5.131 distance_change

DESCRIPTION LINKS GRAPH AUTOMATON

Origin

Derived from change.

Constraint

distance_change(DIST, VARIABLES1, VARIABLES2, CTR)

Synonym

distance.

Arguments

DIST	:	dvar
VARIABLES1	:	collection(var−dvar)
VARIABLES2	:	collection(var−dvar)
CTR	:	atom

Restrictions

$\mathtt{DIST} \geq 0$
$\mathtt{DIST} < |\mathtt{VARIABLES1}|$
required(VARIABLES1, var)
required(VARIABLES2, var)
$|\mathtt{VARIABLES1}| = |\mathtt{VARIABLES2}|$
$\mathtt{CTR} \in [=, \neq, <, \geq, >, \leq]$

Purpose

DIST is equal to the number of times one of the following two conditions is true ($1 \leq i < n$):

- $\mathtt{VARIABLES1}[i].\mathtt{var}$ CTR $\mathtt{VARIABLES1}[i+1].\mathtt{var}$ holds and $\mathtt{VARIABLES2}[i].\mathtt{var}$ CTR $\mathtt{VARIABLES2}[i+1].\mathtt{var}$ does not hold,
- $\mathtt{VARIABLES2}[i].\mathtt{var}$ CTR $\mathtt{VARIABLES2}[i+1].\mathtt{var}$ holds and $\mathtt{VARIABLES1}[i].\mathtt{var}$ CTR $\mathtt{VARIABLES1}[i+1].\mathtt{var}$ does not hold.

Example

$$(1, \langle 3, 3, 1, 2, 2 \rangle, \langle 4, 4, 3, 3, 3 \rangle, \neq)$$

The distance_change constraint holds since the following condition ($\mathtt{DIST} = 1$) is verified: $\left\{ \begin{array}{l} \mathtt{VARIABLES1}[3].\mathtt{var} = 1 \neq \mathtt{VARIABLES1}[4].\mathtt{var} = 2 \wedge \\ \mathtt{VARIABLES2}[3].\mathtt{var} = 3 = \mathtt{VARIABLES1}[4].\mathtt{var} = 3 \end{array} \right.$.

Typical

$\mathtt{DIST} > 0$
$|\mathtt{VARIABLES1}| > 1$
$\mathtt{CTR} \in [=, \neq]$

Symmetries

- Arguments are permutable w.r.t. permutation (DIST) (VARIABLES1, VARIABLES2) (CTR).
- One and the same constant can be added to the var attribute of all items of VARIABLES1.
- One and the same constant can be added to the var attribute of all items of VARIABLES2.

Arg. properties

Functional dependency: DIST determined by VARIABLES1, VARIABLES2 and CTR.

Usage

Measure the distance between two sequences according to the change constraint.

Remark

We measure that distance with respect to a given constraint and not according to the fact that the variables are assigned distinct values.

See also

common keyword: distance_between *(proximity constraint)*.

root concept: change.

Keywords

characteristic of a constraint: automaton, automaton with counters.

constraint arguments: pure functional dependency.

constraint network structure: sliding cyclic(2) constraint network(2).

constraint type: proximity constraint.

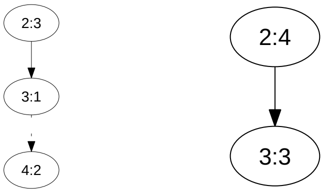
modelling: functional dependency.

Arc input(s)	VARIABLES1/ VARIABLES2
Arc generator	$PATH$ ↦ collection(variables1, variables2)
Arc arity	2
Arc constraint(s)	variables1.var CTR variables2.var
Graph property(ies)	**DISTANCE**= DIST

Graph model Within the **Arc input(s)** slot, the character / indicates that we generate two distinct graphs. The graph property DISTANCE measures the distance between two digraphs G_1 and G_2. This distance is defined as the sum of the following quantities:

- The number of arcs of G_1 that do not belong to G_2,
- The number of arcs of G_2 that do not belong to G_1.

Part (A) of Figure 5.287 gives the final graph associated with the sequence var-3,var-3,var-1,var-2,var-2 (i.e., the second argument of the constraint of the **Example** slot), while part (B) shows the final graph corresponding to var-4,var-4,var-3,var-3,var-3 (i.e., the third argument of the constraint of the **Example** slot). Since arc $3 \rightarrow 4$ belongs to the first final graph but not to the second one, the distance between the two final graphs is equal to 1.



(A) **(B)**

Figure 5.287: Final graphs of the distance_change constraint

Automaton Figure 5.288 depicts the automaton associated with the distance_change constraint. Let $(\texttt{VAR1}_i, \texttt{VAR1}_{i+1})$ and $(\texttt{VAR2}_i, \texttt{VAR2}_{i+1})$ respectively be the i^{th} pairs of consecutive variables of the collections VARIABLES1 and VARIABLES2. To each quadruple $(\texttt{VAR1}_i, \texttt{VAR1}_{i+1}, \texttt{VAR2}_i, \texttt{VAR2}_{i+1})$ corresponds a 0-1 signature variable S_i. The following signature constraint links these variables:

$((\texttt{VAR1}_i = \texttt{VAR1}_{i+1}) \wedge (\texttt{VAR2}_i \neq \texttt{VAR2}_{i+1})) \vee$

$((\texttt{VAR1}_i \neq \texttt{VAR1}_{i+1}) \wedge (\texttt{VAR2}_i = \texttt{VAR2}_{i+1})) \Leftrightarrow S_i$.

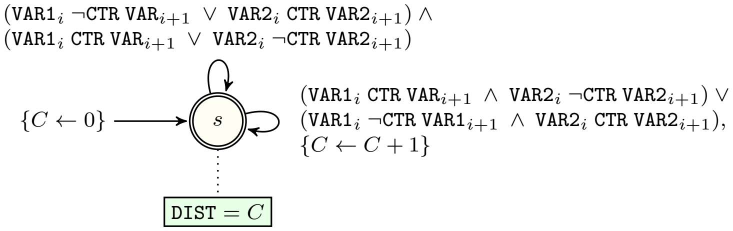

$(\texttt{VAR1}_i\ \neg\texttt{CTR}\ \texttt{VAR}_{i+1}\ \vee\ \texttt{VAR2}_i\ \texttt{CTR}\ \texttt{VAR2}_{i+1}) \wedge$
$(\texttt{VAR1}_i\ \texttt{CTR}\ \texttt{VAR}_{i+1}\ \vee\ \texttt{VAR2}_i\ \neg\texttt{CTR}\ \texttt{VAR2}_{i+1})$
$\{C \leftarrow 0\}$
s
$(\texttt{VAR1}_i\ \texttt{CTR}\ \texttt{VAR}_{i+1}\ \wedge\ \texttt{VAR2}_i\ \neg\texttt{CTR}\ \texttt{VAR2}_{i+1}) \vee$
$(\texttt{VAR1}_i\ \neg\texttt{CTR}\ \texttt{VAR1}_{i+1}\ \wedge\ \texttt{VAR2}_i\ \texttt{CTR}\ \texttt{VAR2}_{i+1})$,
$\{C \leftarrow C+1\}$
$\texttt{DIST} = C$

Figure 5.288: Automaton of the distance_change constraint

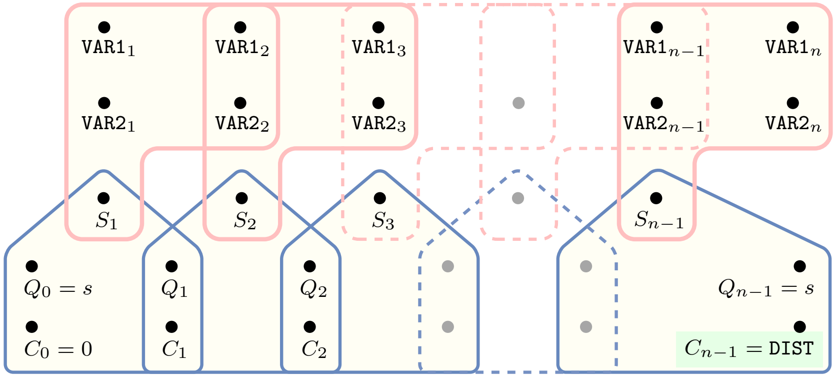


Figure 5.289: Hypergraph of the reformulation corresponding to the automaton of the distance_change constraint