Rina Dechter, fall-99

ICS 275a, Projects

Usually, benchmarks used for testing constraint processing algorithms are either randomly generated csps, or real problems. I propose three problems:

- Golumb ruler
- Combinatorial auctions
- Bandwidth allocation in communication networks
- propose your own

A generic project: Your task is to study one of the above problems, discuss ways for modeling the problem as constraint satisfaction (or as cnfs) and/or as constraint optimization (see also homework 5) and then propose (and implement yourself, or use existing software) several (at least two) ways of solving the problem. Then, generate test cases and run experiments comparing performances. One of the task will be to add the problems to the benchmark collection maintained by Toby Walsh (see links on web page).

Below is a list algorithms (some will be covered in the next 1-2 weeks).

Algorithms

1. Backtracking search:
   1. Backjumping (Graph-based, conflict-directed)
   2. Dynamic variable ordering (DVFC)
   3. Backjumping + Learning (Graph-based, or conflict-based
   4. Partial arc-consistency look-ahead
   5. Full arc-consistency look-ahead
   6. Full path-consistency look-ahead.
   7. Eclipse
   8. Branch and bound for constraint optimization
   9. The DPLL algorithm for cnfs.

2. Inference methods:
   1. Arc-consistency (AC-2,AC-3)
   2. Path-consistency (PC-2)
   3. Adaptive-consistency
   4. Directional consistency followed by backtracking, (e.g. directional arc-consistency (DAC), or directional path-consistency (DPC))

3. Stochastic local search methods (SLS): GSAT, Walksat.

4. Hybrids of search and inference: (The cycle-cutset method)

Step 1: Write a one page description of the project that you choose. During the last week of classes we will present the project, You will write a report... and we will send it to CP2000...

Examples of specific projects

1. Implement algorithms AC2 and AC3 using, C, or C++ or Java in a way that can be placed on the web and be used by others. (and accompany the book?) Choose benchmarks (from the above, or others) to apply and compare the algorithms and run a set of experiments.

2. Implement PC-2. Select benchmarks, both randomly generated and based on the problems above and run experiments. Compare backtracking on top of path-consistency

3. Select a problem domain. Then, describe 2 ways for modeling the problem as a csps (part of homework 5). Discuss the virtues in each formulation. Pick one for implementation.

   Implement (or use existing software) backjumping (conflict-directed) and implement a solver using Eclipse. Generate test cases (for Golumb ruler there is only a single problem) run experiments and compare.

4. As above, but compare Dynamic variable ordering Forward-checking style with dynamic variable ordering with arc-consistency as preprocessing.

5. (To be elaborated later) Try various ordering algorithms to test a good algorithm for finding w*.