**Exam Sequencing and scheduling, 2P450,**
**April 9, 2014, 14:00-17:00**

This test contains four questions on two pages, with 10 sub-items, for which a total of 50 points can be scored. Each good answer scores 5 points. The final grade is obtained by dividing the amount of points by five. Questions may be answered in English and/or in Dutch. Please explain your answers. You are NOT allowed to use the handouts distributed in class/or on the web.

**Question 1.** Consider the use of Lawler's algorithm for solving the problem of minimizing the maximum cost $\max_j f_j(C_j)$ of jobs $j$ scheduled on a single machine.

(a) State the condition on the functions $f_j : \mathbb{R}_+ \to \mathbb{R}$, under which Lawler's rule applies, and state the algorithm.

(b) Consider the special case in which the functions are given by $f_j(C_j) := C_j - d_j$, for $j \in \mathcal{J}$, and where there exist precedence relations between the jobs. Explain which preprocessing steps are needed to find the optimal schedule with a simple rule.

**Question 2.** Consider the problem $1|r_j|L_{\max}$ of minimizing maximum lateness of jobs scheduled on a single machine under release dates. In particular consider the instance given by

| jobs $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $p_j$ | 8 | 18 | 12 | 10 | 10 | 17 | 16 |
| $r_j$ | 0 | 11 | 23 | 29 | 40 | 45 | 70 |
| $d_j$ | 0 | 18 | 13 | 37 | 34 | 78 | 65 |

(a) Formulate and apply an appropriate rule $\rho$ for solving the PREEMPTIVE case. Compute the optimal *preemptive* schedule for the instance given above, and list the results in a table with the following entries:

| jobs $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $C_j^\rho$ | | | | | | | |
| $L_j^\rho$ | | | | | | | |

(b) Verify optimality of the preemptive schedule by providing a *compact proof of optimality* in the form of a subset $S \subseteq \mathcal{J}$, for which $\min_{j \in S} r_j + p(S) - \max_{j \in S} d_j = L_{\max}^\rho$. Explain how you found this set $S$.

(c) State, for this particular instance, an optimal non-preemptive schedule and give an (ad-hoc) reasoning why it should be optimal.

**Question 3.** Consider the problem $(2||\sum_j U_j)$ of minimizing the number of late jobs, on TWO parallel identical machines.

(a) Describe for the general case, how to find the optimal solution, by the use of *Dynamic Programming.* HINT: sort jobs by increasing due date and define $M(i, s, t)$ to be the maximum number of on time jobs $j$ with $1 \leq j \leq i$, with machine load $s$ on machine 1 and machine load $t$ on machine 2; express the correct recursive relations and clarify what you compute to get to the optimal solution.

(b) In particular we may consider the instance given by the following parameters

| jobs $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $p_j$ | 6 | 8 | 5 | 7 | 9 | 5 | 10 |
| $d_j$ | 8 | 10 | 13 | 14 | 17 | 19 | 23 |

Apply the method described in (a) to find the optimal schedule for this instance.

(c) If the number of machines is equal to ONE, what would then be an easy rule for the general case, and what would be the optimal schedule for the instance with the same jobs.

**Question 4.** Consider the problem of scheduling $n$ jobs on $m$ machines, where each job consists of $m$ subtasks, and where for each job $j$ its $i$-th sub-task has to be processed on machine $i$. Here the processing time required is $p_{ij}$. We consider as objective function $\max_j C_{mj}$, where $C_{mj}$ denotes the completion time of the last subtask of job $j$ (hence on machine $m$).

(a) Prove that there exists an optimal schedule in which the jobs are processed on machine 2 in the same order as on machine 1.

(b) For the TWO machine case, describe how an optimal schedule can be found in order of $n \log n$ time.