Department of Computing and Information Systems
The University of Melbourne
COMP90046 Constraint Programming
Second Semester, 2017

Course Summary

Introduction

Welcome to Constraint Programming (COMP90046 2017 SM2).

Constraint programming is a new high-level language paradigm expressly developed for solving complex combinatorial satisfaction and optimization problems. Solving such problems requires searching through a very large space of possible solutions to find a solution or an optimal solution. In the constraint programming paradigm, constraints are used to limit the search as much as possible. Hence, the two main components in a constraint programming system are the constraint solver and the search engine which implements some strategy, such as backtracking, for exploring the search space.

In this course we will investigate methods to solve combinatorial problems. We will begin by learning how to model the problems in a high level manner. We will then learn how the underlying solving technologies solve these problems, which will allow us to write better models that take into account the strengths and weaknesses of the solving technology we use.

Objectives

When you complete the subject you should be able to:

- model constraint satisfaction and optimization problems of reasonable complexity using a modelling language;
- explain (to a senior computer science student) how some constraint solvers work (e.g. linear programming, finite domain propagation, Boolean satisfaction);
- use the MiniZinc modelling language to model integer constraint problems;
- evaluate the suitability of a particular constraint model for solving a problem;
- program different effective search strategies for combinatorial problems;
- improve the execution of a constraint program by reasoning about its search behaviour.

When you complete the subject you should have some appreciation of the uses of constraint programming and the wealth of combinatorial problems. You should also have improved analytical, problem-solving, programming, and team-working skills.

Lectures

Lectures will be held on Mondays from 12:00 - 13:00 in Theatre 2 of the Alan Gilbert Building (Building 104). There are workshops Tuesdays 11:00–12:00 (Alice Hoy 101) and Fridays 11:00–12:00 (Alice Hoy 333) which we shall not always use. There will be 12 hours of lectures. The lecturer is

- Peter Stuckey (phone: 8344-1341, e-mail: pstuckey@unimelb.edu.au, office 6.19).

Syllabus

Modelling: simple modelling, modelling with data structures, predicates, global constraints, effective modelling. Constraint solving methods: Finite domain propagation, linear programming, mixed integer programming, Programming search and optimization.

Recommended Texts

- (Recommended) MiniZinc Tutorial: `http://www.minizinc.org/downloads/doc-latest/minizinc-tute.pdf` A tutorial on MiniZinc, with many detailed examples.
- (Recommended) The OPL Optimization Programming Language. Pascal Van Hentenryck, MIT Press. 1999. *A commercial product similar to MiniZinc*
- (Recommended) Programming with Constraints: an Introduction. Kim Marriott and Peter J. Stuckey, MIT Press. 1998. *Introduction to Constraint Logic Programming*
- (Recommended) Operations Research: Applications and Algorithms. Wayne L. Winston, Brooks Cole, 1998. *Introduction to linear programming, mixed integer programming*
- (Recommended) Principles of Constraint Programming. Krzysztof Apt. Cambridge. 2003. *A highly theoretical book, interesting and well written.*

MiniZinc

MiniZinc and the MiniZincIDE are available for download from `http://www.minizinc.org/`. IMPORTANT: You should download and install MiniZinc and the MiniZincIDE on your laptop before the course commences!

You may also want to download and install various other solvers for use with MiniZinc including

- Gecode: an open source CP solver `www.gecode.org` (preinstalled)
- Chuffed: an open source lazy clause generation CP solver `www.github.com/chuffed/chuffed` (preinstalled)
- OR-tools: an open source CP solver from Google `https://code.google.com/p/or-tools/`
- Gurobi: a commercial Mixed integer programming solver `www.gurobi.com` (free academic licenses available).

Assessment

There will be eight assignments/projects. An exam will be held at the end of the term. The exam will count for 70% of your final grade. The remaining 30% will come from the project, with the marks split between assignments/projects. The usual rules for passing apply, but in addition at least 35/70 must be scored for the exam.

Checklist

- ☐ Check you can see COMP90046_2017_SM2 on the LMS
- ☐ Download and install MiniZinc 2.1.x and the MiniZincIDE from `www.minizinc.org`
- ☐ Enrol in Coursera (`www.coursera.org`) using your University of Melbourne account