DM841

Discrete Optimization

Part II

Lecture 8

Constraint Propagation Algorithms for AC

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

AC4

Binary normalized problems – value oriented (fine grained)

function AC4(**in** X: set)**: Boolean** ;
begin
/* initialization */;
$Q \leftarrow \emptyset$; $S[x_j, v_j] = 0, \forall v_j \in D(x_j), \forall x_j \in X$;
foreach $x_i \in X, c_{ij} \in C, v_i \in D(x_i)$ **do**
initialize counter$[x_i, v_i, x_j]$ to $|\{v_j \in D(x_j) \mid (v_i, v_j) \in c_{ij}\}|$;
if *counter*$[x_i, v_i, x_j] = 0$ **then** remove v_i from $D(x_i)$ and add (x_i, v_i) to Q;
add (x_i, v_i) to each $S[x_j, v_j]$ s.t. $(v_i, v_j) \in c_{ij}$;
if $D(x_i) = \emptyset$ **then** return **false** ;
/* propagation */;
while $Q \neq \emptyset$ **do**
select and remove (x_j, v_j) from Q;
foreach $(x_i, v_i) \in S[x_j, v_j]$ **do**
if $v_i \in D(x_i)$ **then**
counter$[x_i, v_i, x_j]$ = counter$[x_i, v_i, x_j] - 1$;
if *counter*$[x_i, v_i, x_j] = 0$ **then**
remove v_i from $D(x_i)$; add (x_i, v_i) to Q;
if $D(x_i) = \emptyset$ **then** return **false** ;
return **true** ;
end

$O(ed^2)$ time
$O(erd^r)$ time for GAC

AC4
Example

$$\mathcal{P} = \langle X = (x, y, z),\ \mathcal{DE} = \{D(x) = D(y) = \{1, 2, 3, 4\}, D(z) = \{3\}\},$$
$$\mathcal{C} = \{C_1 \equiv x \leq y, C_2 \equiv y \neq z\}\}\rangle$$

$\texttt{counter}[x, 1, y] = 4$ $\quad$ $\texttt{counter}[y, 1, x] = 1$ $\quad$ $\texttt{counter}[y, 1, z] = 1$
$\texttt{counter}[x, 2, y] = 3$ $\quad$ $\texttt{counter}[y, 2, x] = 2$ $\quad$ $\texttt{counter}[y, 2, z] = 1$
$\texttt{counter}[x, 3, y] = 2$ $\quad$ $\texttt{counter}[y, 3, x] = 3$ $\quad$ $\texttt{counter}[y, 3, z] = 0$
$\texttt{counter}[x, 4, y] = 1$ $\quad$ $\texttt{counter}[y, 4, x] = 4$ $\quad$ $\texttt{counter}[y, 4, z] = 1$
$\texttt{counter}[z, 3, y] = 3$

$S[x, 1] = \{(y, 1), (y, 2), (y, 3), (y, 4)\}$
$S[x, 2] = \{(y, 2), (y, 3), (y, 4)\}$
$S[x, 3] = \{(y, 3), (y, 4)\}$
$S[x, 4] = \{(y, 4)\}$

$S[y, 1] = \{(x, 1), (z, 3)\}$
$S[y, 2] = \{(x, 1), (x, 2), (z, 3)\}$
$S[y, 3] = \{(x, 1), (x, 2), (x, 3)\}$
$S[y, 4] = \{(x, 1), (x, 2), (x, 3), (x, 4), (z, 3)\}$
$S[z, 3] = \{(y, 1), (y, 2), (y, 4)\}$

AC6

Binary normalized problems

$S[x_j, v_j]$ list of values (x_i, v_i) currently having (x_j, v_j) as their first support

function AC6(**in** X: set)**: Boolean** ;
  begin
    /* initialization */;
    $Q \leftarrow \emptyset$; $S[x_j, v_j] = 0, \forall v_j \in D(x_j), \forall x_j \in X$;
    foreach $x_i \in X, c_{ij} \in C, v_i \in D(x_i)$ **do**
      $v_j \leftarrow$ smallest value in $D(x_j)$ s.t. $(v_i, v_j) \in c_{ij}$;
      if v_j *exists* **then** add (x_i, v_i) to $S[x_j, v_j]$;
      else remove v_i from $D(x_i)$ and add (x_i, v_i) to Q;
      if $D(x_i) = \emptyset$ **then** return **false** ;
    /* propagation */;
    while $Q \neq \emptyset$ **do**
      select and remove (x_j, v_j) from Q;
      foreach $(x_i, v_i) \in S[x_j, v_j]$ **do**
        if $v_i \in D(x_i)$ **then**
          $v'_j \leftarrow$ smallest value in $D(x_j)$ greater than v_j s.t. $(v_i, v_j) \in c_{ij}$;
          if v'_j *exists* **then** add (x_i, v_i) to $S[x_j, v'_j]$;
          else
            remove v_i from $D(x_i)$; add (x_i, v_i) to Q;
            if $D(x_i) = \emptyset$ **then** return **false** ;
    return **true** ;
  end

$O(ed^2)$ time
$O(ed)$ space

AC6
Example

$$\mathcal{P} = \langle X = (x, y, z),\ \mathcal{DE} = \{D(x) = D(y) = \{1, 2, 3, 4\}, D(z) = \{3\}\},$$
$$\mathcal{C} = \{C_1 \equiv x \leq y, C_2 \equiv y \neq z\}\}\rangle$$

$$S[x,1] = \{(y,1),(y,2),(y,3),(y,4)\}$$
$$S[x,2] = \{\}$$
$$S[x,3] = \{\}$$
$$S[x,4] = \{\}$$

$$S[y,1] = \{(x,1),(z,3)\}$$
$$S[y,2] = \{(x,2)\}$$
$$S[y,3] = \{(x,3)\}$$
$$S[y,4] = \{(x,4)\}$$
$$S[z,3] = \{(y,1),(y,2),(y,4)\}$$

Reverse2001

Binary case

function `Revise2001`(**in** x_i: variable; c_{ij}: constraint)**: Boolean** ;
begin
CHANGE $\leftarrow$ **false**;
foreach $v_i \in D(x_i)$ *s.t.* *Last*$(x_i, v_i, x_j) \notin D(x_j)$ **do**
$v_j \leftarrow$ smallest value in $D(x_j)$ greater than `Last`(x_i, v_i, x_j) s.t. $(v_i, v_j) \in c_{ij}$;
if v_j *exists* **then** `Last`$(x_i, v_i, x_j) \leftarrow v_j$;
else
remove v_i from $D(x_i)$;
CHANGE $\leftarrow$ **true**;
return CHANGE ;
end

function `AC3/GAC3`(**in** X: set)**: Boolean** ;
begin
/* initalisation */;
$Q \leftarrow \{(x_i, c) \mid c \in C, x_i \in X(c)\}$;
/* propagation */;
while $Q \neq \emptyset$ **do**
select and remove (x_i, c) from Q;
if *Revise*(x_i, c) **then**
if $D(x_i) = \emptyset$ **then** return **false** ;
else $Q \leftarrow Q \cup \{(x_j, c') \mid c' \in C \wedge c' \neq c \wedge x_i, x_j \in X(c') \wedge j \neq i\}$;
return **true** ;
end

$O(ed^2)$ time
$O(ed)$ space

Reverse2001
Example

$$\mathcal{P} = \langle X = (x, y, z),\ \mathcal{DE} = \{D(x) = D(y) = \{1, 2, 3, 4\}, D(z) = \{3\}\},$$
$$\mathcal{C} = \{C_1 \equiv x \leq y, C_2 \equiv y \neq z\}\}\rangle$$

$\texttt{Last}[x, 1, y] = 1$	$\texttt{Last}[y, 1, x] = 1$	$\texttt{Last}[y, 1, z] = 3$
$\texttt{Last}[x, 2, y] = 2$	$\texttt{Last}[y, 2, x] = 1$	$\texttt{Last}[y, 2, z] = 3$
$\texttt{Last}[x, 3, y] = 3$	$\texttt{Last}[y, 3, x] = 1$	$\texttt{Last}[y, 3, z] = nil$
$\texttt{Last}[x, 4, y] = 4$	$\texttt{Last}[y, 4, x] = 1$	$\texttt{Last}[y, 4, z] = 3$
		$\texttt{Last}[z, 3, y] = 1$

Limitation of Arc Consistency

Example

$$\langle x < y, y < z, z < x; x, y, z \in \{1..100000\}\rangle$$

is inconsistent.

Proof: Apply revise to $(x, x < y)$

$$\langle x < y, y < z, z < x; x \in \{1..99999\}, y, z \in \{1..100000\}\rangle,$$

ecc. we end in a fail.

- Disadvantage: large number of steps.
  Run time depends on the size of the domains!
- Note: we could prove fail by transitivity of $<$.
  $\rightsquigarrow$ Path consitency involves two constraints together