## 5.334 roots

DESCRIPTION LINKS GRAPH

**Origin** [63]

**Constraint** roots(S, T, VARIABLES)

**Arguments**

| | | |
|---|---|---|
| S | : | svar |
| T | : | svar |
| VARIABLES | : | collection(var−dvar) |

**Restrictions**

S $\leq$ |VARIABLES|
required(VARIABLES, var)

**Purpose** S is the set of indices of the variables in the collection VARIABLES taking their values in T; S $= \{i \mid$ VARIABLES$[i]$.var $\in$ T$\}$. Positions are numbered from 1.

**Example** $(\{2,4,5\},\{2,3,8\},\langle 1,3,1,2,3\rangle)$

The roots constraint holds since values 2 and 3 in T occur in the collection $\langle 1,3,1,2,3\rangle$ only at positions S $= \{2,4,5\}$. The value $8 \in$ T does not occur within the collection $\langle 1,3,1,2,3\rangle$.

**Typical**

|VARIABLES| $> 1$
range(VARIABLES.var) $> 1$

**Usage** Bessière et al. showed [63] that many counting and occurence constraints can be specified with two *global primitives*: roots and range. For instance, the count constraint can be decomposed into one roots constraint: count(VAL, VARS, OP, NVAR) iff roots(S, {VAL}, VARS) $\wedge$ |S| OP NVAR.

roots does not count but collects the set of variables using particular values. It provides then a way of channeling. roots generalises, for instance, the link_set_to_booleans constraint, link_set_to_booleans(S, BOOLEANS) iff roots(S, {1}, BOOLEANS.bool), or may be used instead of the domain_constraint.

Other examples of reformulations are given in [67].

**Algorithm** In [66], Bessière *et al.* shows that enforcing hybrid-consistency on roots is NP-hard. They consider the decomposition of roots into a network of ternary constraints: $\forall i$, $i \in$ S $\Rightarrow$ VARIABLES$[i]$.var $\in$ T and VARIABLES$[i]$.var $\Rightarrow$ T $\wedge\ i \in$ S. Enforcing bound consistency on the decomposition achieves bound consistency on roots. Enforcing hybrid consistency on the decomposition achieves at least bound consistency on roots, until hybrid consistency in some special cases:

- $dom$(VARIABLES$[i]$.var) $\subset \underline{\texttt{T}}$, $\forall i \in \underline{\texttt{S}}$,
- $dom$(VARIABLES$[i]$.var) $\cap\ \overline{\texttt{T}} = \emptyset$, $\forall i \notin \overline{\texttt{S}}$,

- VARIABLES are ground,
- T is ground.

Enforcing hybrid consistency on the decomposition can be done in $O(nd)$ with $n = |\texttt{VARIABLES}|$ and $d$ the maximum domain size of $\texttt{VARIABLES}[i].\texttt{var}$ and T.

**Systems** roots in **Gecode**, roots in **MiniZinc**.

**See also** **common keyword:** link_set_to_booleans *(constraint involving set variables)*.

**related:** among *(can be expressed with* roots*)*, assign_and_nvalues *(can be expressed with* roots *and* range*)*, atleast, atmost *(can be expressed with* roots*)*, common *(can be expressed with* roots *and* range*)*, count *(can be expressed with* roots*)*, domain_constraint, global_cardinality, global_contiguity *(can be expressed with* roots*)*, symmetric_alldifferent, uses *(can be expressed with* roots *and* range*)*.

**Keywords** **characteristic of a constraint:** disequality.

**constraint arguments:** constraint involving set variables.

**constraint type:** counting constraint, value constraint, decomposition.

**filtering:** hybrid-consistency.

**Derived Collection** col(SETS−collection(s−svar, t−svar), [item(s − S, t − T)])

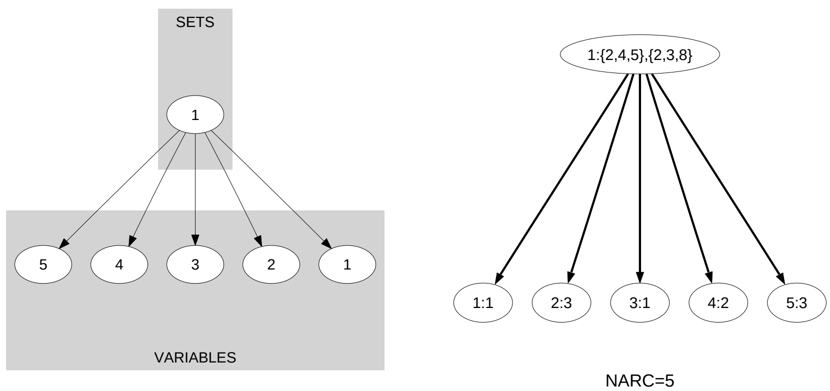
**Arc input(s)** SETS VARIABLES

**Arc generator** $PRODUCT$ ↦ collection(sets, variables)

**Arc arity** 2

**Arc constraint(s)** in_set(variables.key, sets.s) ⇔ in_set(variables.var, sets.t)

**Graph property(ies)** **NARC** = |VARIABLES|




**(A)** **(B)**

Figure 5.681: Initial and final graph of the `roots` constraint

**Graph model**