DM841

Discrete Optimization

Lecture 1

Course Introduction

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

1. Motivation

2. Course Organization

You

- Background
  study program
  programming skills
  other optimization course?
- Expectations

Outline

1. Motivation

2. Course Organization

Main Aim of the Course

To enable the student to solve
discrete, satisfaction and optimization problems
that arise in practical applications
by means of constraint programming,
heuristics and metaheuristics

Discrete and Combinatorial Constrained Optimization

- Discrete optimization emphasizes the difference from continuous optimization, solutions are described by integer numbers or discrete structures
- Combinatorial optimization is a subfield of discrete optimization.
- Combinatorial optimization is the study of the ways discrete structures (eg, graphs) can be selected/arranged/combined: Finding an optimal object from a finite set of objects.
- Discrete/Combinatorial Optimization involves finding a way to efficiently allocate resources in mathematically formulated problems.
- We will assume discrete variables with finte domains.

Satisfaction and Optimization Problems: Different Classifications

- Unconstrained vs Constrained
- Discrete vs Continuous vs Mixed
- Glass box problems (can be formalized mathematically) vs Black box problems
- Linear, Quadratic, Conic, Semidefinite, Convex vs Non Convex
- Derivative-free (non-smooth) vs Smooth
- Global vs Local
- Direct search vs model based
- Deterministic vs Stochastic
- Single objective vs Multiobjective

Discrete Satisfaction and Optimization Problems

They arise in many areas of
Computer Science, Bioinformatics, Artificial Intelligence, Operations Research, Management Science, Economics, ...:

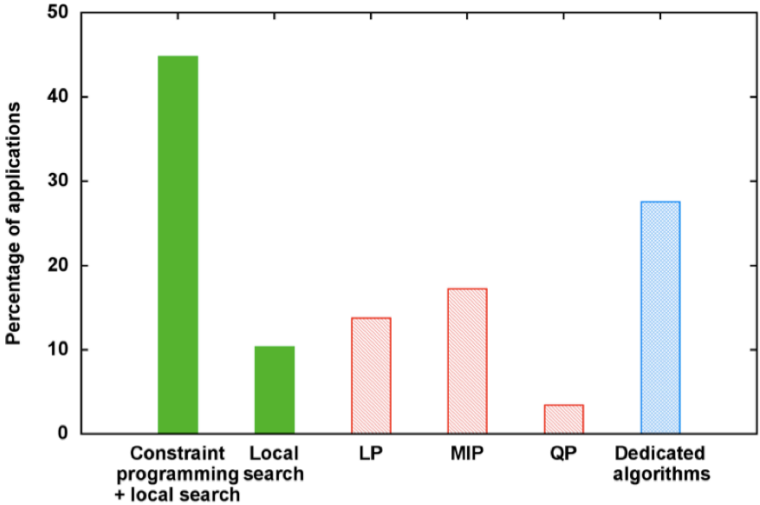
- allocating register memory
- planning, scheduling, timetabling
- Internet data packet routing
- protein structure prediction
- auction winner determination
- portfolio selection
- ...

Applications

- Operation research (optimization problems)
- Graphical interactive systems (to express geometrical correctness)
- Molecular biology (DNA sequencing, 3D models of proteins)
- Finance
- Circuit verification
- Elaboration of natural languages (construction of efficient parsers)
- Scheduling of activities
- Configuration problem in form compilation
- Generation of coerent music programs [**?**].
- Data bases
- ...
- http://hsimonis.wordpress.com/

Applications

Distribution of technology used at Google for optimization applications developed by the operations research team



Discrete Optimization Problems

Simplified models are often used to formalize real life problems

- finding models of propositional formulae (SAT)
- finding variable assignment that satisfy constraints (CSP)
- partitioning graphs or digraphs
- partitioning, packing, covering sets
- finding shortest/cheapest round trips (TSP)
- coloring graphs (GCP)
- finding the order of arcs with minimal backward cost
- ...

Example Problems

- They are chosen because conceptually concise, intended to illustrate the development, analysis and presentation of algorithms
- Although real-world problems tend to have much more complex formulations, these problems capture their essence

Elements of Combinatorial Problems

Combinatorial problems are characterized by an input,
i.e., a general description of conditions (or constraints) and parameters,
and a question (or task, or objective) defining
the properties of a solution.

They involve finding a grouping, ordering, or assignment
of a discrete, finite set of objects that satisfies given conditions.

Candidate solutions are combinations of objects or solution components that need not satisfy all given conditions.

Feasible solutions are candidate solutions that satisfy all given conditions.

Optimal Solutions are feasible solutions that maximize or minimize some criterion or objective function.

Approximate solutions are feasible candidate solutions that are not optimal but good in some sense.

Applied Character

Optimization problems are very challenging, seldom solvable exactly in polynomial time and no single approach is likely to be effective on all problems.
Solving optimization problems remains a very experimental endeavor: what will or will not work in practice is hard to predict.

Hence, the course has applied character:

- We will learn the theory
- but also implement some models $\rightsquigarrow$ programming in Python and MiniZinc
- and solve them with solvers that implement what we learn in the theory
- and later even design and implement our own (heuristic) solvers

Expected prerequisites

Students taking the course are expected to be able to:

- work with algorithms and data structures
- assess the complexity of the algorithms with respect to runtime and space consumption
- implement the algorithms by programming

Outline

1. Motivation

2. Course Organization

Course Organization

Part 1: CP

- Modeling Problems in CP
- Local Consistency
- Constraint Propagation
- Search
- Symmetry Breaking
- Random Restarts
- SAT, SMT, ASP solving

Part 2: (Meta)heuristics

- Local search
- Stochastic local search
- Metaheuristics
- Experimental analysis
- Parameter Tuning / Algorithm Configuration

Schedule

- Class schedule:
  - See course web page.
  - mitsdu.sdu.dk

- Working load:
  - 10 ECTS = 250 working hours
  - Currently scheduled, 3 x 15 classes = 45 classes = 90 hours
  - Intro phase (Introfase): 2 x 15 classes = 30 classes = 60 hours. (58 hours)
  - Exercises (Træningsfase): 15 classes = 30 hours. (30 hours)
  - Study phase: (Studiefase) ?? hours

COVI19 Situation

Online vs on Campus?

COVID19 is transmitted through aerosols. We have enough evidence.
We need to:

- Be outdoors
- Wear fitted, quality masks
- Improve indoor ventilation

Assessment

- Obligatory Assignments:
  Two preparation assignments per part
  One final per part
- Preparation assignments must be passed.
- Final assignments graded with 7-grade scale + internal censor
  grade is weighted (?) average of the two final assignments.
- Preparation assignments can be prepared in pairs $\rightsquigarrow$ Feedback
- Final assignments is individual and only limited communication is allowed.

Learning Objectives

For a top perfomance the student must demonstrate ability to:

- model a problem similar in nature to the ones seen in the course within the framework of constraint programming and heuristics
- argue about the different modeling choices arising from the theory behind the components of constraint programming, including global constraints, propagators, search and branching schemes and the design choices in heuristics including complexity analysis
- develop a solution prototype in a constraint programming system or an heuristic algorithm from scratch in a general purpose programming language (python?)
- undertake an experimental analysis, report the results and draw sound conclusions based on them
- describe the work done in an appropriate language including mathematical formalism

Content of the Graded Assignments

- Modeling
- Implementation (deliverable and checkable source code)
- Written description
- (Analytical) and experimental analysis
- Performance counts! Comparison

Competences wrt Degree

- plan and carry out scientific projects at the high professional level including managing work and development situations that are complex, unpredictable and require new solutions
- describe, analyze and solve advanced computational problems using the learned models
- analyze the advantages and disadvantages of various algorithms, especially in terms of resource consumptions
- elucidate the hypotheses of qualified theoretical background and critically evaluate own and others' research and scientific models
- develop new variants of the methods learned where the specific problem requires
- communicate through a written report research based knowledge and discuss professional and scientific problems with peers
- give expertise in discrete optimization and solution methods from the international research front

Further Skills

- working in groups
- creativity
- independence

Communication media

- Public Web Page [WWW] $\Leftrightarrow$ ItsLearning [ITS]
  (link from `http://www.imada.sdu.dk/u/marco/DM841/`)
- Announcements in BlackBoard
- Resources in [ITS] (unless linked from [WWW])
- Discussion Board (anonymous) in [ITS]
- Personal email: `marco@imada.sdu.dk`
- Office visits
- (A-bit-earlier-than) Mid term evaluation in class
- Others? Group chat?

Literature

Resources

Books

Excerpts from the following books will be used.

- [W] Mark Wallace, Building Decision Support Systems using MiniZinc, Springer, 2020
- [SMT] Peter J. Stuckey, Kim Marriott, Guido Tack. MiniZinc Handbook. 2022
- [A] Krzysztof R. Apt, Principles of Constraint Programming, Cambridge University Press, 2010, ISBN 9780521125499 (errata.ps)
- [RBW] F. Rossi, P. van Beek and T. Walsh (ed.), Handbook of Constraint Programming, Elsevier, 2006
- [Hoo] J.N. Hooker, Integrated Methods for Optimization. Springer, 2007
- [BHM] Biere, Armin; Heule, Marijn; Maaren, Hans van. Handbook of satisfiability Frontiers in Artificial Intelligence and Applications, First edition from 2009 or Second edition from 2021.
- [MAK] W. Michiels, E. Aarts and J. Korst. Theoretical Aspects of Local Search. Springer Berlin Heidelberg, 2007
- [GP] Michel Gendreau and Jean-Yves Potvin (eds) Handbook of Metaheuristics 2019. International Series in Operations Research & Management Science book series (ISOR, volume 272)

Other References

To be updated during the course.

Links

- MiniCP
- Global constraints In MiniZinc
- Collection on Constraints by Gene Freuder
- Global Constraint Catalog by Nicolas Beldiceanu and Sophie Demassey. Full List and CP Systems.
- CSPLib, a problem library for CP
- Hakan's page on solution scripts to 242 CP problems
- [ASlib] Algorithm Selection Library

MOOC Courses

- [C1] EdX's Constraint Programming Instructors: Pierre Schaus, Laurent MIchel, Pascal Van Henteryck
- [C2] Coursera's Discrete Optimization. Instructor: Pascal Van Hentenryck.
- [C3] Coursera's Basic Modeling for Discrete Optimization Instructors: Peter James Stuckey, Jimmy Ho Man Lee
- [C4] Coursera's Advanced Modeling for Discrete Optimization Instructors: Peter James Stuckey, Jimmy Ho Man Lee
- [C5] Coursera's Solving Algorithms for Discrete Optimization Instructor: Jimmy Ho Man Lee, Peter James Stuckey

Associations and Conferences

- Association for Constraint Programming
- International Conference on Principles and Practice of Constraint Programming
- International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)

Agreement for the Exercise Sessions

- Read the recommended material after the meeting in the class
- If you encounter difficulties then take note of the question and bring it in the class; it may be useful also for others
- The meaning with the exercise classes is for you to get feedback, not to deliver new material
- All questions and comments are welcome
- There is not stupid/wrong question, we all learn from mistakes.
- I can ask questions to everybody and it is not to punish someone. You can well say pass.

Class format

Be prepared for:

- Flipped classes: learn content at home, engage with material in class
- Problem solving in class
- Hands on experience with modeling and programming
- Discussion on exercises for home
- Experimental analysis of performance

These activities will be announced

They require study phase (= work outside the classes)