Search Problems and Algorithms
T-79.4201

Ilkka Niemelä & Pekka Orponen

Laboratory for Theoretical Computer Science, TKK

Autumn 2007

T–79.4201 Search Problems and Algorithms (4 ECTS)

“An introduction to the fundamental concepts, techniques and tools used in dealing with large, weakly structured combinatorial search spaces.”

Required course in the A2-level Study Module in TCS.

Why this course?

- With the increase in computing power, continually new computation-intensive application areas emerge (e.g. various types of planning & scheduling, data mining, bioinformatics, . . .)
- Many immediate problems in these areas are both computationally demanding & mathematically weakly structured (“Here is my messy objective function. Find a near-optimal solution to it – quickly!”)
- In such “quick-and-dirty” settings a search problem formulation is often the most effective (if not the only) approach.
- Moreover, the design and analysis of search algorithms is a fascinating research topic in itself!

Practical arrangements

Lectures: Thu 14-16 TB353, alternately by Ilkka Niemelä and Pekka Orponen

Tutorials: Fhu 16–18 TB353, André Schumacher

Registration: by TOPI

Prerequisites: Basic knowledge of problem representations and logic, facility in programming, data structures and algorithms

Requirements: Examination (20 Dec) and three small programming assignments (announced 4 Oct, 18 Oct, 8 Nov, each due in two weeks)

Course home page:
`http://www.tcs.hut.fi/Studies/T-79.4201/`

Grading scheme

Exam: max 40 points

Programming: max 3*5 = 15 points

Tutorials: max 5 points

Total: 60 points

For the programming assignments, a correctly functioning program, returned on time with appropriate work description yields 3 points; the remaining 2 points are allocated based on an efficiency competition among the submitted programs.

Material

No existing textbook: lectures cover a wide range of material from several textbooks & current scientific literature.

Course problems based on lecture slides; updated on the course web site each week after lecture.

Examples of reference material:

- Aarts & Lenstra (Eds.), *Local Search in Combinatorial Optimization.* Wiley 1997.
- Apt, *Principles of Constraint Programming.* Cambrigde University Press, 2003.
- T. Bäck, *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, 1996.
- Hoos & Stützle, *Stochastic Local Search: Foundations and Applications.* Morgan Kaufmann 2005.

1 Overview of the Course

1.1 A Motivating Example

Twelve slightly different types of billets, numbered 1 ... 12, arrive for processing at a factory workshop. The workshop has four machines, numbered I ... IV, and four workers, named A ... D, who have different qualifications for working on the billets. To make things more complicated, there are also four specialised tools, numbered i ... iv, that are needed for processing the various billets. The requirements of machines, tools, and workers for the billets are indicated in the following table:

Machine				Tool				Worker			
I:	1	5	9	i:	1	2	3	A:	1	7	8
II:	2	6	10	ii:	4	9	10	B:	2	3	4
III:	3	7	11	iii:	5	11	12	C:	5	6	12
IV:	4	8	12	iv:	6	7	8	D:	9	10	11

Let's say processing each billet by a combination of the appropriate machine, tool & worker requires 1 hour. Any given machine, tool, or worker can only work on one billet at a time. Since there are 12 billets and 4 machines (as well as tools & workers), processing all the billets requires at least 3 hours. Can it be done in this minimal time?

How would you approach the preceding problem:

(a) By hand? (Design an appropriate schedule!)

(b) By computer, assuming that an arbitrary list of requirements such as above would be given as input? (The numbers of machines, tools, and workers do not need to be the same: this is just a peculiarity of the present example.)

Think about this problem; it will be discussed at next week's tutorial. You do not need to write any program code, but try to think about how you would approach task (b) of minimising the completion time for a given list of requirements.

Lecture 2: Combinatorial search and optimisation problems

I.N. 20 Sep

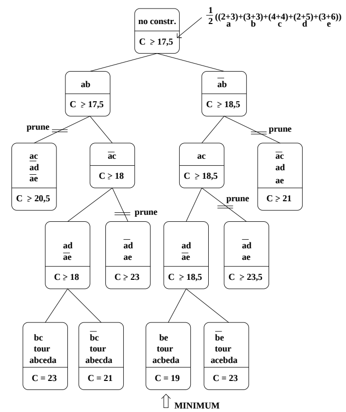
Common mathematical patterns in combinatorial search and optimisation: Satisfiability, Clique, Graph Colouring, Traveling Salesman, Set Cover.

Different types of problems and reductions between them.

Lecture 3: Search spaces and objective functions. Complete search methods

P.O. 27 Sep

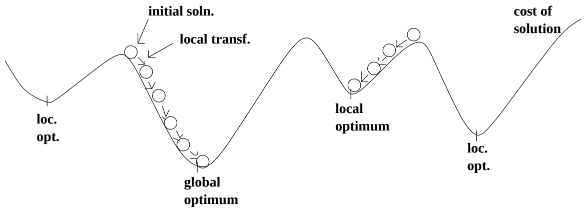
Search spaces and objective functions. Backtrack search. Branch-and-bound search. The A* algorithm.



Lecture 4: Local search techniques

P.O. 4 Oct

Search spaces as “fitness landscapes”. Neighbourhoods and local search. Lin-Kernighan search for TSP. Simulated annealing. Tabu search. Record-to-Record Travel. Local search methods for satisfiability. Instructions for the 1st programming assignment.

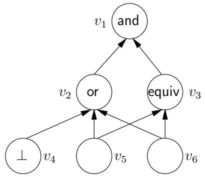


Lecture 5: Constraint satisfaction: formalisms and modelling

I.N. 11 Oct

General representation of search problems as systems of constraints (e.g. propositional formulas or Boolean circuits)

$$(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_4) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$$



Case studies of translations.

Lecture 6 : Constraint satisfaction: algorithms

I.N. 18 Oct

Basic algorithmic framework for solving CSPs. The DPLL algorithm for Boolean constraints in conjunctive normal form. Software tools for constraint satisfaction. Instructions for the 2nd programming assignment.

$$\frac{x \in \{a_1, \ldots, a_k\}}{x \in \{a_1\} \mid \ldots \mid x \in \{a_k\}}$$

Lecture 7: Constraint satisfaction, linear & integer programming

I.N. 1 Nov

Constraint satisfaction by local search; WalkSAT revisited. Tools for SAT and CSP. Representation of CSPs as (mixed) linear programs i.e. systems of linear equations and inequality constraints over reals and integers.

$$\begin{array}{lllllllllllllll}
\min & 2x_2 + x_4 - 5x_7 & \text{s.th.} \\
x_1 & + & x_2 & + & x_3 & + & x_4 & & & & & & & = & 4 \\
x_1 & & & & & & & - & x_5 & & & & & \geq & 2 \\
 & & & & x_3 & & & & & + & x_6 & & & \geq & 3 \\
 & & 3x_2 & + & x_3 & & & & & & & + & x_7 & = & 6 \\
\multicolumn{3}{l}{x_1, \ldots, x_7 \geq 0} \\
\multicolumn{3}{l}{x_2, x_4, x_7 \ \text{integers}}
\end{array}$$

Lecture 8: Linear and integer programming: modelling and tools

I.N. 8 Nov

Normal and standard forms for linear programs. Modelling CSP as mixed linear programs. Software tools for MLP. Instructions for the 3rd programming assignment.

$$
\begin{array}{rl}
\min \displaystyle\sum_{j=1}^{n} c_j x_j & \text{s.th.} \\
\displaystyle\sum_{j=1}^{n} a_{ij} x_j \geq b_i, & i = 1, \ldots, m \\
x_j \geq 0, & j = 1, \ldots, n
\end{array}
$$

Lecture 9: Linear and integer programming: algorithms

I.N. 15 Nov

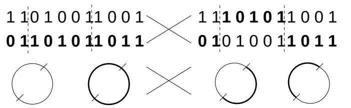
Solving MLPs by relaxations and branch-and-bound search. The simplex algorithm for pure LPs.

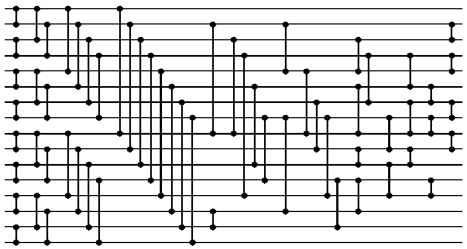
	x_1	x_2	x_3	x_4	x_5
$-\frac{9}{2}$	$\frac{3}{2}$	0	$\frac{3}{2}$	0	0
$\frac{1}{2}$	$\frac{3}{2}$	1	$\frac{1}{2}$	0	0
$\frac{5}{2}$	$\frac{7}{2}$	0	$\frac{1}{2}$	1	0
$\frac{3}{2}$	$-\frac{11}{2}$	0	$-\frac{3}{2}$	0	1

Lecture 10: Genetic algorithms

P.O.. 22 Nov

Genetic algorithms: the simple GA, hyperplane sampling, schema theorem. Coevolutionary genetic algorithms.

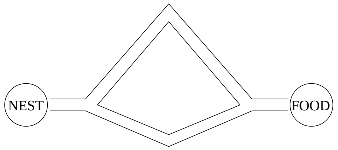


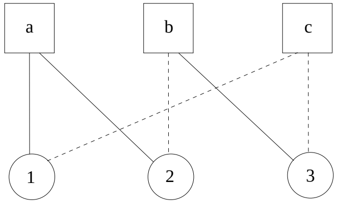


Lecture 11: Novel methods

P.O. 29 Nov

Ant colony optimisation. Belief propagation.





Lecture 12: Complexity of search

P.O. 13 Dec

The “No Free Lunch” theorem. Combinatorial Phase Transitions. Complexity of Local Search