DM826 – Spring 2011

Modeling and Solving Constrained Optimization Problems

Lecture 1

Course Introduction
Hybrid Optimization

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

[*Slides by Stefano Gualandi, Politecnico di Milano*]

Outline

Outline

1. Course Introduction
2. Overview
3. Hybrid Modelling

Schedule and Material

- Schedule (28 lecture hours):
  - Monday 10.15-14
  - Wednesday 10.15-14
  - Friday 8.15-10 (exercises, questions)
  - Last lecture: Friday, 18th March, 2011
- Communication tools
  - Course Public Webpage (Wp) $\Leftrightarrow$ Blackboard (Bb)
  - Announcements (Bb)
    (link from `http://www.imada.sdu.dk/~marco/DM826/`)
  - Documents (Photocopies) in Bb
  - Personal email in Bb
  - You are welcome to visit me in my office in working hours (8-16).

Evaluation

- Two obligatory assignments (50% of final grade)
  - Model
  - Implementation
  - Report (3 pages)
- Final project (50% of final grade)
  - Model
  - Implement
  - Report (Max 10 pages)

References

- Main References:

  - B1 J.N. Hooker, Integrated Methods for Optimization. Springer, 2007
  - B2 F. Rossi, P. van Beek and T. Walsh (ed.), Handbook of Constraint Programming, Elsevier, 2006
  - B3 C. Schulte, G. Tack, M.Z. Lagerkvist, Modelling and Programming with Gecode 2010
  - B4 Apt, K. R. Principles of Constraint Programming Cambridge University Press, 2003
  - B5 Marriott, K. & Stuckey, P. J. Programming with Constraints: An Introduction MIT Press, 1998
  - B6 P. van Hentenryck and M. Milano. Hybrid Optimization, The Ten Years of CPAIOR. Springer, 2011, 45

- Photocopies (from Course Documents left menu of Blackboard)
- Articles from the Webpage
- Lecture slides
- Assignments

- ...but take notes in class!

Outline

Computational Model

We basically have three Computational Model to solve (combinatorial) optimization problems:

- Mathematical Programming (LP, ILP, QP, SDP, ...)
- Constraint Programming (SAT as a very special case)
- Local Search (... and Meta-heuristics)

Constraint Programming

- In MILP we formulate problems as a set of linear inequalities
- In CP we describe substructures (so-called global constraints) and combine them with various combinators.
- Substructures capture building blocks often (but not always) comptuationally tractable by special-purpose algorithms
- CP models can:
  - be linearized and solved by their MIP solvers;
  - be translated in CNF and sovled by SAT solvers;
  - be handled by local search
- In MILP the solver is often seen as a black-box
  In CP and LS solvers leave the user the task of programming the search.
- CP = model + propagation + search
  constraint propagation by domain filtering $\leadsto$ inference
  search = backtracking, branch and bound, local search

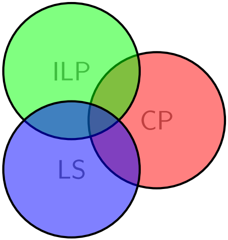
Hybrid Methods

Strengths:

- CP is excellent to explore highly constrained combinatorial spaces quaickly
- Math programming is particulary good at deriving lower bounds
- LS is particualry good at derving upper bounds

How to combine them to get better "solvers"?

- Exploiting OR algorithms for filtering
- Exploiting LP (and SDP) relaxation into CP
- Hybrid decompositions:
    1. Logical Benders decomposition
    2. Column generation
    3. Large-scale neigbhrohood search



Outline

1. Course Introduction
2. Overview
3. Hybrid Modelling

Integrated Modeling

Models interact with solution process hence models in CP and IP are different.

To integrate one needs:

- to know both sides
- to have available a modelling language that allow integration (Comet)

There are typcially alternative ways to formulate a problem. Some may yield faster solutions.

Typical procedure:

- begin with a strightforward model to solve a small problem instance
- alter and refine the model while scaling up the instances to maintain tractability

Linear Programming

Linear Programming

Given A matrix $A \in \mathbb{R}^{m \times n}$ and column vectors $b \in \mathbb{R}^m, c \in \mathbb{R}^n$.

Task Find a column vector $x \in \mathbb{R}^n$ such that $Ax \leq b$ and $c^T x$ is maximum, decide that $\{x \in \mathbb{R}^n \mid Ax \leq b\}$ is empty, or decide that for all $\alpha \in \mathbb{R}$ there is an $x \in \mathbb{R}^n$ with $Ax \leq b$ and $c^T x > \alpha$.

Theory vs. Practice

In theory the Simplex algorithm is exponential, in practice it works.

In theory the Ellipsoid algorithm is polynomial, in practice it is not better than the Simplex.

Integer Programming

Integer Programming

Given A matrix $A \in \mathbb{Z}^{m \times n}$ and vectors $b \in \mathbb{Z}^m, c \in \mathbb{Z}^n$.

Task Find a vector $x \in \mathbb{Z}^n$ such that $Ax \leq b$ and cx is maximum,
or decide that $\{x \in \mathbb{Z}^n \mid Ax \leq b\} = \emptyset$,
or decide that $\sup\{cx \mid x \in \mathbb{Z}^n, \, Ax \leq b\} = \infty$.

Theory vs. Practice

In theory, IP problems can be solved efficiently by exploiting (if you can find-/approximate) the convex hull of the problem.

In practice, we heavily rely on branch&bound search tree algorithms, that solve LP relaxations at every node.

Logical Statements Frequently (but not always) the integer variables are restricted to be in {0,1} representing Yes/No decisions.

Quadratic Programming

Quadratic Programming

Given: Matrices $A, Q_i \in \mathbb{R}^{n \times n}$, with $i = 0, \dots, q$, and column vectors $a_i, b, c \in \mathbb{R}^n$.

Task: Find a column vector $x \in \mathbb{R}^n$ such that $x^T Q_i x + a_i^T x \leq b$ and $x^T Q_0 X + c^T x$ is maximum,
or decide that $\{x \in \mathbb{R}^n \mid x^T Q_i x + a_i^T x \leq b\}$ is empty,
or decide that it is unbounded.

Theory vs. Practice

In theory, this is a richer modeling language (quadratic constraints and/or objective functions).
In practice, we linearize all the time, relying on (most of the time linear) cutting plane algorithms.

In practice

Modeling Languages (e.g., AMPL, Mosel, AIMMS, ZIMPL, Comet, OPL,...)

Write your problem as:

$$\min\{\mathbf{c}^T\mathbf{z} + \mathbf{d}^T\mathbf{y} \mid A\mathbf{z} + B\mathbf{y} \geq b, z \in \mathbb{R}^n, y \in \mathbb{Z}\}$$

push the button solve, and ... cross your fingers!

Theory vs. Practice

In theory, plenty of optimization problem solved in this manner.
In practice, for many real-life discrete (optimization) problems this approach is not suitable (typically, it does not scale well).

The case of Integer Programming

The problem with Integer Programming [Williams [2010]]

IP is essentially concerned with the intersection of two structures:

- Linear inequalities giving rise to polytopes.
- Lattices of integer points.

Mathematical and computational methods and results exist for both these structures on their own. *However mixing them is like mixing oil and water*. Problems arise in both the computation of optimal solutions and the economic interpretation of the results.

Example:

How many times do we really have (an approximation of) the convex hull in our integer problem?

First example: Send More Money

Send + More = Money

You are asked to replace each letter by a different digit so that

$$\begin{array}{cccccc} & S & E & N & D & + \\ & M & O & R & E & = \\ \hline M & O & N & E & Y & \end{array}$$

is correct. Because S and M are the leading digits, they cannot be equal to the 0 digit.

Can you model this problem as an IP?

Send More Money: ILP model 1

- $x_i \in \{0, \dots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- $\delta_{ij} \begin{cases} 0 & \text{if} x_i < x_j \\ 1 & \text{if} x_j < x_i \end{cases}$
- Crypto constraint:

$$\begin{array}{cccccc} & 10^3 x_1 & +10^2 x_2 & +10 x_3 & +x_4 & + \\ & 10^3 x_5 & +10^2 x_6 & +10 x_7 & +x_2 & = \\ \hline 10^4 x_5 & +10^3 x_6 & +10^2 x_3 & +10 x_2 & +x_8 & \end{array}$$

- Each letter takes a different digit:

$$\begin{aligned} x_i - x_j - 10\delta_{ij} &\leq -1, \quad \text{all} i, j,\ i < j \\ x_i - x_j + 10\delta_{ij} &\leq 9, \quad \text{all} i, j,\ i < j \end{aligned}$$

Send More Money: ILP model 2

- $x_i \in \{0, \dots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- $y_{ij} \in \{0, 1\}$ for all $i \in I$, $j \in J = \{0, \dots, 9\}$
- Crypto constraint:

$$\begin{array}{cccccc} & 10^3 x_1 & +10^2 x_2 & +10 x_3 & +x_4 & + \\ & 10^3 x_5 & +10^2 x_6 & +10 x_7 & +x_2 & = \\ \hline 10^4 x_5 & +10^3 x_6 & +10^2 x_3 & +10 x_2 & +x_8 & \end{array}$$

- Each letter takes a different digit:

$$\sum_{j \in J} y_{ij} = 1, \qquad \forall i \in I,$$

$$\sum_{i \in I} y_{ij} \leq 1, \qquad \forall j \in J,$$

$$x_i = \sum_{j \in J} j y_{ij}, \qquad \forall i \in I.$$

Send More Money: ILP model

The quality of these formulations depends on both the tightness of the LP relaxations and the number of constraints and variables (compactness)

- Which of the two models is tighter?
  project out all extra variables in the LP so that the polytope for LP is in the space of the x variables. By linear comb. of constraints:

Model 1	Model 2
$-1 \leq x_i - x_j \leq 10 - 1$	$\sum_{j \in J} x_j \geq \frac{\|J\| (\|J\| - 1)}{2}, \quad \forall J \subset I,$ $\sum_{j \in J} x_j \leq \frac{\|J\| (2k - \|J\|) + 1}{2}, \quad \forall J \subset I.$

- Can you find the convex hull of this problem?
  Williams and Yan [2001] proove that model 2 is facet defining

Suppose we want to maximize MONEY, how strong is the upper bound obtained with this formulation? How to obtain a stronger upper bound?

Send More Money: ILP model (revisited)

- $x_i \in \{0, \ldots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- Crypto constraint:

$$\begin{array}{rrrrrr}
 & 10^3 x_1 & +10^2 x_2 & +10 x_3 & +x_4 & + \\
 & 10^3 x_5 & +10^2 x_6 & +10 x_7 & +x_2 & = \\
\hline
10^4 x_5 & +10^3 x_6 & +10^2 x_3 & +10 x_2 & +x_8 &
\end{array}$$

- Each letter takes a different digit:

$$\sum_{j \in J} x_j \geq \frac{|J|\,(|J|-1)}{2}, \qquad \forall J \subset I,$$

$$\sum_{j \in J} x_j \leq \frac{|J|\,(2k-|J|)+1}{2}, \qquad \forall J \subset I.$$

But exponentially many!

Constraint Programming

The **domain** of a variable x, denoted $D(x)$, is a finite set of elements that can be assigned to x.

A **constraint** C on X is a subset of the Cartesian product of the domains of the variables in X, i.e., $C \subseteq D(x_1) \times \cdots \times D(x_k)$. A tuple $(d_1, \ldots, d_k) \in C$ is called a solution to C.
Equivalently, we say that a solution $(d_1, ..., d_k) \in C$ is an assignment of the value d_i to the variable $x_i, \forall 1 \leq i \leq k$, and that this assignment satisfies C.
If $C = \emptyset$, we say that it is inconsistent.

When a constraint C is defined on a set X of k variables together with a certain set p of l parameters, we will denote it by $C(X, p)$, and regard it to be a set of k-tuples.
(and not $k + l$-tuples).

Constraint Programming

Constraint Satisfaction Problem (CSP)

A CSP is a finite set of variables X, together with a finite set of constraints C, each on a subset of X. A **solution** to a CSP is an assignment of a value $d \in D(x)$ to each $x \in X$, such that all constraints are satisfied simultaneously.

Constraint Optimization Problem (COP)

A COP is a CSP P defined on the variables $x_1, \ldots, x_n$, together with an objective function $f : D(x_1) \times \cdots \times D(x_n) \rightarrow Q$ that assigns a value to each assignment of values to the variables. An **optimal solution** to a minimization (maximization) COP is a solution d to P that minimizes (maximizes) the value of $f(d)$.

Global Constraint: `alldifferent`

Global constraint:

set of more elementary constraints that exhibit a special structure when considered together.

`alldifferent` constraint

Let $x_1, x_2, \ldots, x_n$ be variables. Then:

$$\texttt{alldifferent}(x_1, ..., x_n) = \{(d_1, ..., d_n) \mid \forall i, d_i \in D(x_i), \quad \forall i \neq j,\ d_i \neq d_j\}.$$

Note: different notation and names used in the literature

Send More Money

Constraint Programming model

Send + More = Money

- $X_i \in \{0, \ldots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- Crypto constraint:

$$\begin{array}{rrrrrr}
 & 10^3 X_1 & +10^2 X_2 & +10 X_3 & +X_4 & + \\
 & 10^3 X_5 & +10^2 X_6 & +10 X_7 & +X_2 & = \\
\hline
10^4 X_5 & +10^3 X_6 & +10^2 X_3 & +10 X_2 & +X_8 &
\end{array}$$

- Each letter takes a different digit:

$$\texttt{alldifferent}([X_1, X_2, \ldots, X_8]).$$

Send More Money

Constraint-based Local Search model

Send + More = Money

- $X_i \in \{0, \dots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- Crypto constraint:

$$\begin{array}{rrrrrc} & 10^3 X_1 & +10^2 X_2 & +10 X_3 & +X_4 & + \\ & 10^3 X_5 & +10^2 X_6 & +10 X_7 & +X_2 & = \\ \hline 10^4 X_5 & +10^3 X_6 & +10^2 X_3 & +10 X_2 & +X_8 & \end{array}$$

- Each letter takes a different digit:

$$\text{alldifferent}([X_1, X_2, \dots, X_8]).$$

The convex hull of `alldifferent`

Convex Hull of of alldifferent

Given a set $I = \{1, \dots, n\}$ (variable indices) and a set $D = \{0, \dots, k\}$ with $k \geq n$, we consider

$$\text{alldifferent}([x_1, \dots, x_n]), \text{ with } 0 \leq x_i \leq k.$$

all the facets of the previous ILP formulation for the `alldifferent` constraint are

$$\sum_{j \in J} x_j \geq \frac{|J|\,(|J| - 1)}{2}, \qquad \forall J \subset I,$$

$$\sum_{j \in J} x_j \leq \frac{|J|\,(2k - |J|) + 1}{2}, \qquad \forall J \subset I.$$

[Williams and Yan [2001]]

ILP model + CP propagation

- $x_i \in \{0, \ldots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$
- $y_{ij} \in \{0, 1\}$ for all $i \in I$, $j \in J = \{0, \ldots, 9\}$

$$\begin{array}{cccccc} & 10^3 x_1 & +10^2 x_2 & +10 x_3 & +x_4 & + \\ & 10^3 x_5 & +10^2 x_6 & +10 x_7 & +x_2 & = \\ \hline 10^4 x_5 & +10^3 x_6 & +10^2 x_3 & +10 x_2 & +x_8 & \end{array}$$

-

$$\sum_{j \in J} y_{ij} = 1, \qquad \forall i \in I,$$

$$\sum_{i \in I} y_{ij} \leq 1, \qquad \forall j \in J,$$

$$x_i = \sum_{j \in J} j y_{ij}, \qquad \forall i \in I.$$

- Propagation adds valid inequalities:

$$LB(X_i) \leq x_i \leq UB(X_i) \text{ for all } i \in I$$

.

Send More Money: CP model (revisited)

- $X_i \in \{0, \ldots, 9\}$ for all $i \in I = \{S, E, N, D, M, O, R, Y\}$

$$
\begin{array}{rrrrrl}
 & 10^3 X_1 & +10^2 X_2 & +10 X_3 & +X_4 & + \\
 & 10^3 X_5 & +10^2 X_6 & +10 X_7 & +X_2 & = \\
\hline
10^4 X_5 & +10^3 X_6 & +10^2 X_3 & +10 X_2 & +X_8 &
\end{array}
$$

- $$\text{alldifferent}([X_1, X_2, \ldots, X_8]).$$

- Redundant constraints

$$
\begin{aligned}
X_4 + X_2 &= 10\, r_1 + X_8, \\
X_3 + X_7 + r_1 &= 10\, r_2 + X_2, \\
X_2 + X_6 + r_2 &= 10\, r_3 + X_3, \\
X_1 + X_5 + r_3 &= 10\, r_4 + X_6, \\
+r_4 &= X_5.
\end{aligned}
$$

Can we do better? Can we propagate more?

Send More Money: LP relaxation

```
Solver<LP> lp();
var<LP>{float} y[Letters, Domain](lp, 0..1);
var<LP>{float} x[Letters](lp, Domain);
var<LP>{float} S = x[1];

maximize<lp>
   10000 * M + 1000 * O + 100 * N + 10 * E + Y
subject to {
  lp.post( S >= 1 );
  lp.post( M >= 1 );
  lp.post(             1000 * S + 100 * E + 10 * N + D +
               1000 * M + 100 * O + 10 * R + E ==
   10000 * M + 1000 * O + 100 * N + 10 * E + Y );

  forall (j in Domain)
    lp.post( sum(i in Letters) y[i,j] <= 1);

  forall (i in Letters) {
    lp.post( sum(j in Domain) y[i,j] == 1 );
    lp.post( x[i] == sum(j in Domain) j*y[i,j] );
  }
}
```

Send More Money: CP model

```
range Letters = 1..8;
range Domain = 0..9;

Solver<CP> cp();
var<CP>{int} r[1..4](cp, 0..1);
var<CP>{int} x[Letters](cp, Domain);
var<CP>{int} S = x[1];  [...]

solve<cp> {
  cp.post( S != 0 );
  cp.post( M != 0 );
  cp.post(            1000 * S + 100 * E + 10 * N + D +
                      1000 * M + 100 * O + 10 * R + E ==
           10000 * M + 1000 * O + 100 * N + 10 * E + Y );

  cp.post( alldifferent(x) );

  cp.post(  S + M + r[3] == O + 10*r[4] );
  cp.post(  E + O + r[2] == N + 10*r[3] );
  cp.post(  N + R + r[1] == E + 10*r[2] );
  cp.post(  D + E        == Y + 10*r[1] );
}
```

References

Hooker J.N. (2011). **Hybrid modeling**. In *Hybrid Optimization*, edited by P.M. Pardalos, P. van Hentenryck, and M. Milano, vol. 45 of **Optimization and Its Applications**, pp. 11–62. Springer New York.

Williams H. (2010). **The problem with integer programming**. Tech. Rep. LSE0R 10-118, London School of Economics and Political Science.

Williams H. and Yan H. (2001). **Representations of the all_different predicate of constraint satisfaction in integer programming**. *INFORMS JOURNAL ON COMPUTING*, 13(2), pp. 96–103.