5.121 discrepancy

DESCRIPTION LINKS GRAPH

Origin

[170] and [423]

Constraint

discrepancy(VARIABLES, K)

Arguments

VARIABLES : collection(var−dvar, bad−sint)
K : int

Restrictions

required(VARIABLES, var)
required(VARIABLES, bad)
K ≥ 0
K ≤ |VARIABLES|

Purpose

K is the number of variables of the collection VARIABLES that take their value in their respective sets of bad values.

Example

$$\left(\left\langle\begin{array}{ll}\texttt{var}-4 & \texttt{bad}-\{1,4,6\},\\ \texttt{var}-5 & \texttt{bad}-\{0,1\},\\ \texttt{var}-5 & \texttt{bad}-\{1,6,9\},\\ \texttt{var}-4 & \texttt{bad}-\{1,4\},\\ \texttt{var}-1 & \texttt{bad}-\emptyset\end{array}\right\rangle, 2\right)$$

The discrepancy constraint holds since exactly K = 2 variables (i.e., the first and fourth variables) of the VARIABLES collection take their value within their respective sets of bad values.

Typical

|VARIABLES| > 1
K < |VARIABLES|

Symmetries

- Items of VARIABLES are permutable.
- All occurrences of two distinct values in VARIABLES.var or VARIABLES.bad can be swapped; all occurrences of a value in VARIABLES.var or VARIABLES.bad can be renamed to any unused value.

Arg. properties

- Functional dependency: K determined by VARIABLES.
- Aggregate: VARIABLES(union), K(+).

Remark

Limited discrepancy search was first introduced by M. L. Ginsberg and W. D. Harvey as a search technique in [193]. Later on, discrepancy based filtering was presented in the PhD thesis of F. Focacci [170, pages 171–172]. Finally the discrepancy constraint was explicitly defined in the PhD thesis of W.-J. van Hoeve [423, page 104].

See also

common keyword: among *(counting constraint)*.

used in graph description: in_set.

Keywords

constraint arguments: pure functional dependency.

constraint type: value constraint, counting constraint.

filtering: arc-consistency.

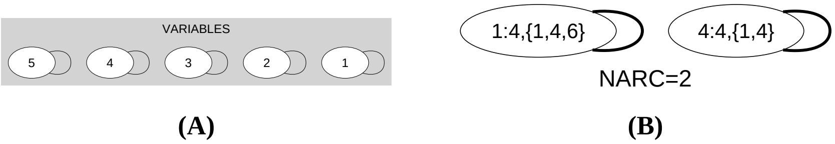
heuristics: heuristics, limited discrepancy search.

modelling: functional dependency.

Arc input(s)	`VARIABLES`
Arc generator	*SELF*↦`collection`(`variables`)
Arc arity	1
Arc constraint(s)	`in_set`(`variables.var`, `variables.bad`)
Graph property(ies)	**NARC**= `K`

Graph model The arc constraint corresponds to the constraint `in_set`(`variables.var`, `variables.bad`) defined in this catalogue. We employ the *SELF* arc generator in order to produce an initial graph with a single loop on each vertex.

Parts (A) and (B) of Figure 5.289 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NARC** graph property, the loops of the final graph are stressed in bold.



(A) **(B)**

Figure 5.289: Initial and final graph of the `discrepancy` constraint