5.163 global_cardinality

DESCRIPTION LINKS GRAPH AUTOMATON

Origin CHARME [298]

Constraint `global_cardinality(VARIABLES, VALUES)`

Synonyms `count`, `distribute`, `distribution`, `gcc`, `card_var_gcc`, `egcc`, `extended_global_cardinality`.

Arguments

$$\begin{array}{lll} \texttt{VARIABLES} & : & \texttt{collection}(\texttt{var}-\texttt{dvar}) \\ \texttt{VALUES} & : & \texttt{collection}(\texttt{val}-\texttt{int}, \texttt{noccurrence}-\texttt{dvar}) \end{array}$$

Restrictions

$$\begin{array}{l} \texttt{required}(\texttt{VARIABLES}, \texttt{var}) \\ \texttt{required}(\texttt{VALUES}, [\texttt{val}, \texttt{noccurrence}]) \\ \texttt{distinct}(\texttt{VALUES}, \texttt{val}) \\ \texttt{VALUES.noccurrence} \geq 0 \\ \texttt{VALUES.noccurrence} \leq |\texttt{VARIABLES}| \end{array}$$

Purpose Each value $\texttt{VALUES}[i].\texttt{val}$ (with $i \in [1, |\texttt{VALUES}|]$) should be taken by exactly $\texttt{VALUES}[i].\texttt{noccurrence}$ variables of the `VARIABLES` collection.

Example

$$\left(\begin{array}{l} \langle 3, 3, 8, 6 \rangle, \\ \left\langle \begin{array}{ll} \texttt{val} - 3 & \texttt{noccurrence} - 2, \\ \texttt{val} - 5 & \texttt{noccurrence} - 0, \\ \texttt{val} - 6 & \texttt{noccurrence} - 1 \end{array} \right\rangle \end{array} \right)$$

The `global_cardinality` constraint holds since values 3, 5 and 6 respectively occur 2, 0 and 1 times within the collection $\langle 3, 3, 8, 6\rangle$ and since no restriction was specified for value 8.

All solutions Figure 5.369 gives all solutions to the following non ground instance of the `global_cardinality` constraint: $\texttt{V}_1 \in [3,4]$, $\texttt{V}_2 \in [2,3]$, $\texttt{V}_3 \in [1,2]$, $\texttt{V}_4 \in [2,4]$, $\texttt{V}_5 \in [2,3]$, $\texttt{V}_6 \in [1,2]$, $\texttt{O}_1 \in [1,1]$, $\texttt{O}_2 \in [2,3]$, $\texttt{O}_3 \in [0,1]$, $\texttt{O}_4 \in [2,3]$, $\texttt{global\_cardinality}(\langle \texttt{V}_1, \texttt{V}_2, \texttt{V}_3, \texttt{V}_4, \texttt{V}_5, \texttt{V}_6\rangle, \langle 1\ \texttt{O}_1,\ 2\ \texttt{O}_2,\ 3\ \texttt{O}_3,\ 4\ \texttt{O}_4\rangle)$.

Typical

$$\begin{array}{l} |\texttt{VARIABLES}| > 1 \\ \texttt{range}(\texttt{VARIABLES.var}) > 1 \\ |\texttt{VALUES}| > 1 \\ |\texttt{VARIABLES}| \geq |\texttt{VALUES}| \\ \texttt{minval}(\texttt{VARIABLES.var}) = 0 \vee \texttt{in\_attr}(\texttt{VARIABLES}, \texttt{var}, \texttt{VALUES}, \texttt{val}) \end{array}$$

① $(\langle 4,2,1,4,2,2\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{3},\ 3\ \mathbf{0},\ 4\ \mathbf{2}\rangle)$
② $(\langle 4,2,1,4,3,2\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{2},\ 3\ \mathbf{1},\ 4\ \mathbf{2}\rangle)$
③ $(\langle 4,2,2,4,2,1\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{3},\ 3\ \mathbf{0},\ 4\ \mathbf{2}\rangle)$
④ $(\langle 4,2,2,4,3,1\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{2},\ 3\ \mathbf{1},\ 4\ \mathbf{2}\rangle)$
⑤ $(\langle 4,3,1,4,2,2\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{2},\ 3\ \mathbf{1},\ 4\ \mathbf{2}\rangle)$
⑥ $(\langle 4,3,2,4,2,1\rangle, \langle 1\ \mathbf{1},\ 2\ \mathbf{2},\ 3\ \mathbf{1},\ 4\ \mathbf{2}\rangle)$

Figure 5.369: All solutions corresponding to the non ground example of the `global_cardinality` constraint of the **All solutions** slot

Symmetries

- Items of `VARIABLES` are permutable.
- Items of `VALUES` are permutable.
- An occurrence of a value of `VARIABLES.var` that does not belong to `VALUES.val` can be replaced by any other value that also does not belong to `VALUES.val`.
- All occurrences of two distinct values in `VARIABLES.var` or `VALUES.val` can be swapped; all occurrences of a value in `VARIABLES.var` or `VALUES.val` can be renamed to any unused value.

Arg. properties

- Functional dependency: `VALUES.noccurrence` determined by `VARIABLES` and `VALUES.val`.
- Contractible wrt. `VALUES`.

Usage

We show how to use the `global_cardinality` constraint in order to model the magic series problem [415, page 155] with a single `global_cardinality` constraint. A non-empty finite series $S = (s_0, s_1, \ldots, s_n)$ is *magic* if and only if there are s_i occurrences of i in S for each integer i ranging from 0 to n. This leads to the following model:

$$\texttt{global\_cardinality}\left(\begin{array}{l}\langle\ \texttt{var}-s_0, \texttt{var}-s_1, \ldots, \texttt{var}-s_n\ \rangle,\\ \left\langle\begin{array}{ll}\texttt{val}-0 & \texttt{noccurrence}-s_0,\\ \texttt{val}-1 & \texttt{noccurrence}-s_1,\\ & \vdots\\ \texttt{val}-n & \texttt{noccurrence}-s_n\end{array}\right\rangle\end{array}\right)$$

Remark

This is a generalised form of the original `global_cardinality` constraint: in the original `global_cardinality` constraint [342], one specifies for each value its minimum and maximum number of occurrences (i.e., see `global_cardinality_low_up`). Here we give for each value v a domain variable that indicates how many time value v is actually used. By setting the minimum and maximum values of this variable to the appropriate constants we can express the same thing as in the original `global_cardinality` constraint. However, as shown in the *magic series* problem, we can also use this variable in other constraints. By reduction from 3-SAT, Claude-Guy Quimper shows in [331] that it is NP-hard to achieve arc-consistency for the count variables.

A last difference with the original `global_cardinality` constraint comes from the fact that there is no constraint on the values that are not explicitly mentioned in the `VALUES` collection. In the original `global_cardinality` these values could not be assigned to the variables of the `VARIABLES` collection. However allowing values that are not mentioned in `VALUES` to be assigned to variables of `VARIABLES` can potentially avoid mentioning a huge number of unconstrained values in the `VALUES` collection, and as a side effect, prevent possibly[6] generating a dense graph (i.e., see DFS-bottleneck) for the corresponding underlying flow model).

Within [83] the `global_cardinality` constraint is called `distribution`. Within [350] the `global_cardinality` constraint is called `card_var_gcc`. Within [70] the `global_cardinality` constraint is called `egcc` or `rgcc`. This later case corresponds to the fact that some variables are duplicated within the `VARIABLES` collection.

The `global_cardinality` constraint can be seen as a system (i.e., a conjunction) of `among` constraints.

When all count variables (i.e., the variables $\texttt{VALUES}[i].\texttt{noccurrence}$ with $i \in [1, |\texttt{VALUES}|]$) *do not occur* in any other constraints of the problem, it may be operationally more efficient to replace the `global_cardinality` constraint by a `global_cardinality_low_up` constraint where each count variable $\texttt{VALUES}[i].\texttt{noccurrence}$ is replaced by the corresponding interval $[\underline{\texttt{VALUES}[i].\texttt{noccurrence}}, \overline{\texttt{VALUES}[i].\texttt{noccurrence}}]$. This stands for two reasons:

- First, by using a `global_cardinality_low_up` constraint rather than a `global_cardinality` constraint, we avoid the filtering algorithm related to the count variables.
- Second, unlike the `global_cardinality` constraint where we need to fix all its variables to get entailment, the `global_cardinality_low_up` constraint can be entailed before all its variables get fixed. As a result, this potentially avoid unnecessary calls to its filtering algorithm.

When all values that can be assigned to the variables of the `VARIABLES` collection occur in the `val` attribute of the `VALUES` collection, two implicit *necessary* conditions[7] inferred by double counting with the `global_cardinality` constraint are depicted by the following expressions:

$$|\texttt{VARIABLES}| = \sum_{i=1}^{|\texttt{VALUES}|} \texttt{VALUES}[i].\texttt{noccurrence}$$

$$\sum_{i=1}^{|\texttt{VARIABLES}|} \texttt{VARIABLES}[i].\texttt{var} = \sum_{i=1}^{|\texttt{VALUES}|} \texttt{VALUES}[i].\texttt{val} \cdot \texttt{VALUES}[i].\texttt{noccurrence}$$

Within [317, pages 50–51] the previous condition where terms involving identical variables are grouped together (i.e., rule 5 of MALICE [316]) is mentioned as a crucial deduction rule for the autoref problem.

[6] Of course one could also, while generating a flow model, detect all unconstrained values in order to generate a single vertex in the flow model for the set of unconstrained values.

[7] Note that such necessary conditions can be derived by assigning an integer weight to each value [385], e.g. 1 for the first condition, the value itself for the second condition.

W.-J. van Hoeve *et al.* present two soft versions of the `global_cardinality` constraint in [424].

In **MiniZinc** (`http://www.minizinc.org/`) there is also a `distribute` constraint where the `val` attribute is not necessarily initially fixed and where a same value may occur more than once. Their is also a `global_cardinality_closed` constraint where all variables must be assigned a value from the `val` attribute.

Algorithm A flow algorithm that handles the original `global_cardinality` constraint is described in [342]. The two approaches that were used to design bound-consistency algorithms for `alldifferent` were generalised for the `global_cardinality` constraint. The algorithm in [334] identifies Hall intervals and the one in [233] exploits convexity to achieve a fast implementation of the flow-based arc-consistency algorithm. The later algorithm can also compute bound-consistency for the count variables [234, 231]. An improved algorithm for achieving arc-consistency is described in [333].

Systems `globalCardinality` in **Choco**, `count` in **Gecode**, `gcc` in **JaCoP**, `global_cardinality` in **MiniZinc**, `global_cardinality` in **SICStus**.

See also

common keyword: `count`, `max_nvalue`, `min_nvalue` *(value constraint,counting constraint)*, `nvalue` *(counting constraint)*, `open_global_cardinality_low_up` *(assignment,counting constraint)*.

cost variant: `global_cardinality_with_costs` *(*`cost` *associated with each* `variable`*,*`value` *pair)*.

implied by: `global_cardinality_with_costs` *(forget about cost)*, `same_and_global_cardinality` *(conjoin* `same` *and* `global_cardinality`*)*.

part of system of constraints: `among`.

related: `roots`, `sliding_card_skip0` *(counting constraint of a set of values on maximal sequences)*.

shift of concept: `global_cardinality_no_loop` *(assignment of a* `variable` *to its position is ignored)*, `ordered_global_cardinality` *(restrictions are done on nested sets of values, all starting from first value)*, `symmetric_cardinality`, `symmetric_gcc`.

soft variant: `open_global_cardinality` *(a* `set variable` *defines the set of variables that are actually considered)*.

specialisation: `alldifferent` *(each value should occur at most once)*, `cardinality_atleast`, `cardinality_atmost` *(individual* `count variable` *for each value replaced by single* `count variable`*)*, `cardinality_atmost_partition` *(individual* `count variable` *for each value replaced by single* `count variable` *and* `variable` $\in$ `partition` *replaced by* `variable`*)*, `global_cardinality_low_up` *(*`variable` *replaced by* `fixed interval`*)*.

system of constraints: `colored_matrix` *(one* `global_cardinality` *constraint for each* `row` *and each* `column` *of a* `matrix` *of* `variables`*)*.

uses in its reformulation: `tree_range`, `tree_resource`.

Keywords

application area: assignment.

characteristic of a constraint: core, automaton, automaton with array of counters.

complexity: 3-SAT.

constraint arguments: pure functional dependency.

constraint type: value constraint, counting constraint, system of constraints.

filtering: Hall interval, bound-consistency, flow, duplicated variables, DFS-bottleneck.

modelling: functional dependency.

modelling exercises: magic series.

puzzles: magic series, autoref.

Cond. implications

- `global_cardinality`(VARIABLES, VALUES)
  with `minval`(VARIABLES.var) $= 0$
  implies `and`(VAR, VARIABLES)
  when VAR $= 0$.

- `global_cardinality`(VARIABLES, VALUES)
  with `maxval`(VARIABLES.var) $= 1$
  implies `or`(VAR, VARIABLES)
  when VAR $= 1$.

- `global_cardinality`(VARIABLES, VALUES)
  with `minval`(VARIABLES.var) > 0
  implies `min_size_full_zero_stretch`(MINSIZE, VARIABLES)
  when MINSIZE $=$ |VARIABLES|.

- `global_cardinality`(VARIABLES, VALUES)
  with `maxval`(VARIABLES.var) < 0
  implies `min_size_full_zero_stretch`(MINSIZE, VARIABLES)
  when MINSIZE $=$ |VARIABLES|.

- `global_cardinality`(VARIABLES, VALUES)
  with `range`(VALUES.val) $=$`nval`(VALUES.val)
  and `minval`(VALUES.val) $\leq$`minval`(VARIABLES.var)
  and `maxval`(VALUES.val) $\geq$`maxval`(VARIABLES.var)
  implies `among_diff_0`(NVAR, VARIABLES).

- `global_cardinality`(VARIABLES, VALUES)
  with `range`(VALUES.val) $=$`nval`(VALUES.val)
  and `minval`(VALUES.val) $\leq$`minval`(VARIABLES.var)
  and `maxval`(VALUES.val) $\geq$`maxval`(VARIABLES.var)
  implies `atmost_nvalue`(NVAL, VARIABLES).

- `global_cardinality`(VARIABLES, VALUES)
  with `range`(VALUES.noccurrence) $= 1$
  and `range`(VALUES.val) $=$`nval`(VALUES.val)
  and `minval`(VALUES.val) $=$`minval`(VARIABLES.var)
  and `maxval`(VALUES.val) $=$`maxval`(VARIABLES.var)
  implies `balance`(BALANCE, VARIABLES)
  when BALANCE $= 0$.

- `global_cardinality`(VARIABLES, VALUES)
  with `range`(VALUES.val) $=$`nval`(VALUES.val)
  and `minval`(VALUES.val) $\leq$`minval`(VARIABLES.var)
  and `maxval`(VALUES.val) $\geq$`maxval`(VARIABLES.var)
  implies `max_n`(MAX, RANK, VARIABLES).

- global_cardinality(VARIABLES, VALUES)
  with range(VALUES.val) = nval(VALUES.val)
  and minval(VALUES.val) ≤ minval(VARIABLES.var)
  and maxval(VALUES.val) ≥ maxval(VARIABLES.var)
  implies max_nvalue(MAX, VARIABLES).

- global_cardinality(VARIABLES, VALUES)
  with range(VALUES.val) = nval(VALUES.val)
  and minval(VALUES.val) ≤ minval(VARIABLES.var)
  and maxval(VALUES.val) ≥ maxval(VARIABLES.var)
  implies min_n(MIN, RANK, VARIABLES).

- global_cardinality(VARIABLES, VALUES)
  with range(VALUES.val) = nval(VALUES.val)
  and minval(VALUES.val) ≤ minval(VARIABLES.var)
  and maxval(VALUES.val) ≥ maxval(VARIABLES.var)
  implies min_nvalue(MIN, VARIABLES).

- global_cardinality(VARIABLES, VALUES)
  with range(VALUES.val) = nval(VALUES.val)
  and minval(VALUES.val) ≤ minval(VARIABLES.var)
  and maxval(VALUES.val) ≥ maxval(VARIABLES.var)
  implies range_ctr(VARIABLES, CTR, R).

For all items of `VALUES`:

Arc input(s)	`VARIABLES`
Arc generator	$SELF \mapsto$ `collection`(`variables`)
Arc arity	1
Arc constraint(s)	`variables.var` $=$ `VALUES.val`
Graph property(ies)	**NVERTEX** $=$ `VALUES.noccurrence`

Graph model Since we want to express one unary constraint for each value we use the "For all items of `VALUES`" iterator. Part (A) of Figure 5.370 shows the initial graphs associated with each value 3, 5 and 6 of the `VALUES` collection of the **Example** slot. Part (B) of Figure 5.370 shows the two corresponding final graphs respectively associated with values 3 and 6 that are both assigned to the variables of the `VARIABLES` collection (since value 5 is not assigned to any variable of the `VARIABLES` collection the final graph associated with value 5 is empty). Since we use the **NVERTEX** graph property, the vertices of the final graphs are stressed in bold.

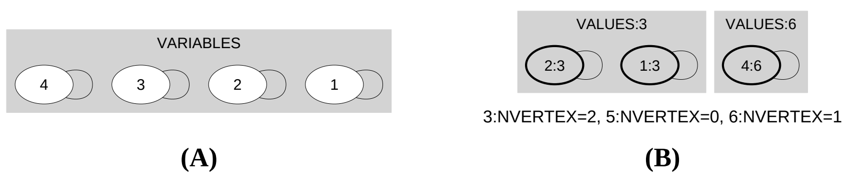


Figure 5.370: Initial and final graph of the `global_cardinality` constraint

Automaton Figure 5.371 depicts the automaton associated with the `global_cardinality` constraint. To each item of the collection `VARIABLES` corresponds a signature variable S_i that is equal to 0. To each item of the collection `VALUES` corresponds a signature variable $S_{i+|\mathtt{VARIABLES}|}$ that is equal to 1.

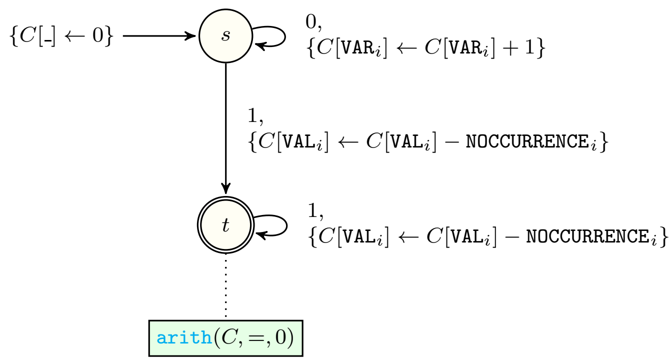


Figure 5.371: Automaton of the `global_cardinality` constraint

Quiz

EXERCISE 1 (checking whether a ground instance holds or not)[a]

A. Does the constraint
`global_cardinality`$(\langle 2, 4, 2, 2, 1\rangle, \langle 0\ 0, 1\ 1, 2\ 3, 3\ 0, 4\ 1\rangle)$ hold?

B. Does the constraint
`global_cardinality`$(\langle 0, 0, 1, 1\rangle, \langle 0\ 2, 1\ 2, 2\ 1, 3\ 0, 4\ 0\rangle)$ hold?

C. Does the constraint
`global_cardinality`$(\langle 2, 3, 4, 5\rangle, \langle 0\ 0, 1\ 0, 2\ 1, 3\ 1, 4\ 1\rangle)$ hold?

[a]Hint: go back to the definition of `global_cardinality`.

EXERCISE 2 (finding all solutions)[a]

Give all the solutions to the constraint:

$$\left\{\begin{array}{lll} V_1 \in [1,2], & V_2 \in [1,2], & V_3 \in [1,2], \\ V_4 \in [2,3], & V_5 \in [3,3], & \\ O_1 \in [1,2], & O_2 \in [2,3], & O_3 \in [0,1], \\ \multicolumn{3}{l}{\mathtt{global\_cardinality}\left(\begin{array}{l}\langle V_1, V_2, V_3, V_4, V_5\rangle, \\ \left\langle\begin{array}{ll}\mathtt{val}-1 & \mathtt{occurrence}-O_1, \\ \mathtt{val}-2 & \mathtt{occurrence}-O_2, \\ \mathtt{val}-3 & \mathtt{occurrence}-O_3\end{array}\right\rangle\end{array}\right).} \end{array}\right.$$

[a]Hint: focus on the variables of the first argument (since the counting variables of the second argument are functionally determined by the first argument), and enumerate solutions in lexicographic order.

EXERCISE 3 (identifying infeasible values)[a]

Identify all variable-value pairs (V_i, val) (respectively (O_i, val)) (with $i \in [1, 5]$), such that the following constraint has no solution when variable V_i (respectively O_i) is assigned value val:

$$\left\{\begin{array}{l} V_1 \in [2,3], \quad V_2 \in [1,5], \quad V_3 \in [3,4], \\ V_4 \in [1,3], \quad V_5 \in [1,4], \\ O_1 \in [1,4], \quad O_2 \in [0,1], \quad O_3 \in [0,1], \\ O_4 \in [1,5], \quad O_5 \in [1,4], \\ \texttt{global\_cardinality}\left(\begin{array}{l} \langle V_1, V_2, V_3, V_4, V_5 \rangle, \\ \left\langle \begin{array}{ll} \texttt{val}-1 & \texttt{occurrence}-O_1, \\ \texttt{val}-2 & \texttt{occurrence}-O_2, \\ \texttt{val}-3 & \texttt{occurrence}-O_3, \\ \texttt{val}-4 & \texttt{occurrence}-O_4, \\ \texttt{val}-5 & \texttt{occurrence}-O_5 \end{array} \right\rangle \end{array}\right). \end{array}\right.$$

[a]Hint: first restrict the occurrence variables $O_1, O_2, \ldots, O_5$, second restrict the decision variables $V_1, V_2, \ldots, V_5$, third check that all remaining values occur in at least one solution.

EXERCISE 4 (modelling a nurse assignment problem)[a]

Given a 24 hour period, you must schedule a pool of six nurses `Bea`, `Lea`, `Leo`, `Lio`, `Lili` and `Tom` to *at least two* and *at most three morning shifts*, to *at least two* and *at most three afternoon shifts*, to *at least one night shift*, while the other nurses are off-duty. In addition, due to past work, we have the following extra requirements:

- Since on the previous 24 hour period `Bea`, `Lea` and `Leo` were working in the afternoon shift they cannot be assigned to the night shift.
- `Leo`, `Lio` and `Lili` have to work since they already took all their days off.
- `Bea` and `Tom` have to work together since `Bea` supervises `Tom`.

Provide a model of this problem that uses the `global_cardinality` constraint.

A. Provide a solution that satisfies all the constraints, i.e., for each nurse give his/her assignment (`morning`, `afternoon`, `night`, `off-duty`).

B. Identify the decision variables and the values of the problem, i.e., how do we model the fact that nurse $x \in \{$`Bea`, `Lea`, `Leo`, `Lio`, `Lili`, `Tom`$\}$ is assigned shift $y \in \{$`morning`, `afternoon`, `night`, `off-duty`$\}$?

C. Using a bipartite graph, draw the relations between the variables and the values identified in the previous question and display the solution you came up with in the first question.

D. Provide a model of the problem that uses a single `global_cardinality` constraint.

- Explain how the minimum/maximum capacity constraints (i.e., at least/at most) are modelled.
- Explain how each extra requirement is modelled in your solution.

[a]Hint: focus on *what is a variable* and *what is a value* in your model, and how to model the capacity constraints with `global_cardinality`.

SOLUTION TO EXERCISE 1

A. *Yes, since within* $\langle 2,4,2,2,1\rangle$, *values* 0, 1, 2, 3 *and* 4 *are respectively used zero, one, three, zero, and one times.*

B. *No, since within* $\langle 0,0,1,1\rangle$, *value* 2 *is not used one time.*

C. *Yes, since within* $\langle 2,3,4,5\rangle$, *value* 0, 1, 2, 3 *and* 4 *are respectively used zero, zero, one, one, and one times. The presence of a* 5 *in the solution does not matter since value* 5 *is not mentioned in the values of the second argument of the* `global_cardinality` *constraint.*

SOLUTION TO EXERCISE 2

the six solutions

$\langle V_1, V_2, V_3, V_4, V_5\rangle$ $\langle 1\ O_1, 2\ O_2, 3\ O_3\rangle$

① $(\langle 1,1,2,2,3\rangle, \langle 1\ \mathbf{2}, 2\ \mathbf{2}, 3\ \mathbf{1}\rangle)$

② $(\langle 1,2,1,2,3\rangle, \langle 1\ \mathbf{2}, 2\ \mathbf{2}, 3\ \mathbf{1}\rangle)$

③ $(\langle 1,2,2,2,3\rangle, \langle 1\ \mathbf{1}, 2\ \mathbf{3}, 3\ \mathbf{1}\rangle)$

④ $(\langle 2,1,1,2,3\rangle, \langle 1\ \mathbf{2}, 2\ \mathbf{2}, 3\ \mathbf{1}\rangle)$

⑤ $(\langle 2,1,2,2,3\rangle, \langle 1\ \mathbf{1}, 2\ \mathbf{3}, 3\ \mathbf{1}\rangle)$

⑥ $(\langle 2,2,1,2,3\rangle, \langle 1\ \mathbf{1}, 2\ \mathbf{3}, 3\ \mathbf{1}\rangle)$

SOLUTION TO EXERCISE 3

As suggested by the hint we go through the following steps:

A. [*RESTRICTING THE OCCURRENCE VARIABLES* $O_1, O_2, \ldots, O_5$]

(a) [*PRUNING WRT THE MAXIMUM NUMBER OF OCCURRENCES OF EACH VALUE*]
Since values 1, 2, 3, 4 *and* 5 *can respectively be assigned to at most* 3, 4, 5, 3 *and* 1 *decision variables (e.g., value* 1 *can only be assigned to* V_2, V_4 *and* V_5*) we have* $O_1 \leq \min(3,4)$, $O_2 \leq \min(4,1)$, $O_3 \leq \min(5,1)$, $O_4 \leq \min(3,5)$, *and* $O_5 \leq \min(1,4)$.

(b) [*PRUNING WRT* $\sum_{i=1}^{5} O_i = 5$ *AND THE DOMAIN OF* V_1]
Since we have five decision variables the sum of the occurrence variables is equal to five (i.e., $O_1 + O_2 + O_3 + O_4 + O_5 = 5$*). Since values* 2 *or* 3 *have to be assigned to the decision variable* V_1 *we have* $O_2 + O_3 \geq 1$. *It follows that* $O_1 + O_4 + O_5 \leq 4$. *Since* $O_1 \in [1,3]$, $O_4 \in [1,3]$ *and* $O_5 = 1$ *we get* $O_1 + O_4 \leq 3$ *and consequently* $O_1 \leq 2$ *and* $O_4 \leq 2$.

B. [*RESTRICTING THE DECISION VARIABLES* $V_1, V_2, \ldots, V_5$]
At the end of step **A** *we obtain* $O_1 \in [1,2]$, $O_2 \in [0,1]$, $O_3 \in [0,1]$, $O_4 \in [1,2]$, *and* $O_5 \in [1,1]$. *Since* $O_5 = 1$ *and since* V_2 *is the only decision variable that can be assigned value* 5 *we have* $V_2 = 5$. *Consequently* $V_1 \in [2,3]$, $V_2 \in [5,5]$, $V_3 \in [3,4]$, $V_4 \in [1,3]$, *and* $V_5 \in [1,4]$.

C. [*CHECKING FOR A SUPPORT*]
To show that no value can be removed from the domain of the decision and occurrence variables we show that every value that is still in the domain of a variable is part of a solution.

(a) *A solution with* $O_1 = 2$ *is* $V_1 = 2$, $V_2 = 5$, $V_3 = 4$, $V_4 = 1$, $V_5 = 1$ *and* $O_1 = 2$, $O_2 = 1$, $O_3 = 0$, $O_4 = 1$, $O_5 = 1$.

(b) *A solution with* $O_4 = 2$ *is* $V_1 = 2$, $V_2 = 5$, $V_3 = 4$, $V_4 = 1$, $V_5 = 4$ *and* $O_1 = 1$, $O_2 = 1$, $O_3 = 0$, $O_4 = 2$, $O_5 = 1$.

(c) *We now assume that* $O_1 = O_2 = O_3 = O_4 = O_5 = 1$, *i.e., all decision variables must be distinct. Without loos of generality we ignore* V_2, *which is fixed to* 5. *We provide a set of solutions where* V_1, V_3, V_4 *and* V_5 *can respectively be assigned to all the values of their domains:*

i. $V_1 = 2,\ V_3 = 3,\ V_4 = 1,\ V_5 = 4,$
ii. $V_1 = 2,\ V_3 = 4,\ V_4 = 3,\ V_5 = 1,$
iii. $V_1 = 2,\ V_3 = 4,\ V_4 = 1,\ V_5 = 3,$
iv. $V_1 = 3,\ V_3 = 4,\ V_4 = 1,\ V_5 = 2,$
v. $V_1 = 3,\ V_3 = 4,\ V_4 = 2,\ V_5 = 1.$

SOLUTION TO EXERCISE 4

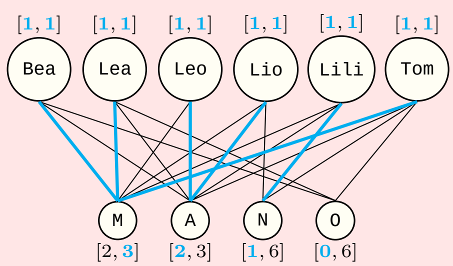
A. *A feasible solution is the following assignment*

$$\left(\begin{array}{lll} \texttt{Bea}:\texttt{morning} & \texttt{Lea}:\texttt{morning} & \texttt{Leo}:\texttt{afternoon} \\ \texttt{Lio}:\texttt{afternoon} & \texttt{Lili}:\texttt{night} & \texttt{Tom}:\texttt{morning} \end{array}\right) \textit{ since:}$$

- *the number of morning shifts is between* 2 *and* 3,
- *the number of afternoon shifts is between* 2 *and* 3,
- *the number of night shifts is at least* 1,
- `Bea`, `Lea` *and* `Leo` *are not assigned to a night shift,*
- `Leo`, `Lio` *and* `Lili` *work,*
- `Bea` *and* `Tom` *are both assigned the same shift.*

B. *To each nurse corresponds a variable whose initial domain is set to the types of shifts that nurse can actually perform (i.e., each shift type is encoded by a unique integer value).*

C. *The next figure provides a graphical representation of the assignment problem. To each nurse and to each shift type corresponds a vertex. There is an edge between a given nurse and a given shift type if and only if that nurse can perform that shift type. The solution given to question* **A** *is displayed with thick blue lines. The interval on top or below each vertex indicates the minimum and maximum number of edges that can reach the corresponding vertex in any solution; values in blue correspond to the number of edges of the displayed solution.*



D. *We get the following model*

$$\left\{\begin{array}{llll} \texttt{M}=1, & \texttt{A}=2, & \texttt{N}=3, & \texttt{O}=4, \\ \texttt{Bea}\in[\texttt{M},\texttt{O}], & \texttt{Lea}\in[\texttt{M},\texttt{O}], & \texttt{Leo}\in[\texttt{M},\texttt{O}], & \\ \texttt{Lio}\in[\texttt{M},\texttt{O}], & \texttt{Lili}\in[\texttt{M},\texttt{O}], & \texttt{Tom}\in[\texttt{M},\texttt{O}], & \\ O_{\texttt{M}}\in[2,3], & O_{\texttt{A}}\in[2,3], & O_{\texttt{N}}\in[1,6], & O_{\texttt{O}}\in[0,6], \\ \texttt{Bea}\neq\texttt{N}, & \texttt{Lea}\neq\texttt{N}, & \texttt{Leo}\neq\texttt{N}, & \\ \texttt{Leo}\neq\texttt{O}, & \texttt{Lio}\neq\texttt{O}, & \texttt{Lili}\neq\texttt{O}, & \\ \texttt{Bea}=\texttt{Tom}, & & & \\ \texttt{global\_cardinality} & \multicolumn{3}{l}{\left(\begin{array}{l} \langle\texttt{Bea},\texttt{Lea},\texttt{Leo},\texttt{Lio},\texttt{Lili},\texttt{Tom}\rangle, \\ \left\langle\begin{array}{ll} \texttt{val}-\texttt{M} & \texttt{occurrence}-O_{\texttt{M}}, \\ \texttt{val}-\texttt{A} & \texttt{occurrence}-O_{\texttt{A}}, \\ \texttt{val}-\texttt{N} & \texttt{occurrence}-O_{\texttt{N}}, \\ \texttt{val}-\texttt{O} & \texttt{occurrence}-O_{\texttt{O}} \end{array}\right\rangle \end{array}\right),} \end{array}\right.$$

where:

- *line 1 declares the integer value of each shift type,*
- *lines 2, 3 and 4 declare the nurse and occurrence variables,*
- *line 5 enforces* `Bea`, `Lea` *and* `Leo` *not to work on a night shift,*
- *line 6 imposes* `Leo`, `Lio` *and* `Lili` *to work,*
- *line 7 constrains* `Bea` *and* `Tom` *to work on the same shift,*
- *line 8 restricts each shift type to occur within a given range.*