

5.275 nequivalence

	DESCRIPTION	LINKS	GRAPH
Origin	Derived from nvalue .		
Constraint	<code>nequivalence(NEQUIV, M, VARIABLES)</code>		
Arguments	NEQUIV : dvar M : int VARIABLES : collection (<code>var—dvar</code>)		
Restrictions	required (VARIABLES, <code>var</code>) $NEQUIV \geq \min(1, VARIABLES)$ $NEQUIV \leq \min(M, VARIABLES)$ $NEQUIV \leq \text{range}(VARIABLES.\text{var})$ $M > 0$		
Purpose	NEQUIV is the number of distinct rests obtained by dividing the variables of the collection VARIABLES by M.		
Example	$(2, 3, \langle 3, 2, 5, 6, 15, 3, 3 \rangle)$ Since the expressions $3 \bmod 3 = 0$, $2 \bmod 3 = 2$, $5 \bmod 3 = 2$, $6 \bmod 3 = 0$, $15 \bmod 3 = 0$, $3 \bmod 3 = 0$, and $3 \bmod 3 = 0$ involve two distinct values (values 0 and 2), the first argument NEQUIV of the <code>nequivalence</code> constraint is set to value 2.		
Typical	$NEQUIV > 1$ $NEQUIV < VARIABLES $ $NEQUIV < \text{range}(VARIABLES.\text{var})$ $M > 1$ $M < \text{maxval}(VARIABLES.\text{var})$		
Symmetries	<ul style="list-style-type: none"> Items of VARIABLES are permutable. An occurrence of a value u of VARIABLES.<code>var</code> can be replaced by any other value v such that v is congruent to u modulo M. 		
Arg. properties	<ul style="list-style-type: none"> Functional dependency: NEQUIV determined by M and VARIABLES. Contractible wrt. VARIABLES when $NEQUIV = 1$ and $VARIABLES > 0$. Contractible wrt. VARIABLES when $NEQUIV = VARIABLES$. Extensible wrt. VARIABLES when $NEQUIV = M$. 		
Algorithm	Since constraints $X = Y$ and $X \equiv Y \pmod{M}$ are similar, one should also use a similar algorithm as the one [27, 40] provided for constraint nvalue .		

See also

related: `nclass`(variable mod constant *replaced by* variable \in partition),
`ninterval`(variable mod constant *replaced by* variable/constant),
`npair`(variable mod constant *replaced by* pair of variables).
specialisation: `nvalue`(variable mod constant *replaced by* variable).

Keywords

constraint arguments: pure functional dependency.
constraint type: counting constraint, value partitioning constraint.
final graph structure: strongly connected component, equivalence.
modelling: number of distinct equivalence classes, functional dependency.

Arc input(s)	VARIABLES
Arc generator	<i>CLIQUE</i> \mapsto collection(variables1, variables2)
Arc arity	2
Arc constraint(s)	variables1.var mod M = variables2.var mod M
Graph property(ies)	<i>NSCC</i> = NEQUIV

Graph model

Parts (A) and (B) of Figure 5.593 respectively show the initial and final graph associated with the **Example** slot. Since we use the *NSCC* graph property we show the different strongly connected components of the final graph. Each strongly connected component corresponds to one equivalence class: We have two equivalence classes that respectively correspond to values {3, 6, 15} and {2, 5}.

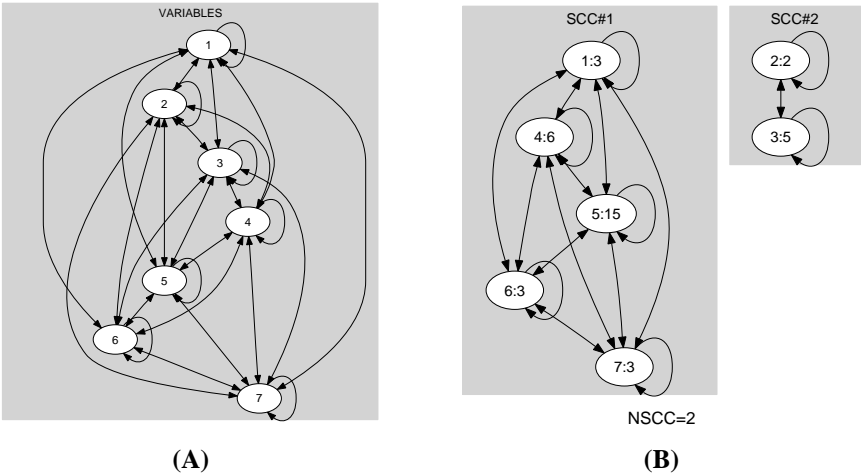


Figure 5.593: Initial and final graph of the nequivalence constraint

