INF-MAT 5360 Mathematical optimization

Compulsory Project 2, Part 2
Completed by November 16, 2010

The compulsory project shall be made and handed in individually (paper version or as a single pdf file). You may discuss the problems with fellow students, but copying other students answers is not permitted; see the general rules for compulsory projects at Ifi. To pass you should at least device the separation algorithm, and show reasonable attempts to implement the dynamic simplex method as explained below.

Consider the 0,1 knapsack problem:

$$\begin{array}{rll} \max & \sum_{i\in N} w_i x_i & \\ s.t. & & \\ (i) & \sum_{i\in N} a_i x_i \leq b; & \\ (ii) & 0 \leq x_i \leq 1; & (i \in N) \\ (iii) & x \in \{0,1\}^N. & \end{array} \tag{1}$$

where $N = \{1, \ldots, n\}$ is the set of items, $w \in \mathbb{R}^n$ is the weight vector and $a \in \mathbb{R}^n$ is the knapsack coefficients vector. In our case we assume a_i to be a **non-negative integer** for all $i \in N$. Denote by $S \subseteq \{0,1\}^n$ the set of solutions satisfying (i), (ii) and (iii).

Constraint (i) is the so called *knapsack constraint*. If we drop the integrality stipulation (iii) on x we obtain a formulation for S, namely the set (polytope) P^1 of the points satisfying (i) and (ii). By solving the associated linear program we can compute an upper bound on the optimal value of the original 0,1 knapsack problem. However, a better formulation (than P^1) can be obtained by considering the so called *cover inequalities*.

A set of items $C \subseteq N$ is a cover of the knapsack if $\sum_{i\in C} a_i > b$.

Consider, for example, the following knapsack constraint:

$$4x_1 + 3x_2 + 3x_3 + 2x_4 \leq 6 \tag{2}$$

where $N = \{1, 2, 3, 4\}$. Then a cover of (2) is $C^1 = \{1, 2\}$, since $a_1 + a_2 = 7 > 6$. Also $C^2 = \{2, 3, 4\}$ is a cover.

A set of items $T \subseteq N$ is a *knapsack solution* if $\sum_{i\in T} a_i \leq b$ (and the set S previously introduced is the set of incidence vectors of all such knapsack solutions).

Since all knapsack coefficients are strictly positive, a knapsack solution T cannot contain a cover C, that is $T \cap C \neq C$. This is equivalent to $|T \cap C| \leq |C| - 1$.

Now, let $x \in S$, that is x is the incidence vector of a knapsack solution T. Then the above condition can be immediately rewritten as a linear inequality in x:

$$\sum_{i \in C} x_i \leq |C| - 1. \tag{3}$$

The above inequality is called *cover inequality* associated with the cover C.

In our example, the cover inequality associated with C^1 is

$$x_1 + x_2 \leq 1,$$

whereas the one associated with C^2 reads as:

$$x_2 + x_3 + x_4 \leq 2.$$

Cover inequalities must be satisfied by the incidence vector of every knapsack solution and thus they can be added to our formulation P^1. Denoting by $\mathcal{C}_{a,b}$ the set of all knapsack covers, we can thus consider the following linear program:

$$\begin{array}{rl} \max & \sum_{i \in N} w_i x_i \\ s.t. & \\ (i) & \sum_{i \in N} a_i x_i \leq b; \\ (ii) & 0 \leq x \leq 1; \\ (iii) & \sum_{i \in C} x_i \leq |C| - 1 \quad (C \in \mathcal{C}_{a,b}). \end{array} \tag{4}$$

Solving (4) instead of (1) will provide, in general, a better upper bound for the optimal solution value to the 0,1 knapsack problem. However, there is typically a very large number of constraints (iii) and they cannot be considered all explicitly. We may instead apply the dynamic simplex method. That is, we start by solving the linear relaxation (P^1) associated to constraints (i) and (ii). We get an optimal solution, say x^1. Then we invoke a separation oracle for constraints (iii), that is an algorithm which either establishes that x^1 satisfies all cover inequalities (and in this case we stop) or finds one such inequality violated by x^1. In the latter case the violated inequality is added to (P^1) to build a new program (P^2) which is then solved to optimality and the method iterates.

The student is required to devise a separation oracle for inequalities of type (iii). As a **hint**, one possible way is to define a suitable 0,1 linear program, much likely the separation oracle for the subtour elimination constraints of the forest polytope (I invite the student to carefully go through the technique described in the course notes and in the lecture notes).

In particular, the feasible solutions $Z \subseteq \{0,1\}^n$ to such 0,1 linear program should be the incidence vectors of all the knapsack covers. Another hint: when writing down the constraints of your program, recall that they should be either equalities, or inequalities of the form $\geq$ or $\leq$ (why?). You may exploit the integrality of the knapsack coefficients to obtain this easily.

Now, we need to find a solution $z \in Z$, that is the incidence vector of a cover $C \subseteq N$, such that the corresponding cover inequality is violated by the current point $\hat{x}$, that is $\sum_{i \in C} \hat{x}_i > |C|-1$. Try to state such condition as a linear constraint in z (recall that in your separation problem z is unknown, while $\hat{x}$ is a given point). Use also the fact that, if z is the incidence vector of C, then $|C| = \sum_{i \in N} z_i$. Then this constraint should suggest a form for a possible objective function of your 0,1 linear program. Finally by solving such 0,1 linear program and by looking at the objective function optimal value you may identify a cover (inequality) violated by $\hat{x}$ or prove that it does not exist.

Once such oracle has been identified (with the corresponding linear programming problem), the student is required to implement both the sequence of programs $(P^1), (P^2), \ldots$ and the (0,1 linear program) separation oracle by using OPL. In principle, a single **.mod** file suffices for all programs $(P^1), (P^2), \ldots$, and only the corresponding **.dat** files must be updated. The same applies to the separation oracle. Monday, November the 2nd, I will post on the web page a **.dat** OPL file named **project2.dat** containing the knapsack weight vector w and the knapsack coefficient vector a. That is, the file will look more or less like this:

Nitems = 4;

w = [5, 3, 5, 2];

a = [4, 3, 3, 2];

Once problem (P^1) is solved to optimality by CPLEX (the LP solver used by OPL), the solution found must be printed out to be used by the separation oracle (it could be included in the corresponding **.dat** file simply by cut-and-paste). The separation oracle must also be implemented using OPL. If the separation

oracle returns a violated inequality, this should be printed out and added to the **.dat** file associated to (P^1) in order to create (P^2). Again this can be done by cut-and-paste. This half-manual procedure must be iterated until either the separation oracle finds no violated inequalities or the student is exhausted (at least 6 iterations).

The student should finally write a report containing:

1. a formal description of the separation oracle for the cover inequalities
2. the single model file associated to the sequence of problems $(P^1), (P^2), \ldots$.
3. the single model file associated to the separation oracle
4. the sequence of optimal solutions, optimal solutions value and violated inequalities associated to problems $(P^1), (P^2), \ldots$.

For a discussion on separation and see our notes on Combinatorial Optimization Section 1.2 and available in the web at

http://heim.ifi.uio.no/ geird/comb_notes.pdf

Good Luck again.