5.298 smooth

DESCRIPTION LINKS GRAPH AUTOMATON

Origin Derived from `change`.

Constraint `smooth(NCHANGE, TOLERANCE, VARIABLES)`

Arguments

NCHANGE	:	dvar
TOLERANCE	:	int
VARIABLES	:	collection(var−dvar)

Restrictions

NCHANGE ≥ 0
NCHANGE $<$ |VARIABLES|
TOLERANCE ≥ 0
required(VARIABLES, var)

Purpose NCHANGE is the number of times that $|X - Y| >$ TOLERANCE holds; X and Y correspond to consecutive variables of the collection VARIABLES.

Example $(1, 2, \langle 1, 3, 4, 5, 2\rangle)$

In the example we have one change between values 5 and 2 since the difference in absolute value is greater than the tolerance (i.e., $|5 - 2| > 2$). Consequently the NCHANGE argument is fixed to 1 and the `smooth` constraint holds.

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Usage This constraint is useful for the following problems:

- Assume that VARIABLES corresponds to the number of people that work on consecutive weeks. One may not normally increase or decrease too drastically the number of people from one week to the next week. With the `smooth` constraint you can state a limit on the number of drastic changes.
- Assume you have to produce a set of orders, each order having a specific attribute. You want to generate the orders in such a way that there is not a too big difference between the values of the attributes of two consecutive orders. If you can't achieve this on two given specific orders, this would imply a set-up or a cost. Again, with the `smooth` constraint, you can control this kind of drastic changes.

Algorithm A first incomplete algorithm is described in [29]. The sketch of a filtering algorithm for the conjunction of the `smooth` and the `stretch` constraints based on dynamic programming achieving arc-consistency is mentioned by Lars Hellsten in [184, page 60]. An arc-consistency algorithm in linear time of the sum of domain sizes is described in [47].

See also

common keyword: `change` *(number of changes in a sequence with respect to a binary constraint)*.

related: `distance`.

Keywords

characteristic of a constraint: automaton, automaton with counters, non-deterministic automaton, non-deterministic automaton.

constraint network structure: sliding cyclic(1) constraint network(2), Berge-acyclic constraint network.

constraint type: timetabling constraint.

filtering: dynamic programming, arc-consistency.

modelling: number of changes.

modelling exercises: n-Amazon.

puzzles: n-Amazon.

Arc input(s)	VARIABLES
Arc generator	$PATH$ ↦ collection(variables1, variables2)
Arc arity	2
Arc constraint(s)	abs(variables1.var − variables2.var) > TOLERANCE
Graph property(ies)	**NARC** = NCHANGE

Graph model Parts (A) and (B) of Figure 5.536 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NARC** graph property, the unique arc of the final graph is stressed in bold.

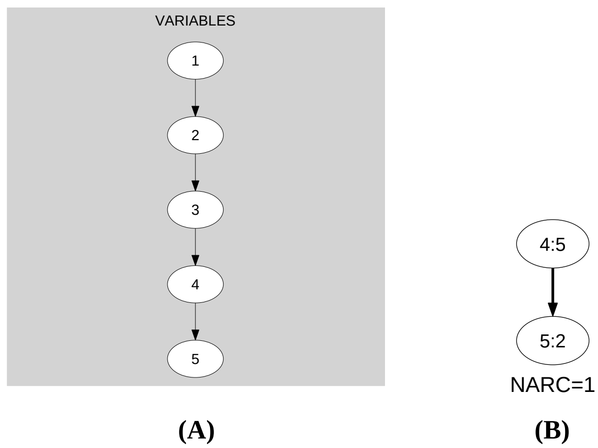

Figure 5.536: Initial and final graph of the smooth constraint

Automaton Figure 5.537 depicts a first automaton that only accepts all the solutions of the `smooth` constraint. This automaton uses a counter in order to record the number of satisfied constraints of the form $(|\texttt{VAR}_i - \texttt{VAR}_{i+1}|) > \texttt{TOLERANCE}$ already encountered. To each pair of consecutive variables $(\texttt{VAR}_i, \texttt{VAR}_{i+1})$ of the collection `VARIABLES` corresponds a 0-1 signature variable $\texttt{S}_i$. The following signature constraint links $\texttt{VAR}_i$, $\texttt{VAR}_{i+1}$ and $\texttt{S}_i$: $(|\texttt{VAR}_i - \texttt{VAR}_{i+1}|) > \texttt{TOLERANCE} \Leftrightarrow \texttt{S}_i = 1$.

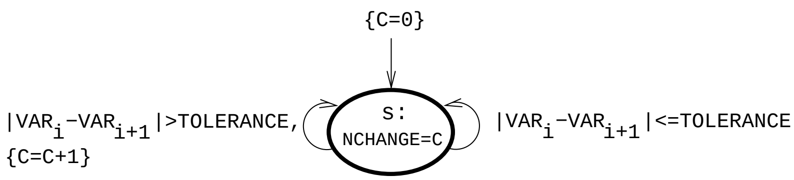

Figure 5.537: Automaton (with a counter) of the `smooth` constraint

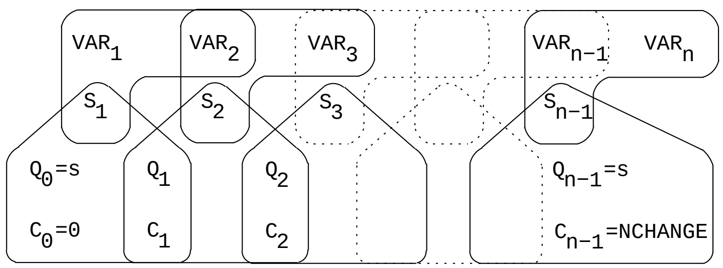

Figure 5.538: Hypergraph of the reformulation corresponding to the automaton (with a counter) of the `smooth` constraint

Since the reformulation associated with the previous automaton is not Berge-acyclic, we now describe a second counter free automaton that also only accepts all the solutions of the `smooth` constraint. Without loss of generality, assume that the collection of variables `VARIABLES` contains at least two variables (i.e., $|\texttt{VARIABLES}| \geq 2$). Let n, min, max, and $\mathcal{D}$ respectively denote the number of variables of the collection `VARIABLES`, the smallest value that can be assigned to the variables of `VARIABLES`, the largest value that can be assigned to the variables of `VARIABLES`, and the union of the domains of the variables of `VARIABLES`. Clearly, the maximum number of changes (i.e., the number of times the constraint $(|\texttt{VAR}_i - \texttt{VAR}_{i+1}|) > \texttt{TOLERANCE}$ $(1 \leq i < n)$ holds) cannot exceed the quantity $m = \min(n-1, \overline{\texttt{NCHANGE}})$. The $(m+1) \cdot |\mathcal{D}| + 2$ states of the automaton that only accepts all the solutions of the `smooth` constraint are defined in the following way:

- We have an initial state labelled by s_I.
- We have $m \cdot |\mathcal{D}|$ intermediate states labelled by s_{ij} $(i \in \mathcal{D}, j \in [0, m])$. The first subscript i of state s_{ij} corresponds to the value currently encountered. The second subscript j denotes the number of already encountered satisfied constraints of the form $(|\texttt{VAR}_k - \texttt{VAR}_{k+1}|) > \texttt{TOLERANCE}$ from the initial state s_I to the state s_{ij}.
- We have a final state labelled by s_F.

Four classes of transitions are respectively defined in the following way:

1. There is a transition, labelled by i from the initial state s_I to the state s_{i0}, $(i \in \mathcal{D})$.
2. There is a transition, labelled by j, from every state s_{ij}, $(i \in \mathcal{D}, j \in [0, m])$, to the final state s_F.
3. $\forall i \in \mathcal{D}, \ \forall j \in [0, m], \ \forall k \in \mathcal{D} \cap [\max(min, i - \texttt{TOLERANCE}), \min(max, i + \texttt{TOLERANCE})]$ there is a transition labelled by k from s_{ij} to s_{kj} (i.e., the counter j does not change for values k that are too closed from value i).
4. $\forall i \in \mathcal{D}, \ \forall j \in [0, m-1], \ \forall k \in \mathcal{D} \setminus [\max(min, i - \texttt{TOLERANCE}), \min(max, i + \texttt{TOLERANCE})]$ there is a transition labelled by k from s_{ij} to s_{kj+1} (i.e., the counter j is incremented by $+1$ for values k that are too far from i).

We have $|\mathcal{D}|$ transitions of type 1, $|\mathcal{D}| \cdot (m+1)$ transitions of type 2, and at least $|\mathcal{D}|^2 \cdot m$ transitions of types 3 and 4. Since the maximum value of m is equal to $n-1$, in the worst case we have at least $|\mathcal{D}|^2 \cdot (n-1)$ transitions. This leads to a worst case time complexity of $O(|\mathcal{D}|^2 \cdot n^2)$ if we use Pesant's algorithm for filtering the `regular` constraint [275].

Figure 5.539 depicts the corresponding counter free non deterministic automaton associated with the `smooth` constraint under the hypothesis that (1) all variables of VARIABLES are assigned a value in $\{0, 1, 2, 3\}$, (2) $|\texttt{VARIABLES}|$ is equal to 4, and (3) TOLERANCE is equal to 1.

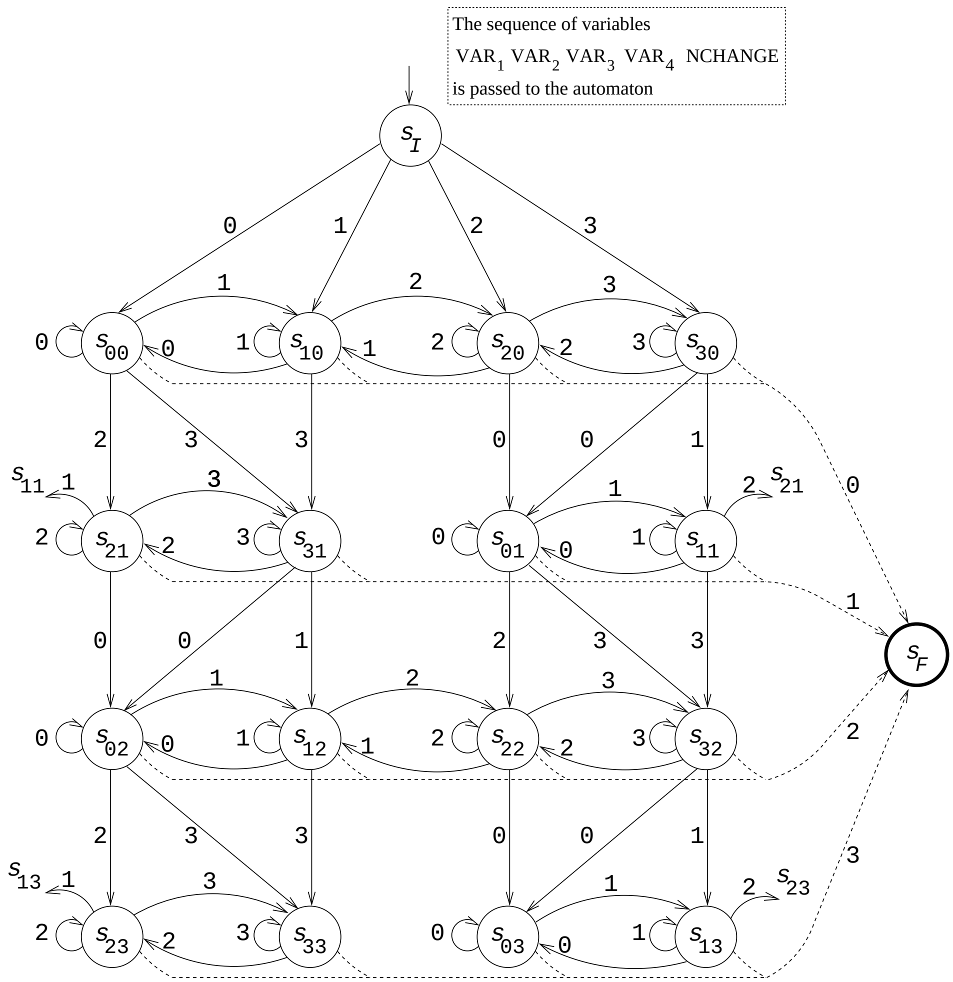

Figure 5.539: Counter free non deterministic automaton of the smooth(NCHANGE, $1, \langle$VAR$_1$, VAR$_2$, VAR$_3$, VAR$_4\rangle$) constraint assuming VAR$_i \in [0,3]$ $(1 \leq i \leq 3)$, with initial state s_I and final state s_F