

Project Assignment Two

Production planning

Jesper Larsen David Pisinger

42113 – Fall 2010

Introduction

The project period formally starts on Friday 5 November and the report must be handed in at the latest Tuesday 30 November at 5pm to Jesper Larsen. The report is expected to be appr. 5 pages long excluding tables, figures and appendices. The report must be written in Danish or in English. The project may be solved in groups – permissible group sizes are 1, 2 and 3.

Background

A company is producing a specific product in n variants, J - in the following called *jobs*. The variation stems from that each variant is processed exactly once in a specific sequence on each of m given machines - the set of these is denoted M . In other words, each job i consists of a sequence of m operations o_{ik} , $k = 1, \dots, m$, each with a specific processing time p_{ik} and a specific machine $M(ik) \in M$, on which it is processed. The specific sequence and the processing times vary from job to job.

The total number of operations in the system is $n \times m$, and the operations constitute the set O . Each machine has to process exactly n of these, but cannot process more than one at a time. The optimization problem is now to determine the starting time for each operation on the specific machine corresponding to the operation such that the finishing time for the last operation is as early as possible (this is also known as the makespan, i.e. the difference between finish and start time in a production process).

Question 1 – Formulating the model

For each operation o_{ik} we define a variable t_{ik} , which denotes the starting time for the operation.

- *First, formulate a MIP model for the problem of minimizing the completion time for the operation that finishes last. The model may contain disjunctive constraints, i.e. constraints of the type “(expression 1) or (expression 2)” even though such constraints cannot be used in a standard MIP model.*

- Afterwards, rephrase your disjunctive constraints to ordinary conjunctive constraints by introducing binary variables y_{ikjl} . Such variables are valuable for pairs of operations (o_{ik}, o_{jl}) for which $M(ik) = M(jl)$, and takes the value 1 if and only if operation o_{ik} precedes operation o_{jl} .

Question 2 – Graph Representation

A small test problem with 4 jobs to be processed on 3 machines is given in Table .

	Operation 1	Operation 2	Operation 3
Job 1	Machine 1 - 3	Machine 2 - 7	Machine 3 - 9
Job 2	Machine 2 - 7	Machine 1 - 10	Machine 3 - 2
Job 3	Machine 3 - 3	Machine 2 - 4	Machine 1 - 4
Job 4	Machine 1 - 5	Machine 3 - 5	Machine 2 - 6

Table 1: Sequence and processing time for 4 variants of a product to be processed on 3 machines. For each operation in each variant, the particular machine and the processing time is given.

One way of visualizing the problem is to build a graph, in which each operation corresponds to a vertex. Additionally, the graph has two special vertices, s and t . If operation $o_{i(k+1)}$ is the immediate successor of operation o_{ik} , a directed edge from o_{ik} to $o_{i(k+1)}$ is added in the graph. In addition, there is a directed edge from s to the first operation in each job, and from the last operation in each job to t .

Undirected edges are used to model that a set of operations belongs to a specific machine. Two operations o_{ik} and o_{jl} are connected with an undirected edge if and only if they are to be processed on the same machine. These edges are called *disjunctive edges*. We then have a *clique* for each machine. A clique is a subgraph with n vertices, in which all pairs of vertices are connected by an undirected edge. To describe the sequence, in which operations are processed on a particular machine, all undirected edges in the clique must be given a direction (corresponding to a precedence). A *feasible solution* satisfies that the resulting graph is acyclic, and the solution value - called the *makespan* - corresponds to the *longest path* in the graph starting in s and ending in t . The length of the path is the sum of the processing times of the operations corresponding to the vertices of the path.

- Visualize the test problem, and find a feasible solution to the problem. What is the number of potential solutions to the general problem with n jobs to be processed on m machines (i.e. number of distinct assignments of directions to the disjunctive edges). Can you find a better bound on the number of feasible solutions?

Question 3 – Partial solutions, heads and tails.

In a *partial solution* the sequence of some, but not necessarily all operations, has been fixed for each machine. For each machine-clique, some of the disjunctive edges has been

given a direction. Note that also the situation, in which no disjunctive edges have been given a direction, constitutes a partial solution. The *head* of an operation in a partial solution is defined to be the earliest starting time respecting the sequencing of the partial solution, and likewise, the *tail* is defined to be the minimum time that must pass after the completion of the operation before all jobs might be finished. The head resp. tail of an operation o_{ik} is denoted r_{ik} resp. q_{ik} .

Heads and tails for a given partial solution can be calculated by disregarding all disjunctive edges without direction, and then for each operation o_{ik} calculate the longest path from s to the vertex of o_{ik} , and the longest path from this to t .

- *Explain the modification necessary to Dijkstra's algorithm for solving the single source shortest path problem in a digraph with non-negative edge weights in order to obtain an algorithm for the single source longest path problem in an acyclic network.*
- *What is the complexity of the resulting algorithm?*
- *Apply the algorithm to your feasible solution of the test problem.*

Question 4 – Limiting the search space by using heads and tails

In Branch-and-Bound it is of prime importance to limit the size of the search space. This can be achieved by investigating whether decisions already made in combination with the incumbent rules out the possibility for some particular values for variables.

Consider machine 2 and assume that a solution with makespan 26 has been found.

- *Argue that (job2, operation1) has to be the first operation to be sequenced on machine 2 in a solution with a better makespan, and that given this information, the last operation of such a solution on machine 2 must be (job4, operation3).*
- *Formulate two general conditions which, if satisfied for a given operation, excludes that the operation is first resp. last on the machine corresponding to the operation.*

Question 5 – Exploiting heads and tails - again

The sequence of operations on machine 2 in the test problem is now partially defined - only the sequencing of (job1, operation2) and (job3, operation2) needs to be decided.

- *Compute their respective heads and tails, and use these to determine the sequence.*
- *Formulate a general condition for two operations to be processed on the same machine, which when fulfilled allows the sequence of these to be determined.*

Question 6 – Finding a good feasible solution

In Branch-and-Bound algorithms a good initial solution makes it possible to cut-off parts of the search tree.

- *Formulate a heuristic to produce a good feasible solution for the Job Shop Scheduling problem, and apply the heuristic to the test problem.*

Question 7 – Solving the test problem

- *Solve the test problem by Branch-and-bound, where the lower bound in each node of the search tree is calculated as the longest path from s to t in the graph corresponding to the partial solution in the node. You may use the information that a solution with makespan 26 exist.*

You should use the the conditions derived previously as much as possible to limit the size of the search tree. One way is systematically to consider the machines in turn to see if the conditions imply that some decisions regarding direction of disjunctive edges can be taken. If decisions are taken, heads and tails are re-calculated, and the next machine is tried with the new heads and tails. The process stops when all machines have been considered and no new decisions have been taken.

Question 8 – Gomory cuts

You suddenly come to think about Gomory cuts. Maybe these are applicable? You realize that using CPLEX in interactive mode, it is not possible to extract the updated coefficients in a tableau row, but you can extract the updated coefficients in the objective function (the reduced costs).

- *Is it in theory possible to generate a Gomory cut from the updated coefficients in the objective function? If no, why not? If yes, give the exact derivation of the cut.*