LUNDS UNIVERSITET
Lunds Tekniska Högskola

Electrical and Information Technology, LTH
Professor Bjorn Landfeldt
Chair of Network Architecture and Services

Take Home Exam, ETS061 Simulation, Summer 2017

There are five problems in the exam. Two of these are related to discrete event simulation and three of them to optimization and meta-heuristic methods. Each problem gives one point. To pass the exam you have to solve at least one of the simulation problems and two of the optimization and heuristics problems. To pass with distinction you have to solve at least four of the problems.
The deadline is September 1, 2017. Upload the solutions to the moodle web site for the course. The solutions should be in the form of a written report. The code should be included in the report, in addition to the results.
Note that this is an individual exam. (No teams!)

Discrete event simulation problems
Here follow the problems in discrete event simulation.

Problem 1 in discrete event simulation
In this problem you shall use antithetic variables to simulate a simple system. Simulate the following queuing system:

- There are two servers
- The service times are uniformly distributed between 0 and 2
- The time between the arrivals are exponentially distributed with mean 1.2 seconds

Find the mean number of customers in the queuing system. Supplement your code to your results.

Problem 2 in discrete event simulation
Assume that you have a random number generator that gives uniformly distributed numbers between 0 and 1. Describe how you can form random numbers with the following distributions with the use of the uniformly distributed numbers.

1. Geometrical distribution, that is you shall get an integer N where $P(N = k) = \alpha^k (1 - \alpha)$ for $k = 0, 1, 2, \ldots$

Address: Box 118, 221 00 Lund *Visiting Address:* Ole Römers Väg 3 *Phone:* +46 46-222 44 07
Email: bjorn.landfeldt@eit.lth.se

We assume that $0 < \alpha < 1$.

2. A real number with the following density function:

$$f(t) = \begin{cases} 0.5t & \text{if } 0 \leq t \leq 1 \\ 0.5 & \text{if } 1 \leq t \leq 2 \\ 1.5 - 0.5t & \text{if } 2 \leq t \leq 3 \\ 0 & \text{otherwise} \end{cases}$$

Optimization and heuristics problems

Here follow the problems in optimization and heuristic methods.

Problem 1 in Optimization

Implement a Branch and Bound (B&B) algorithm for the knapsack problem.

A general-form binary knapsack problem is described as follows:

$$\begin{aligned} max \quad & r_1x_1 + r_2x_2 + ...r_nx_n && (1a)\\ s.t. \quad & w_1x_1 + w_2x_2 + ...w_nx_n \leq W && (1b)\\ & x_1, x_2, ..., x_n \in \{0, 1\}. && (1c) \end{aligned}$$

where W is the knapsack capacity, r_i is the value of object i, w_i is the weight of object i, and x_i is the decision variable indicating whether or not to include object i in the knapsack.

In the implementation you will need to use a procedure for solving the linear relaxation of (1). However, instead of a Matlab (or other software) solver for linear programs, you must implement the following simple procedure to solve the relaxed subproblems encountered in the B&B procedure. First, you should arrange the knapsack objects according to decreasing value-to-weight ratio, then include the consecutive objects, starting from the first, into the knapsack until the knapsack capacity becomes saturated. Note that the last object will be in general included only partially, and in this case, new branches are added to the B&B tree.

Input data: $n, w_1, w_2, ..., w_n, r_1, r_2, ..., r_n, W$, and initial lower bound integer solution (Z^{best}).

Output data: optimal solution, graphical demonstration of the derived B&B tree, and the B&B nodes that are actually visited.

The details and test examples to be discussed individually.
A detailed example of a B&B algorithm for maximization problems is found from slide no. 21 of Lecture-O2. Pay special attention to the fact that the problem at hand is binary, thus you need to create branches by adding x = 0 and x = 1 when x is fractional (not x ≤ 0 and x ≥ 1 as in regular integer programs).

Problem Meta Heuristics 1

Implement a Simulated Annealing program for a production management problem described as follows:

A manufacturing company decides on when to produce a good, how much to produce to satisfy a demand forecast while minimizing its total costs. Below, you find detail information as well as a formulation of the optimization problem.

Constraints:
• Maximum number of goods, G, that can be produced in any period
• Must satisfy a fixed demand vector D over a time length of L periods • Initially, at time t = 0, there is no good in stock
Costs:
• CF : fixed cost of production in periods t ∈ {1, 2, ..., L}
• CP : production cost proportional to the amount of goods produced in periods t ∈ {1, 2, ..., L}
• CS: cost of storage proportional to the amount of goods stored at the end of periods t ∈ {1, 2, ..., L}

Problem formulation

$$\text{minimize} \quad \sum_{t=1}^{L} C_P(t)X(t) + C_F(t)Y(t) + C_S(t)Z(t) \tag{2a}$$

$$\text{subject to:} \quad Z(t-1) + X(t) = D(t) + Z(t) \qquad \forall t \in \{1, ..., L\} \tag{2b}$$

$$X(t) \leq G\,Y(t) \qquad \forall t \in \{1, ..., L\} \tag{2c}$$

$$Z(t) = 0 \qquad \text{for} \quad t = 0 \tag{2d}$$

$$Y(t) \quad \text{is binary,} \quad X(t) \quad \text{and} \quad Z(t) \quad \text{are positive integers} \tag{2e}$$

where, X is the vector representing the amount of goods produced in each period, Y is the decision vector representing whether or not we should produce at any time period, Z is the storage vector representing the amount of goods stored at the end of each time period. D is the demand forecast vector.

Input data: your implemented procedure must work for any given set of parameters as far as the demand vector is feasible and as far as L ≤ 12. A sample input data for validation purpose is as follows:
G = 10, L = 6, and

	t = 1	t = 2	t = 3	t = 4	t = 5	t = 6
C_P	3	4	3	4	4	5
C_F	12	15	30	23	19	45
C_S	1	1	1	1	1	1
D	6	7	4	6	3	8

Note that initially (i.e. at time t = 0) the stock is empty (i.e. $Z(t = 0) = 0$).
Output data: optimal solution in terms of number of goods produced in each period and the resulting total cost. Also indicate the total number of iterations performed by your simulated annealing program.

A general procedure for Simulated Annealing is specified in slide no. 30 of Lecture-H2. You will need to adapt it to the above problem, by designing suitable operators for neighbor generation, evaluation function, neighbor acceptance criteria, temperature schedule, number of iterations per temperature, and the termination criteria.

Problem Meta Heuristics 2

Implement a Tabu Search program for the production management problem described in Problem Meta Heuristics 1.

The problem constraints, costs, formulation, and input data are similar to Problem Meta Heuristics 1. The output will be the optimal solution found by Tabu Search, and the total number of iterations.

A general procedure for Tabu Search is specified in slide no. 55 of Lecture-H2 and a detailed example appears from slide no. 84 onward. You will need to adapt the general procedure to the above problem, by designing suitable neighbor generation operator, Tabu criteria, Aspiration criteria, Tabu memory, Tabu Tenure, and the termination criteria. Designing suitable diversification and intensification mechanisms will be a bonus.