universität innsbruck

Constraint Solving | WS 2022/2023 | LVA 703304

Test-Exam – Solution | January 27, 2023

1 First we replace $\neg\forall$ with $\exists\neg$ and compute an equivalent NNF:

$$\exists x.\,(3y - 1 < 3x \;\wedge\; y \neq 2x - 6)$$

Next we eliminate $\neq$, resulting in the formula

$$\exists x.\,(3y - 1 < 3x \;\wedge\; (y < 2x - 6 \;\vee\; 2x - 6 < y))$$

In the next step we move the terms containing x to one side of the inequalities:

$$\exists x.\,(3y - 1 < 3x \;\wedge\; (y + 6 < 2x \;\vee\; 2x < y + 6))$$

The coefficients of x are 2 and 3, hence we let $\delta' = \mathsf{lcm}\,\{2, 3\} = 6$ and obtain the formula

$$\exists x'.\,(6y - 2 < x' \;\wedge\; (3y + 18 < x' \;\vee\; x' < 3y + 18) \;\wedge\; 6 \,|\, x')$$

The four literals are classified as follows:

(B) $6y - 2 < x'$ (B) $3y + 18 < x'$ (A) $x' < 3y + 18$ (C) $6 \,|\, x'$

At this point we compute the left infinite projection:

$$\bot \;\wedge\; (\bot \;\vee\; \top) \;\wedge\; 6 \,|\, x'$$

which simplifies to $\bot$. The lower bounds in the (B) literals are $B = \{6y - 2, 3y + 18\}$ and $\delta = 6$. Hence the following quantifier-free formula is obtained:

$$\bigvee_{j=1}^{6} \bigvee_{t \in B} (6y - 2 < t + j \;\wedge\; (3y + 18 < t + j \;\vee\; t + j < 3y + 18) \;\wedge\; 6 \,|\, t + j)$$

Since $|A| < |B|$, it is actually more efficient to compute the right infinite projection:

$$\top \;\wedge\; (\top \;\vee\; \bot) \;\wedge\; 6 \,|\, x'$$

which simplifies to $6 \,|\, x'$. The upper bound in the (A) literal is $3y + 18$ and $\delta = 6$, resulting in the following quantifier-free formula:

$$\bigvee_{j=1}^{6} 6 \,|\, -j \;\vee\; \bigvee_{j=1}^{6} (6y - 2 < t - j \;\wedge\; (3y + 18 < t - j \;\vee\; t - j < 3y + 18) \;\wedge\; 6 \,|\, t - j)$$

with $t = 3y + 18$.

2 The constraints are equivalent to

$$4x + 2y \geq -3$$
$$8x + 4y \leq -5$$

So we get the following initial tableau and bounds and an initial assignment where everything becomes 0:

tableau	x	y	bounds	assignment	x	y	s	t
s	4	2	$s \geq -3$		0	0	0	0
t	8	4	$t \leq -5$					

There is a violation for t. Both x and y are suitable, but Bland's rule will select x. Pivoting results in:

tableau	t	y	bounds	assignment	x	y	s	t
s	1/2	0	$s \geq -3$		0	0	0	0
x	1/8	−1/2	$t \leq -5$					

Updating the assignment $t := -5$ results in:

tableau	t	y	bounds	assignment	x	y	s	t
s	$1/2$	0	$s \geq -3$		$-5/8$	0	$-5/2$	-5
x	$1/8$	$-1/2$	$t \leq -5$					

Since all bounds are satisfied, the solution $x = -5/8$ and $y = 0$ is returned.

3 We associate an integer variable x_c with every cell c and use LIA as underlying theory. We consider the constraints separately:

(a) For each room consisting of the n cells $c_1, \ldots, c_n$ we add the constraint

$$\bigwedge_{i=1}^{n} 1 \leqslant x_{c_i} \leqslant n \quad \wedge \quad \sum_{i=1}^{n} x_{c_i} = \tfrac{n(n+1)}{2} \quad \wedge \quad \bigwedge_{i=1}^{n-1} \bigwedge_{j=i+1}^{n} x_{c_i} \neq x_{c_j}$$

(b) For neighbouring cells a and b in adjacent rooms we add the constraint $x_a \neq x_b$. For the given puzzle we obtain 12 such constraints.

(c) A cell with an arrow can have up to four neighbouring cells. Here we consider the case of exactly four neighbours. Let a, b, c and d be these neighbours, and suppose the arrow points to cell a. Then we add

$$a > b \ \wedge \ a > c \ \wedge \ a > d$$

(d) Finally, if cell c is filled with a concrete number n then we simply add the constraint $x_c = n$.

4 (a) Since the inequality $A\vec{x} \leq \vec{b}$ is the same as demanding all row-inequalities we just build a large conjunction, i.e., we apply (2) on each row separately and obtain:

$$\begin{aligned} -1z_1 + 3z_2 &\leq 2 - 4 \\ 2z_1 - 1z_2 &\leq 4 - 3 \\ -1z_1 - 1z_2 &\leq 7 - 2 \end{aligned}$$

or equivalently: $A\vec{z} \leq \vec{b} - \begin{pmatrix} 4 \\ 3 \\ 2 \end{pmatrix}$. This problem can then be solved by the simplex algorithm.

(b) $s = 1/2$ is the smallest possible value of s, because exactly then it is guaranteed that an integral vector $\vec{x}$ is contained in $\mathsf{cube}_s(\vec{z})$. It is the vector $\vec{x}$ defined as $x_i := \mathsf{round}(z_i)$, where round rounds a rational number to the closest integer.

(c) We assume (A) $\vec{x} \in \mathsf{cube}_s(\vec{z})$ and (B) $\vec{a} \cdot \vec{z} \leq c - s\sum_{i=1}^{n} |a_i|$ and have to prove $\vec{a} \cdot \vec{x} \leq c$. This can be done as follows:

$$\begin{aligned} \vec{a} \cdot \vec{x} &= \vec{a} \cdot (\vec{z} + (\vec{x} - \vec{z})) \\ &= \vec{a} \cdot \vec{z} + \vec{a} \cdot (\vec{x} - \vec{z}) \\ &\overset{(B)}{\leq} \left(c - s\sum_{i=1}^{n} |a_i| \right) + \vec{a} \cdot (\vec{x} - \vec{z}) \\ &= c - s\sum_{i=1}^{n} |a_i| + \sum_{i=1}^{n} a_i \cdot (x_i - z_i) \\ &\leq c - s\sum_{i=1}^{n} |a_i| + \sum_{i=1}^{n} |a_i| \cdot |x_i - z_i| \\ &\overset{(A)}{\leq} c - s\sum_{i=1}^{n} |a_i| + \sum_{i=1}^{n} |a_i| \cdot s \\ &= c \end{aligned}$$

5 (1) Yes, this can happen. Consider the weighted graph

$$a \xrightarrow{-4} b \xrightarrow{1} c \xrightarrow{1} d$$

(2) No, this cannot happen. The reason is that if they were pending distance updates in iteration $|V| - 1$, then this leads to a negative-cycle detection in the modified algorithm.

(3) No, this cannot happen. If there is a negative cycle, then in every iteration there would be distance updates. So, in particular when performing the cyclicity-check in the modified algorithm.

6 (a) A suitable invariant is $i \geq 1$. We obtain the following formulas:

$$\begin{aligned} i = 1 &\longrightarrow i \geq 1 && \text{(loop start)} \\ i < N \wedge i \geq 1 \wedge i' = i + 1 &\longrightarrow i' \geq 1 && \text{(loop iteration)} \\ i < N \wedge i \geq 1 &\longrightarrow 0 \leq i + 1 \leq N \wedge 0 \leq i \leq N \wedge 0 \leq i - 1 \leq N && \text{(array accessess)} \end{aligned}$$

(b) We first convert the formula $\neg\psi$ into NNF:

$$\varphi(a, i) \wedge i < N \wedge b = a\{i + 1 \leftarrow a[i] + a[i - 1]\} \wedge j = i + 1 \wedge \big(\exists k.\, 0 \leq k \leq j \wedge b[k] \neq \mathit{fib}(k)\big)$$

Next we eliminate the array updates via the write rule:

$$\begin{aligned} &\varphi(a, i) \wedge i < N \wedge b[i + 1] = a[i] + a[i - 1] \wedge \big(\forall k.\, k \leq i \vee k \geq i + 2 \longrightarrow b[k] = a[k]\big) \\ &\wedge j = i + 1 \wedge \big(\exists k.\, 0 \leq k \leq j \wedge b[k] \neq \mathit{fib}(k)\big) \end{aligned}$$

Next we eliminate the existential quantifier:

$$\begin{aligned} &\varphi(a, i) \wedge i < N \wedge b[i + 1] = a[i] + a[i - 1] \wedge \big(\forall k.\, k \leq i \vee k \geq i + 2 \longrightarrow b[k] = a[k]\big) \\ &\wedge j = i + 1 \wedge 0 \leq \ell \leq j \wedge b[\ell] \neq \mathit{fib}(\ell) \end{aligned}$$

Next we eliminate the universal quantifier where $\mathcal{I} = \{0, i - 1, i, i + 1, i + 2, \ell\}$:

$$\begin{aligned} &\left(\bigwedge_{k \in \mathcal{I}} 0 \leq k \leq i \longrightarrow a[k] = \mathit{fib}(k)\right) \wedge i < N \wedge b[i + 1] = a[i] + a[i - 1] \\ &\wedge \left(\bigwedge_{k \in \mathcal{I}} k \leq i \vee k \geq i + 2 \longrightarrow b[k] = a[k]\right) \wedge j = i + 1 \wedge 0 \leq \ell \leq j \wedge b[\ell] \neq \mathit{fib}(\ell) \end{aligned}$$

Finally, we eliminate array accesses to obtain the desired formula χ:

$$\begin{aligned} \chi := &\Big(\bigwedge_{k \in \mathcal{I}} 0 \leq k \leq i \longrightarrow A(k) = \mathit{fib}(k)\Big) \wedge i < N \wedge B(i + 1) = A(i) + A(i - 1) \\ &\wedge \left(\bigwedge_{k \in \mathcal{I}} k \leq i \vee k \geq i + 2 \longrightarrow B(k) = A(k)\right) \wedge j = i + 1 \wedge 0 \leq \ell \leq j \wedge B(\ell) \neq \mathit{fib}(\ell) \end{aligned}$$