

5.8 alldifferent_cst

	DESCRIPTION	LINKS	GRAPH
Origin	CHIP		
Constraint	alldifferent_cst(VARIABLES)		
Synonyms	alldiff_cst, alldistinct_cst.		
Argument	VARIABLES : <code>collection(var-dvar, cst-int)</code>		
Restriction	<code>required(VARIABLES, [var, cst])</code>		
Purpose	For all pairs of items (VARIABLES[i], VARIABLES[j]) ($i \neq j$) of the collection VARIABLES enforce $\text{VARIABLES}[i].\text{var} + \text{VARIABLES}[i].\text{cst} \neq \text{VARIABLES}[j].\text{var} + \text{VARIABLES}[j].\text{cst}$.		
Example	$\left(\left\langle \begin{array}{ll} \text{var} - 5 & \text{cst} - 0, \\ \text{var} - 1 & \text{cst} - 1, \\ \text{var} - 9 & \text{cst} - 0, \\ \text{var} - 3 & \text{cst} - 4 \end{array} \right\rangle \right)$ <p>The <code>alldifferent_cst</code> constraint holds since all the expressions $5 + 0 = 5$, $1 + 1 = 2$, $9 + 0 = 9$ and $3 + 4 = 7$ correspond to distinct values.</p>		
Typical	$ \text{VARIABLES} > 2$ <code>range(VARIABLES.cst) > 0</code>		
Symmetries	<ul style="list-style-type: none"> Items of VARIABLES are permutable. Attributes of VARIABLES are permutable w.r.t. permutation (var, cst) (<i>permutation not necessarily applied to all items</i>). One and the same constant can be added to the var attribute of all items of VARIABLES. One and the same constant can be added to the cst attribute of all items of VARIABLES. 		
Usage	The <code>alldifferent_cst</code> constraint was originally introduced in CHIP in order to express the n -queen problem with 3 global constraints (see the Usage slot of the <code>alldifferent</code> constraint).		
Algorithm	See the filtering algorithms of the <code>alldifferent</code> constraint.		
Systems	<code>distinct</code> in Gecode .		
See also	specialisation: alldifferent (variable + constant <i>replaced by</i> variable).		

Keywords

characteristic of a constraint: all different, disequality.

constraint type: value constraint.

filtering: bipartite matching, bipartite matching in convex bipartite graphs,
convex bipartite graph, arc-consistency.

final graph structure: one_succ.

modelling exercises: n-Amazon.

puzzles: n-Amazon, n-queen.

Arc input(s)	VARIABLES
Arc generator	<i>CLIQUE</i> \mapsto <code>collection(variables1, variables2)</code>
Arc arity	2
Arc constraint(s)	<code>variables1.var + variables1.cst =</code> <code>variables2.var + variables2.cst</code>
Graph property(ies)	<u>MAX_NSCC</u> \leq 1
Graph class	<u>ONE_SUCC</u>

Graph model

We generate a *clique* with an *equality* constraint between each pair of vertices (including a vertex and itself) and state that the size of the largest strongly connected component should not exceed one.

Parts (A) and (B) of Figure 5.9 respectively show the initial and final graph associated with the **Example** slot. Since we use the MAX_NSCC graph property we show one of the largest strongly connected component of the final graph. The `alldifferent_cst` holds since all the strongly connected components have at most one vertex: a value is used at most once.

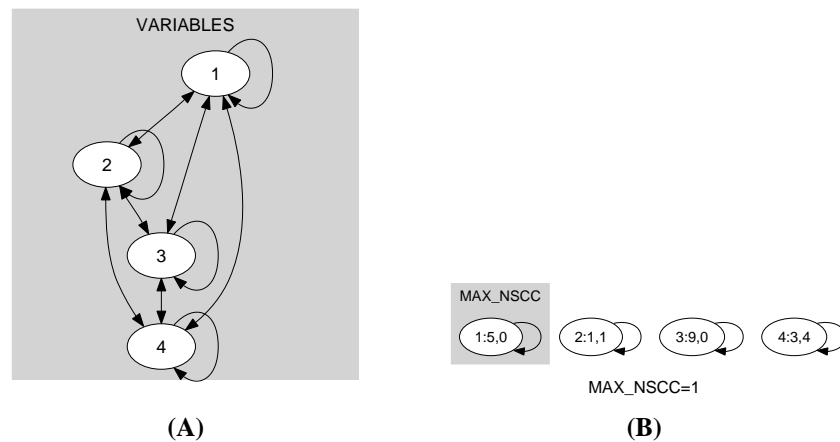


Figure 5.9: Initial and final graph of the `alldifferent_cst` constraint

20051104

413