Combinatorial Problem Solving (CPS)

2016-17 Spring Term. Final Exam: 2 hours
Publication of grades: 20/06
Revision of exams: 23/06 (office 113, Ω building, under request by e-mail: `erodri@cs.upc.edu`)

---

1. *(3 pts.)* Let x, y be variables with finite domains $D_x, D_y \subseteq \mathbb{Z}$. Let $C_1(x, y)$ and $C_2(x, y)$ be two constraints.

   (a) *(1.5 pts.)* If at least one of C_1, C_2 is arc-consistent, is it true that the constraint C_3 defined as $C_3(x, y) \equiv C_1(x, y) \vee C_2(x, y)$ is arc-consistent?
   If the answer is positive, prove so. If the answer is negative, give a counterexample.

   True. Without loss of generality, let us assume that C_1 is arc consistent. Let us prove that for any $a \in D_x$ there exists $b \in D_y$ such that $C_3(a, b)$ is true. Indeed: since C_1 is arc-consistent, there exists $b \in D_y$ such that $C_1(a, b)$ is true, and therefore $C_3(a, b) \equiv C_1(a, b) \vee C_2(a, b)$ is true. The same argument can be applied to prove that that for any $a \in D_y$ there exists $b \in D_x$ such that $C_3(b, a)$ is true. Hence, C_3 is arc-consistent.

   (b) *(1.5 pts.)* If C_1 and C_2 are both arc-consistent, is it true that the constraint C_4 defined as $C_4(x, y) \equiv C_1(x, y) \wedge C_2(x, y)$ is arc-consistent?
   If the answer is positive, prove so. If the answer is negative, give a counterexample.

   False. Let us consider domains $D_x = \{0, 1\}$ and $D_y = \{0, 1\}$, and constraints $C_1(x, y) \equiv x = y$ and $C_2(x, y) \equiv x \neq y$.
   We have that:

   i. C_1 is arc-consistent: value 0 for x (for y) has support 0 in D_y (in D_x); and value 1 for x (for y) has support 1 in D_y (in D_x).
   ii. C_2 is arc-consistent: value 0 for x (for y) has support 1 in D_y (in D_x); and value 1 for x (for y) has support 0 in D_y (in D_x).
   iii. C_4 is not arc-consistent, since no pair of values can satisfy it.

2. *(3 pts.)* Let P_0 be a (minimization) linear program over $0-1$ variables $x = (x_1, ..., x_n)$ with cost function $c^T x$. Consider the following simplification of the branch-and-bound procedure for finding the minimum cost of P_0:

   $S := \{P_0\}$
   $UB := \infty$
   while $(S \neq \emptyset)$ $\{$
       $P :=$ *choose_from*(S);
       $S := S - \{P\}$
       $\beta :=$ *basic_solution_of_optimal_basis_of* $(LP(P))$
       if $(\beta \neq \perp \wedge c^T\beta < UB)$ $\{$
           if $(\forall x_i \; \beta(x_i) \in \{0, 1\})$
               $UB := c^T\beta$
           else $\{$
               $x_j :=$ *choose_from*$(\{\, x_i \mid 0 < \beta(x_i) < 1 \,\})$
               $S := S \cup \{P \wedge x_j = 0\} \; \cup \; \{P \wedge x_j = 1\}$
   $\}$ $\}$ $\}$
   return UB

   where:

- function *choose_from*$(\cdot)$ returns an element of a given set;
- function *LP*$(\cdot)$ returns the linear programming relaxation of a given 0-1 linear program;
- function *basic_solution_of_optimal_basis_of* $(\cdot)$ calls the simplex algorithm on a given (real) linear program and returns the basic solution of an optimal basis ($\perp$ if infeasible or unbounded)

(a) *(0.5 pts.)* Can the linear programming relaxation $LP(P)$ at line 6 be an unbounded linear program? Justify your answer.

No, because variables are bounded between 0 and 1. So the cost function is bounded over the feasible solutions of P_0.

(b) *(0.5 pts.)* Does the branch-and-bound procedure terminate? Justify your answer.

Yes. The tree of the search space has finite height because whenever a node is branched, one new variable becomes fixed. This, together with the termination of the simplex algorithm, guarantees the termination of the branch-and-bound procedure.

(c) *(2 pts.)* At any iteration of the loop, the value of variable UB is an *upper* bound on the minimum cost of P_0. Explain the changes you would make to the above pseudo-code so that it also maintained a variable LB giving a *lower* bound on the minimum cost of P_0. *Note:* The grade will depend on how accurate the lower bounds are.

When new 0-1 problems are generated at line 12, their linear programming relaxations are immediately solved and stored in S together with their corresponding 0-1 problems. Then LB has to be the minimum of the optimal solutions of the relaxations of the problems in S.

3. *(4 pts.)* Let us consider $n = p \times q$ Boolean variables

$$\begin{array}{cccc} x_{(0,0)} & x_{(0,1)} & \cdots & x_{(0,q-1)} \\ x_{(1,0)} & x_{(1,1)} & \cdots & x_{(1,q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(p-1,0)} & x_{(p-1,1)} & \cdots & x_{(p-1,q-1)} \end{array}$$

Let $X = \{\, x_{(i,j)} \mid 0 \le i < p,\ 0 \le j < q \,\}$. The *product encoding* of the At-Most-One constraint

$$\text{AMO}(X) \equiv \sum_{0 \le i < p,\ 0 \le j < q} x_{(i,j)} \le 1$$

is defined as follows. Let us introduce auxiliary variables $R = \{r_0, \ldots, r_{p-1}\}$ and $C = \{c_0, \ldots, c_{q-1}\}$ and the following set of clauses:

$$\{\, \neg x_{(i,j)} \vee r_i, \quad \neg x_{(i,j)} \vee c_j \mid 0 \le i < p,\ 0 \le j < q \,\}$$

together with the clauses resulting from encoding AMO(R) and AMO(C) (recursively, or with another encoding for AMO constraints).

(a) *(2 pts.)* Show that, if the encodings used for AMO(R) and AMO(C) are consistent, then the product encoding is consistent.

Let us assume that two different variables $x_{(i,j)}$ and $x_{(i',j')}$ are set to true. Since they are different, either $i \ne i'$ or $j \ne j'$. Without loss of generality, let us assume that $i \ne i'$. Then $x_{(i,j)}$ propagates r_i thanks to clause $\neg x_{(i,j)} \vee r_i$, and $x_{(i',j')}$ propagates $r_{i'}$ thanks to clause $\neg x_{(i',j')} \vee r_{i'}$. Since the encoding for AMO(R) is consistent, unit propagation leads to a conflict.

(b) *(2 pts.)* Show that, if the encodings used for AMO(R) and AMO(C) are arc-consistent, then the product encoding is arc-consistent.

Let us assume that variable $x_{(i,j)}$ is set to true. This propagates r_i thanks to clause $\neg x_{(i,j)} \vee r_i$, and c_j thanks to clause $\neg x_{(i,j)} \vee c_j$. Since the encodings for AMO(R) and AMO(C) are arc-consistent, unit propagation propagates $\neg r_{i'}$ for all $i' \neq i$ and $\neg c_{j'}$ for all $j' \neq j$. But this in turn propagates $\neg x_{(i',j')}$ for all $x_{(i',j')}$ different from $x_{(i,j)}$.
This, together with the previous exercise, justifies that if the encodings used for AMO(R) and AMO(C) are arc-consistent, then the product encoding is arc-consistent.