Binary Constraint Satisfaction Problems:
A “Telsell” Presentation

Jano van Hemert
jvhemert@liacs.nl
http://www.liacs.nl/~jvhemert

LIACS
Niels Bohrweg 1
2333 CA Leiden
The Netherlands
Constraint satisfaction

What is a constraint satisfaction problem?

Given a set of variables, assign from a set of values to each variable one value. A set of constraints prohibits certain combinations of value assignments to occur.

What makes a constraint satisfaction problem binary?

In binary constraint satisfaction problem the constraints only occurs between at most two variables.
Examples

✔ n-Queens: given a $n \times n$ chess board and $n$ queens, place the queens on the board such that no queen attacks another queen

✔ SAT: given a boolean formula, find an assignment of variables such that the formula evaluates to true

✔ Graph colouring: given a graph find a $k$-colouring of the nodes such that nodes connected are coloured with a different colour

\[
\begin{align*}
\text{domain} &= \{\text{red, blue, green}\} \\
\text{variables} &= \{x_1, x_2, x_3, x_4, x_5\} \\
\text{constraints} &= \{\text{adjacent nodes}\} \\
\text{solution} &= \{(x_1, \text{red}), (x_2, \text{blue}), (x_3, \text{red}), (x_4, \text{green}), (x_5, \text{red})\}
\end{align*}
\]
What do we (need to) know?

✔ Binary constraint satisfaction problems are generally classified under NP-complete (read: extremely difficult to solve)
✔ We know there exist incredibly extremely difficult to solve problem instances
✔ We can predict quite accurately where these hard problem instances can be found
Over to you, the customer

We have the problem instances, now you go solve them!
The things you get, documentation

✔ The Online Guide to Constraint Programming by Roman Barták (HTML, 1998)

✔ Assorted papers to help you get ideas, and a list to even more papers (PS, 1991–2001)

✔ Full web site of RandomCsp, the library you may use, comes with a complete manual and reference guide (HTML & PS, 2002)

✔ These slides (PS & PDF, 2002)

✔ Another (more detailed, less Telsell) presentation (PS & PDF, 2002)
The things you get, for you to work with

✔ A set of problem instances that are currently used in empirical research
✔ RandomCsp library setup and ready to go
✔ Documented results to compare with
✔ An example to show the basic usage of the library
✔ An experiments manager that takes care of doing all the experiments for you
#include <static_csp.h>
#include <strstream>

int main (int argc, char * argv [])
{
    istrstream input ( argv[1] ); // Read in random seed
    int RandomSeed = 0; input >> RandomSeed; // Set random seed
    StaticCsp csp(argc); // Read in CSP instance

    ValueT * solution = new ValueT [csp.GetNumberOfVariables() * sizeof(ValueT)]; // Create a random solution
    for (unsigned int i = 0; i < csp.GetNumberOfVariables(); i++)
    {
        solution[i] = (ValueT) (csp.GetDomainSize(i) * (rand() / (RAND_MAX + 1.0))); // Create a random solution
    }

    cout << csp.GetNumberOfConflicts(solution) << ","; // Output number of conflicts
    for (unsigned int i = 0; i < csp.GetNumberOfVariables(); i++)
    {
        cout << solution[i] << " "; // Output solution
    }
    cout << "," << RandomSeed << "," << argv[2] << endl; // Output random seed and CSP filename
    return 0;
}
It really *is* easy to use

To run this example first do a `make` then start the experiment manager with the appropriate experiments:

```
./experiment.pl ../problem_instances/n=15-m=15
```

Output looks like this:

```
conflicts, solution, random seed, problem file
```

```
16,12 5 11 13 2 5 11 4 8 9 7 14 1,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
10,10 1 1 5 6 10 0 8 9 14 4 1 14 2,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
14,8 3 5 6 4 2 12 13 3 6 2 9 14 3,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
11,15 2 3 3 5 6 9 2 4 7 1 8 10 13 4,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
13,4 0 1 4 2 1 4 7 13 0 4 1 14 6 5,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
10,2 14 11 6 1 12 9 3 4 24 7 14 13 11 6,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
11,7 13 3 0 7 14 2 9 0 7 1 11 12 15 ,7,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
15,5 11 13 0 14 3 13 12 13 9 8 4 10 5 13 ,5,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
12,3 10 3 5 13 6 10 9 10 0 2 13 12 9,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
4,8 9 7 12 3 8 6 4 0 1 14 11 3 ,10,../problem_instances/n=15-m=15/p_e=12/00.solvable.mv15.d15.c_t0.12_x1.csp
```
Blatant advertisement

☞ Serious problem and serious work
☞ All the boring stuff has been done
☞ You just focus on creating a novel solving method
☞ Leaving you with plenty of time for fun