: JNDI

In the lecture two examples of JNDI were given, first the access to the RMI registry and the access to the local file system.

- a) What exactly is JNDI and what is it used for?
- b) Familiarize yourself with the basic use of JNDI. Change your HelloWorld program from the last exercises so that you can access the RMI registry using JNDI.
- c) Also implement and run the second example (“Local File System Access”) from chapter 4.2 of the lecture. Please refer to the jar files provided for download in the archive [u06eFiles.zip](#)¹ on the lecture’s web page and to [this JNDI Tutorial](#) for help.

Exercise 2: Load balancing strategies

- a) How do static, dynamic and preemptive dynamic load balancing differ from each other? Which advantages and disadvantages do the methods have and which prerequisites must be given in each case in order to be able to apply one of the methods?
- b) All load distribution strategies base their decisions on some *load metric*. The simplest, often considered load metric for theoretical analysis of algorithms is the number of (ready) processes per node. Why is this metric sometimes not sufficient in practice? What would be a better metric and what new problems arise then?

Exercise 3: Graph partitioning and list scheduling

In the lecture (chapter 5.1.1) two types of distributed scheduling algorithms based on graphs were presented:

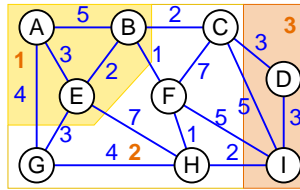
- load distribution through graph partitioning and
- list scheduling.

What similarities and differences can you find between these two classes of algorithms? Under what conditions are they applicable?

Exercise 4: Load Distribution via Graph Partitioning

Remember the load distribution by graph partitioning (see lecture chapter 5.1.1). Assume that in the figure below the process H changes from node 2 to node 3. How large is the resulting (total) network traffic between the nodes?

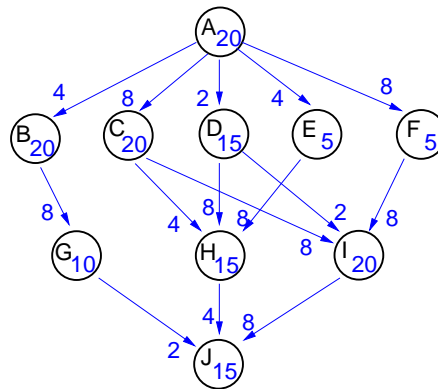
¹<http://www.bs.informatik.uni-siegen.de/web/wismueller/v1/vs/u06eFiles.zip>



Compulsory Exercise 5: List Scheduling
Submit via moodle!

The lecture presented different approaches to distributed scheduling (see section 5.1.1). Two frequently used list scheduling algorithms are: *High Level First with Estimated Time* (HLFET) and *Earliest Task First* (ETF). The first one was already shown in the lecture (see animation in section 5.1.1: List Scheduling with HLFET). For the second algorithm you can find a detailed description in the article by Hagras and Janacek (accessible via the lecture page or via the URL <https://ojs.cvut.cz/ojs/index.php/ap/article/download/490/322>) on page 19.

Given is a system with 3 processors and the DAG shown below, which models a program consisting of several dependent tasks. The nodes contain tasks with execution times and the edges show the necessary communication with transmission time.



How are the tasks scheduled on the three processors (with HLFET / with ETF)? Draw one Gantt chart for each. Suppose a local communication doesn't cost any time.

Compulsory Exercise 6: Programming: Vector Clock
Submit via moodle!

In this exercise, you will use the distributed systems simulator to program a vector clock (see Section 6.3 of the lecture slides).

To do this, complete the task „[Vector Clocks](#)²“ in the wiki (accessible only from within the Uni-VPN!).

²<https://git.bs.informatik.uni-siegen.de/dsbox/exercises/wiki/3-vector-clocks>