STABLE MARRIAGE PROBLEM

1. Problem description

This problem appears in many areas, we quote the nice introduction from [4, p. 148]:

> *"Assume that two disjoint sets A and B of equal cardinality n are given. Find a set of n pairs (a, b) such that $a \in A$ and $b \in B$ satisfy some constraints. Many different criteria for such pairs exist; one of them is the rule called 'stable marriage rule.'*
>
> *Assume that A is a set of men and B is a set of women. Each man and each woman has stated distinct preferences for their partners. If the n couples are chosen such that there exists a man and a woman who are not married, but who would prefer each other to their actual marriage partners, then the assignment is said to be unstable. If no such pair exists, it is called* stable.
>
> *This situation characterizes many related problems in which assignments have to be made according to preferences, such as, for example, the choice of a school by pupils, the choice of recruits by different branches of the armed services, etc. The example of marriages is particularly intuitive; note, however, that the stated list of preferences is invariant and does not change after a particular assignment has been made. This rule simplifies the problem, but it also represents a distortion of reality (called abstraction)."*

This defines the input data as two-dimensional matrices of rankings, where women rank men and conversely. Within this ranking system, a marriage between a man m and woman w is thus considered stable given that

- whenever m ranks another female o higher than w, o prefers her man to m.
- whenever w ranks another male o higher than m, o prefers his wife to w.

2. OPL model

The problem model in figure 1 on the following page is taken from [2, sec. 2.2.3]. Table 1 and table 2 on the following page (also taken from [2, sec. 2.2.3]) present the instance data in terms of mutual ranking values.

	Richard	James	John	Hugh	Greg
Helen	1	2	4	3	5
Tracy	3	5	1	2	4
Linda	5	4	2	1	3
Sally	1	3	5	4	2
Wanda	4	2	3	5	1

Table 1. Instance data: female ranking

The Appendix on page 4 lists further sample instances, along with solutions.

Date: January 2004.

```
enum Women ...;
enum Men ...;

int rankWomen[Women,Men] = ...;
int rankMen[Men,Women] = ...;

var Women wife[Men];
var Men husband[Women];
solve {
     forall(m in Men)
        husband[wife[m]] = m;
     forall(w in Women)
        wife[husband[w]] = w;

     forall(m in Men & o in Women)
        rankMen[m,o] < rankMen[m,wife[m]] => rankWomen[o,husband[o]] < rankWomen[o,m];
     forall(w in Women & o in Men)
        rankWomen[w,o] < rankWomen[w,husband[w]] rankMen[o,wife[o]] < rankMen[o,w];
};
```

FIGURE 1. Stable marriage problem in OPL

	Helen	Tracy	Linda	Sally	Wanda
Richard	5	1	2	4	3
James	4	1	3	2	5
John	5	3	2	4	1
Hugh	1	5	4	3	2
Greg	4	3	2	1	5

TABLE 2. Instance data: male ranking

3. Z MODEL

We introduce the sets of *Men* and *Women* as given sets, in case of concrete data we could also use free types.

$[Men, Women]$

Furthermore, we note that in this problem the function that assigns a *wife* to a *husband* is injective, for every man must have a distinct wife, and, since all men and women have to be married, this function is also surjective. Thus, we can represent the function *wife* as bijection in the schema below. Consequently, we just have to declare *husband* as the inverse of *wife*, which covers the two complicated synchronization expressions at the beginning of the `solve` block in figure 1.

Stable_Marriage

$$rankWomen : Women \times Men \longrightarrow \mathbb{N}$$
$$rankMen : Men \times Women \longrightarrow \mathbb{N}$$
$$wife : Men \rightarrowtail\!\!\!\!\!\twoheadrightarrow Women$$
$$husband : Women \rightarrowtail\!\!\!\!\!\twoheadrightarrow Men$$

$$husband = wife^{\sim}$$
$$\forall\, m : Men;\ o : Women \mid rankMen(m, o) < rankMen(m, wife(m)) \bullet$$
$$\quad rankWomen(o, husband(o)) < rankWomen(o, m)$$
$$\forall\, w : Women;\ o : Men \mid rankWomen(w, o) < rankWomen(w, husband(w)) \bullet$$
$$\quad rankMen(o, wife(o)) < rankMen(o, w)$$

The remainder almost follows immediately from the model in figure 1.

4. Literature references

Apart from the OPL model in figure 1 on the preceding page, a discussion of the problem using the same instance data can also be found in [3, pp. 64–68]. Stepwise refinement of a PASCAL recursive backtracking program for the problem is documented in [4, sec. 3.6]. That program uses preference lists directly, as opposed to the ranking matrices of figure 1 on the facing page. Both formats can however easily be converted into another.

Two CSP encodings, a binary and a Boolean one, were introduced in [1] for the problem and a variation, which admits incomplete preference lists. In both cases, establishing arc-consistency was sufficient to warrant backtrack-free search for all solutions.

References

[1] Ian P. Gent, Robert W. Irving, David Manlove, Patrick Prosser, and Barbara M. Smith. A Constraint Programming Approach to the Stable Marriage Problem. In Toby Walsh, editor, *Proceedings of CP-01*, volume 2239 of *LNCS*, pages 225–239. Springer, 2001.

[2] Pascal Van Hentenryck. *The OPL Optimization Programming Language*. MIT Press, January 1999.

[3] ILOG, France. *Ilog Solver 4.4, User's Manual*, May 1999.

[4] Niklaus Wirth. *Algorithms + Data Structures = Programs*. Series in Automatic Computation. Prentice-Hall, 1976.

Appendix A. Sample instances and solutions

A.1. Solution for the named instance. The solution to this instance are three alphabetic lists of wives.

	Richard	James	John	Hugh	Greg
Solution 1	Tracy	Linda	Wanda	Helen	Sally
Solution 2	Tracy	Helen	Wanda	Linda	Sally
Solution 3	Sally	Helen	Tracy	Linda	Wanda

A.2. The same as before, but numbered.

Man	#1	#2	#3	#4	#5
`woman#1`	1	2	4	3	5
`woman#2`	3	5	1	2	4
`woman#3`	5	4	2	1	3
`woman#4`	1	3	5	4	2
`woman#5`	4	2	3	5	1

Woman	#1	#2	#3	#4	#5
`man#1`	5	1	2	4	3
`man#2`	4	1	3	2	5
`man#3`	5	3	2	4	1
`man#4`	1	5	4	3	2
`man#5`	4	3	2	1	5

The solution to this instance consists of three numeric lists of wives.

	`man#1`	`man#2`	`man#3`	`man#4`	`man#5`
Solution 1	2	3	5	1	4
Solution 2	2	1	5	3	4
Solution 3	4	1	2	3	5

A.3. A small 3×3 instance.

Man	#1	#2	#3
`woman#1`	3	1	2
`woman#2`	3	2	1
`woman#3`	3	2	1

Woman	#1	#2	#3
`man#1`	3	2	1
`man#2`	2	3	1
`man#3`	2	1	3

There is only one single solution $\{1 \mapsto 1, 2 \mapsto 3, 3 \mapsto 2\}$.

A.4. The 6×6 instance from [1].

Man	#1	#2	#3	#4	#5	#6
`woman#1`	1	5	4	6	2	3
`woman#2`	4	1	5	2	6	3
`woman#3`	6	4	2	1	5	3
`woman#4`	1	5	2	4	3	6
`woman#5`	4	2	1	5	6	3
`woman#6`	2	6	3	5	1	4

Woman	#1	#2	#3	#4	#5	#6
`man#1`	1	4	2	5	6	3
`man#2`	3	4	6	1	5	2
`man#3`	1	6	4	2	3	5
`man#4`	6	5	3	4	2	1
`man#5`	3	1	2	4	5	6
`man#5`	2	3	1	6	5	4

This one has three solutions.

	`man#1`	`man#2`	`man#3`	`man#4`	`man#5`	`man#6`
Solution 1	1	2	4	6	5	3
Solution 2	1	2	4	5	6	3
Solution 3	1	2	4	3	6	5