The seesaw problem[21]

Example (seesaw)

Adam (36 kg), Boris (32 kg) and Cecil (16 kg) want to sit on a 10-foot long seesaw such that they are at least 2 feet apart and the seesaw is balanced.

Write a general model for any number of people.
Possible decision variables?

1. Position on the seesaw for each person.
2. Distances between persons, position of the first person, and order of persons.
3. Person or empty for each position on the seesaw.

Multiple modeling?

How to improve performance of our model?

[21] From R. Barták's practical

Symmetry breaking[22]

Add constraints to choose only one of symmetric variants of a (partial) assignment; many useful global constraints

- **Bin packing:** when trying to pack items into bins, any two bins that have the same capacity are symmetric.
- **Graph colouring:** When trying to assign colours to nodes in a graph such that adjacent nodes must have different colours, we typically model colours as integer numbers. However, any permutation of colours is again a valid graph colouring.
- **Vehicle routing:** if the task is to assign customers to certain vehicles, any two vehicles with the same capacity may be symmetric (this is similar to the bin packing example).
- **Rostering/time tabling:** two staff members with the same skill set may be interchangeable, just like two rooms with the same capacity or technical equipment.

[22] From The MiniZinc Handbook

Search annotations

Specify how to search: `solve::<annotation>`

`int_search(<variables>,<varchoice>,<constrainchoice>)`

- `<variables>` is a 1-dim array of `var int` ,
- `<varchoice>` is a variable choice annotation, and
- `<constrainchoice>` is a choice of how to constrain a variable.

Example: *n*-queens

```
solve::int_search(q, first_fail, indomain_min)
satisfy;
```

Similarly we have `bool_search,set_search` .

Search annotations: variable choice

- `input_order` choose in order from the array
- `first_fail` choose the variable with the smallest domain size
- `smallest` choose the variable with the smallest value in its domain
- `dom_w_deg` choose the variable with the smallest value of domain size divided by weighted degree, which is the number of times it has been in a constraint that caused failure earlier in the search.

See the documentation for more.

Search annotations: constrain choice

- `indomain_min` assign the variable its smallest domain value
- `indomain_median` assign the variable its median domain value
- `indomain_random` assign the variable a random value from its domain
- `indomain_split` bisect the variables domain excluding the upper half.

See the documentation for more.

Restart (and warm start)

Return to the top of the search tree (for nonedeterministic search strategies).

- `restart_constant(n)` restart after n nodes searched
- `restart_linear(n)` k-th restart after kn nodes
- `restart_geometric(b,n)` k-th restart after $n \cdot b^k$ nodes

Example:

```
solve::int_search(q, first_fail, indomain_random)
::restart_linear(1000) satisfy;
```

Warm start: supply a partial or suboptimal solution, or ranges for variables to start with (currently not supported in Gecode)

Choosing between models[23]

The better model is likely to have some of the following features:

- smaller number of variables, or at least those that are not functionally defined by other variables
- smaller domain sizes of variables
- more succinct, or direct, definition of the constraints of the model
- uses global constraints as much as possible

In reality all this has to be tempered by how effective the search is for the model. Usually the effectiveness of search is hard to judge except by experimentation.

[23] From The MiniZinc Handbook

Globalizer

The Holy Grail: anyone with domain knowledge can write (efficient!) models. Analyze the model and suggest global constraints.[24]

- https://www.minizinc.org/doc-2.5.0/en/globalizer.html
- Under development
- Only supports a subset of the language, no set or enum types, no command line data.
- Example: queens.mzn
  ```
  gcc(queens,[1,1,1,1,1,1,1,1]);) %no longer supported
  ```
  Instead:
  ```
  global_cardinality(queens,[i|i in 1..n],[1|i in 1..n]);
  ```
- Global cardinality constraints

[24] K. Leo et al, "Globalizing Constraint Models", CP'2013.