

5.28 assign_and_nvalues

	DESCRIPTION	LINKS	GRAPH
Origin	Derived from <code>assign_and_counts</code> and <code>nvalues</code> .		
Constraint	<code>assign_and_nvalues</code> (ITEMS, RELOP, LIMIT)		
Arguments	ITEMS : <code>collection</code> (bin-dvar, value-dvar) RELOP : <code>atom</code> LIMIT : <code>dvar</code>		
Restrictions	<code>required</code> (ITEMS, [bin, value]) RELOP ∈ [=, ≠, <, ≥, >, ≤]		
Purpose	<div style="border: 1px solid pink; padding: 5px;"> Given several items (each of them having a specific value that may not be initially fixed), and different bins, assign each item to a bin, so that the number n of distinct values in each bin satisfies the condition n RELOP LIMIT. </div>		
Example	<div style="border: 1px solid blue; padding: 10px; display: inline-block;"> $\left(\left\langle \begin{array}{l} \text{bin} - 2 \quad \text{value} - 3, \\ \text{bin} - 1 \quad \text{value} - 5, \\ \text{bin} - 2 \quad \text{value} - 3, \\ \text{bin} - 2 \quad \text{value} - 3, \\ \text{bin} - 2 \quad \text{value} - 4 \end{array} \right\rangle, \leq, 2 \right)$ </div>		

Figure 5.53 depicts the solution corresponding to the example. The `assign_and_nvalues` constraint holds since for each used bin (i.e., namely bins 1 and 2) the number of distinct colours of the corresponding assigned items is less than or equal to the limit 2.

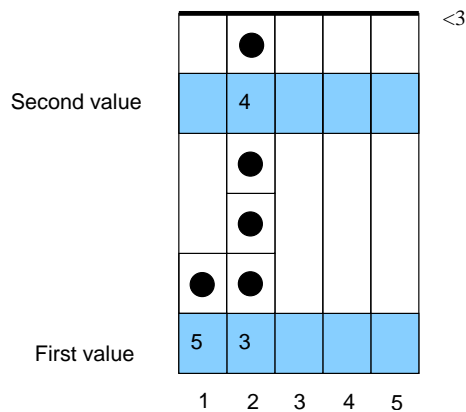


Figure 5.53: An assignment with at most two distinct values in parallel

Typical

```
|ITEMS| > 1  
range(ITEMS.bin) > 1  
range(ITEMS.value) > 1  
LIMIT > 1  
LIMIT < |ITEMS|
```

Symmetries

- Items of ITEMS are [permutable](#).
- All occurrences of two distinct values of ITEMS.bin can be [swapped](#); all occurrences of a value of ITEMS.bin can be [renamed](#) to any unused value.

Usage

Let us give two examples where the `assign_and_nvalues` constraint is useful:

- Quite often, in bin-packing problems, each item has a specific type, and one wants to assign items of similar type to each bin.
- In a vehicle routing problem, one wants to restrict the number of towns visited by each vehicle. Note that several customers may be located at the same town. In this example, each bin would correspond to a vehicle, each item would correspond to a visit to a customer, and the colour of an item would be the location of the corresponding customer.

See also

[assignment dimension removed: nvalue, nvalues](#).
[common keyword: nvalues_except_0 \(number of distinct values\)](#).
[related: roots](#).
[used in graph description: nvalues](#).

Keywords

[application area: assignment](#).
[final graph structure: acyclic, bipartite, no loop](#).
[modelling: assignment dimension, number of distinct values](#).

Arc input(s)	ITEMS ITEMS
Arc generator	<code>PRODUCT</code> \mapsto <code>collection(items1, items2)</code>
Arc arity	2
Arc constraint(s)	<code>items1.bin = items2.bin</code>
Graph class	<ul style="list-style-type: none"> • <code>ACYCLIC</code> • <code>BIPARTITE</code> • <code>NO_LOOP</code>
Sets	<code>SUCC</code> \mapsto $\left[\begin{array}{l} \text{source,} \\ \text{variables} - \text{col} \left(\begin{array}{l} \text{VARIABLES} - \text{collection}(\text{var} - \text{dvar}), \\ [\text{item}(\text{var} - \text{ITEMS.value})] \end{array} \right) \end{array} \right]$
Constraint(s) on sets	<code>nvalues(variables, RELOP, LIMIT)</code>

Graph model

We enforce the `nvalues` constraint on the items that are assigned to the same bin.

Parts (A) and (B) of Figure 5.54 respectively show the initial and final graph associated with the **Example** slot. The final graph consists of the following two connected components:

- The connected component containing 8 vertices corresponds to the items that are assigned to bin 2.
- The connected component containing 2 vertices corresponds to the items that are assigned to bin 1.

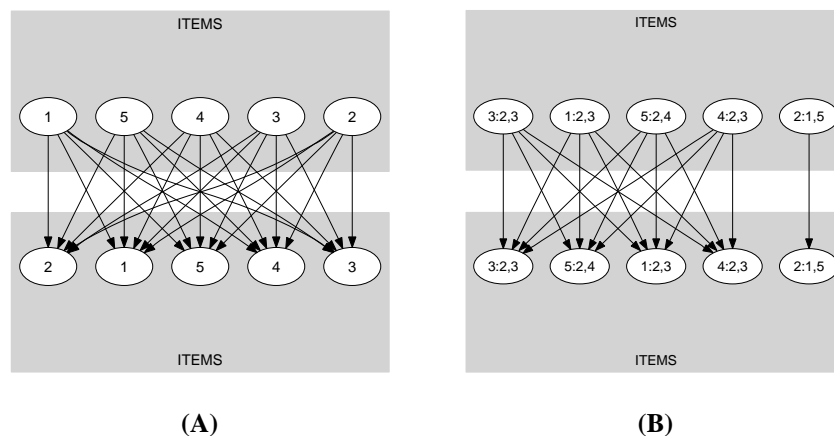


Figure 5.54: Initial and final graph of the `assign_and_nvalues` constraint

The `assign_and_nvalues` constraint holds since for each set of successors of the vertices of the final graph no more than two distinct values are used:

- The unique item assigned to bin 1 uses value 5.
- Items assigned to bin 2 use values 3 and 4.

20000128

491