T-79.4001 Seminar on Theoretical Computer Science

Ilkka Niemelä

21.1.2009

Outline

Practicalities

- **Prerequisites**: T-79.3001/144 Logic in computer science: foundations (or corresponding knowledge)
- **Seminars**: Wednesdays 12–14, TB353
- **Coordinator**: Prof. Ilkka Niemelä, TB337, phone 451 3290, e-mail: Ilkka.Niemela@tkk.fi.
- **Homepage:**
  `https://noppa.tkk.fi/kurssi/t-79.4001/`
  All course information is available on the Noppa home pages of the course and all course announcements in Noppa news.
- **Seminar material**:
  Handbook of Constraint Programming. P. Rossi, P. Van Beek, and T. Walsh (Eds.). Elsevier, 2006.
- Talks cover chapters: 3,4,5,6,11,...

Goals of the Course

- Learning key methods and approaches to constraint programming
- Improving skills to read and exploit research results from scientific literature
- Learning how to structure and prepare of a longer technical talk with slides
- Practising how to give a convincing presentation of technical results
- Identifying key features of effective presentations

Passing the Course

Each student

- gives a 45 min seminar talk with archivable slides (preferably in PDF) and
- provides feedback for three other presentations by filling in a feedback form.

- The course grade is determined by the grade of the seminar talk (85%) and the grades of the three feedback reports (5% each).

Practical Arrangements

- Preliminary slides for the presentation must be sent by email to the coordinator (Ilkka.Niemela@tkk.fi) by Tuesday noon of the presentation week. The slides will then be made available by the coordinator at the home page of the course.
- The language of the presentation is Finnish or English. The slides can be in English even if the oral presentation is in Finnish.
- The final (revised) slides must be sent to the coordinator in two days after the presentation (i.e., by next Friday noon).

Practical Arrangements

- Each student must provide feedback on three presentations according to the assignments at the home page of the course.
- The feedback is given by filling in the feedback form available on the course home page in text form (using Latex for mathematics) and sending the form to the coordinator by email.
- The feedback is also graded and will be taken into account in the final grade of the reviewing student.
- The feedback form must be returned in one week (i.e., by next Wednesday noon).

Preparing your talk

- The idea is that you use considerable amount of time (c. one week) for preparing your talk. Remember that the course is 3 cr.
- Read carefully the material for your talk and identify the main points; consult further literature given in the references when necessary.
- Plan the structure of your presentation and decide what can be presented and at what level of detail in the allocated time.
- Develop illustrations and examples clarifying the main concepts and results.
- Prepare and polish the slides
- Rehearse and time the talk
- Hint: `http://latex-beamer.sourceforge.net/` LaTeX (beamer) examples of slides.

Introduction to Constraint Programming

- Constraint programming: an effective approach to solving combinatorial search problems
- Examples of combinatorial search problems: n-queens, graph coloring, travelling salesperson, knapsack, bin packing, . . .
- Constraint programming uses techniques from many areas including artificial intelligence, computer science, databases, programming languages, and operations research.
- Applied successfully, for example, in scheduling, planning, routing, configuration, networks, and bioinformatics.

Constraint Programming

- In constraint programming a (combinatorial search) problem is solved by
  - **modelling** (encoding) the problem using a set of **constraints** on a set of **variables** such that a solution to the constraints (a value assignment to the variables satisfying all constraints) gives a solution to the original problem and
  - using a **constraint solver** to find a solution to the constraints.

Example

The n-coloring problem. Given a graph (V, E) introduce

- for each node $i \in V$ a variable X_i with a domain $\{1, \ldots, n\}$,
- for each edge $(i, j) \in E$ a constraint $X_i \neq X_j$.

Now a value assignment to variables X_i satisfying all constraints gives a valid n-coloring of the graph (V, E).

Constraint solvers

- A constraint solver takes a set of constraints on a set of variables with given domains and finds a solution to the constraints.
- **Systematic solvers** interleave **search** and **constraint propagation** where information contained in one constraint is propagated to variables domains and other constraints.
- Constraint propagation is important because it may reduce the parts of the search space that need to be visited.
  Example. Consider a constraint $x < y$ where $x \in [50..200], y \in [0..100]$. Using a simple propagation rule it can be inferred that $x \in [50..99], y \in [51..100]$.
- **Local solvers** search for solutions by moving from one candidate solution to another using local changes guided by a heuristic function.

Key issues

- For solving a problem efficiently using constraint programming the key issues are
  - choosing a suitable **modelling** technique and
  - appropriate (global) **constraints** together with
  - **constraint propagation** techniques and
  - **search methods**.
- Seminar talks address these topics.