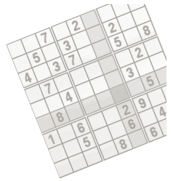


Constraint Programming

Practical Exercises

Roman Barták
Department of Theoretical Computer Science and Mathematical Logic

Constraint Modelling

- The smuggler has a knapsack with capacity **9 units**. It is possible to fill it in with **whisky bottles** (each consumes **4 units**), **perfume bottles** (each consumes **3 units**), and **cigarette boxes** (each consumes **2 units**). Any mix od products can be used. The profit from whisky is **15 dollars**, from perfumes **10 dollars**, and from cigarettes **7 dollars**. What can be placed to the knapsack if the required total profit is at least 30 dollars?
- Propose a constraint model to solve the problem.


```
:-use_module(library(clpfd)).
smuggler(Goods):-
  Goods = [W,P,C],
  domain(Goods,0,4),
  4*W + 3*P + 2*C #=< 9,
  15*W + 10*P + 7*C #>= 30,
  labeling([],Goods).
```

```
?- smuggler(G).
G = [0,1,3] ? ;
G = [0,3,0] ? ;
G = [1,1,1] ? ;
G = [2,0,0] ? ;
no
```

And what is the best possible solution?

– substitute the last two lines of code with

```
  15*W + 10*P + 7*C #= Cost,
  labeling([maximize(Cost)],Goods)
```


- Let the knapsack has capacity m. There are n products with given capacites and profits. Select the set of products fitting in the knapsack and maximizing the total profit.

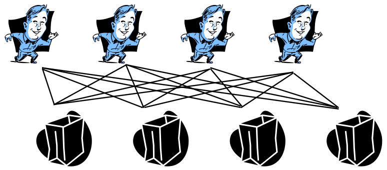
```
knapsack(+Size,+ListOfSizes,
  +ListOfCosts,-Goods,-Cost)
```

- The following constraint will be useful:
  - `scalar_product(Coeffs, Variables, RelOp, Value)`
  - Coeffs*Variables RelOp Value

Assignment problem

The problem:

There are 4 workers and 4 products and a table describing the efficiency of producing the product by a given worker. The task is assign workers to products (one to one) in such a way that the total efficiency is at least 19.



	P1	P2	P3	P4
W1	7	1	3	4
W2	8	2	5	1
W3	4	3	7	2
W4	3	1	6	3

A CSP model:

- W1,W2,W3,W4 in 1..4 — a product per worker
- all_different([W1,W2,W3,W4]) — different products
- $T_{1,W1}+T_{2,W2}+T_{3,W3}+T_{4,W4} \geq 19$ — total efficiency

Assignment problem - implementation

```
:-use_module(library(clpfd)).

assignment_p(Sol):-
    Sol = [W1,W2,W3,W4],

    domain(Sol,1,4),
    all_different(Sol),
    element(W1,[7,1,3,4],EW1),
    element(W2,[8,2,5,1],EW2),
    element(W3,[4,3,7,2],EW3),
    element(W4,[3,1,6,3],EW4),
    EW1+EW2+EW3+EW4 #>= 19,

    labeling([ff],Sol).
```

```
?- assignment_p(X).

X = [1,2,3,4] ? ;   19
X = [2,1,3,4] ? ;   19
X = [4,1,2,3] ? ;   21
X = [4,1,3,2] ? ;   20

no
```

Optimization using B&B

```
    EW1+EW2+EW3+EW4 #= E,

    maximize(labeling([ff],Sol),E).
```

```
?- assignment_p(X).

X = [4,1,2,3] ? ;

no
```

How does it work?

- **find** first **feasible instantiation** of variables
- **find better instantiation** of variables
- **repeat** until some instantiation of variables exists

Assignment problem - a dual model

Why do we **assign products to workers?**

Cannot we do it in an opposite way, that is, to **assign a worker to a product?**

Of course, we can **swap the role of values and variables!**

- This new model is called a **dual model.**

```
:-use_module(library(clpfd)).

assignment_dual(Products):-
    Products = [P1,P2,P3,P4],

    domain(Products,1,4),
    all_different(Products),
    element(P1,[7,8,4,3],EP1),
    element(P2,[1,2,3,1],EP2),
    element(P3,[3,5,7,6],EP3),
    element(P4,[4,1,2,3],EP4),
    EP1+EP2+EP3+EP4 #>= 19,

    labeling([ff],Products).
```

Number of choice points	
Primal model	15
Dual model	11

```
P1 in 1..2
P2 in 1..4
P3 in 2..4
P4 in 1..4
```

Which model is better?

- In this particular case, the dual model propagates earlier (thus it is assumed to be better).

Assignment problem - composing models

We can combine both primal and dual model in a single model to get better domain pruning.

```
:-use_module(library(clpfd)).

assignment_combined(Workers):-
    Workers= [W1,W2,W3,W4],
    domain(Workers,1,4),
    all_different(Workers),
    element(W1,[7,1,3,4],EW1),
    element(W2,[8,2,5,1],EW2),
    element(W3,[4,3,7,2],EW3),
    element(W4,[3,1,6,3],EW4),
    EW1+EW2+EW3+EW4 #>= 19,

    Products = [P1,P2,P3,P4],
    domain(Products,1,4),
    all_different(Products),
    element(P1,[7,8,4,3],EP1),
    element(P2,[1,2,3,1],EP2),
    element(P3,[3,5,7,6],EP3),
    element(P4,[4,1,2,3],EP4),
    EP1+EP2+EP3+EP4 #>= 19,

    assignment(Workers,Products),

    labeling([ff],Workers).
```

- **a primal model**

```
W1 in (1..2)\/{4}
W2 in 1..4
W3 in 2..4
W4 in 2..4
```

- **a dual model** (redundant)

```
P1 in 1..2
P2 in 1..4
P3 in 2..4
P4 in 1..4
```

- **a channelling constraint**
- labelling one model is enough

- Petra needs to move her flat. She has three friends that can help her, but only 60 minutes to do the job. The following table shows how many people are necessary to move a given item and how much time it takes.

item	time (min.)	people
piano	30	3
chair	10	1
bed	15	3
table	15	2

- When should we move each item?

Homework

Department of Theoretical Computer Science and Mathematical Logic
bartak@ktiml.mff.cuni.cz