Mathematical Programming as a Complement to Bio-Inspired Optimization

Ofer M. Shir (Tel-Hai College & Migal Institute, ISRAEL)

ofersh@telhai.ac.il

15th Int'l Conf. on Parallel Problem Solving from Nature — PPSN2018: Tutorial
September 2018, Coimbra, PORTUGAL

about the presenter

Ofer Shir is a Senior Lecturer (CS Dept) at Tel-Hai College, and a Principal Investigator at Migal Research Institute – Upper Galilee, ISRAEL.

Previously:

- IBM-Research
- Princeton University:
  Postdoctoral Research Associate
- PhD in CS: Leiden-U
  adv. : Th. Bäck & M. Vrakking;
  BSc in CS&Phys at Hebrew-U

why are we here?

- Global optimization has been for several decades addressed by algorithms and Mathematical Programming (MP) — branded as Operations Research (OR), yet rooted at Theoretical CS [1].
- Also – it has been treated by dedicated heuristics ("Soft Computing") – where EC resides (!)
- These two branches complement each other, yet practically studied under two independent CS disciplines

further motivation

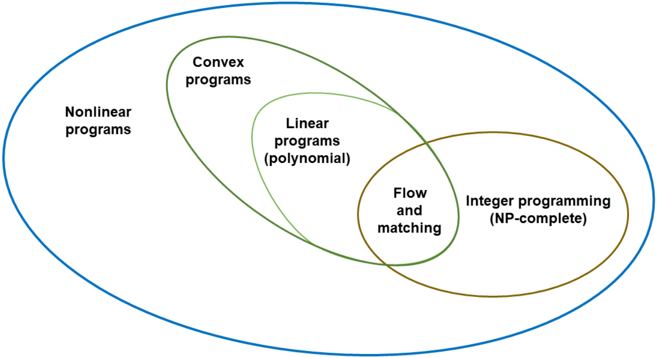
EC scholars become stronger, better-equipped researchers when obtaining knowledge on this so-called "optimization complement"

Commonly-encountered **misbeliefs**:

- *"if the problem is non-linear, there is no choice but to employ a Randomized Search Heuristic"*
- *"if it's a combinatorial NP-complete problem, EAs are the most reasonable option to approach it"*
- *"neither Pareto optimization nor uncertainty is/are addressed by OR"*
- *"OR is the art of giving bad answers to problems, to which, otherwise worse answers are given"*

outline

1. MP fundamentals
   - LP and polyhedra
   - simplex and duality
   - the ellipsoid algorithm
   - discrete optimization
2. MP in practice
   - solving an LP
   - basic modeling using OPL
   - QP
   - TSP
3. extended topics
   - robust optimization
   - multiobjective exact optimization
   - hybrid metaheuristics
4. discussion



Mathematical Programming: fundamentals

based on (i) MIT's "Optimization Methods" course material by D. Bertsimas, (ii) "Combinatorial Optimization" by Ch. Papadimitriou & K. Steiglitz, and (iii) IBM's ILOG/OPL tutorials and documentation.

the field of operations research

- Developed during WW-II: mathematicians assisted the US-army to solve hard strategical and logistical problems; mainly planning of operations and deployment of military resources. Due to the strong link to military *operations*, the term *Operations Research* was coined.
- Post-war: knowledge transfer into industry
- Roots: linear programming (LP), pioneered by George B. Dantzig
- Dantzig worked for the US-government, formulating the generalized LP problem, and devising the Simplex algorithm for tackling it. He also pursued an academic career (Berkeley, Stanford)

mathematical optimization

- Partitioning into 2 main approaches: constraints programming (CP) *versus* mathematical programming (MP). CP is concerned with constraints satisfaction problems, which possess no explicit objective functions (sometimes because impossible to model)

- MP includes the following techniques:
  linear programming (LP)
  integer programming (IP)
  mixed-integer programming (MIP)
  quadratic programming (QP) and mixed-integer QP (MIQP)
  nonlinear programming (NLP)

the canonical optimization problem

The general nonlinear problem formulated in the canonical form [2]:

$$\boxed{\begin{aligned} &\text{minimize}_{\vec{x}}\ f(\vec{x}) \\ &\text{subject to: } g_1(\vec{x}) \geq 0 \\ &\qquad\qquad\qquad \vdots \\ &\qquad\qquad\qquad g_m(\vec{x}) \geq 0 \\ &\qquad\qquad\qquad h_1(\vec{x}) = 0 \\ &\qquad\qquad\qquad \vdots \\ &\qquad\qquad\qquad h_\ell(\vec{x}) = 0 \end{aligned}} \tag{1}$$

solving the general problem

- Convexity:

  $f : \mathcal{S} \to \mathbb{R}$

  The function is convex **iff** $\forall s_1, s_2 \in \mathcal{S}$, $\lambda \in \mathbb{R}$

  $$f\left(\lambda s_1 + (1-\lambda)\, s_2\right) \leq \lambda f\left(s_1\right) + (1-\lambda)\, f\left(s_2\right)$$

  $f\left(\vec{x}\right)$ is concave if $-f\left(\vec{x}\right)$ is convex.

- The problem is called a *convex programming problem* when
  - i f is convex
  - ii g_i are all concave
  - iii h_j are all linear

- Strongest property: local optimality implies global optimality
- Sufficient conditions for optimality exist (Kuhn-Tucker)

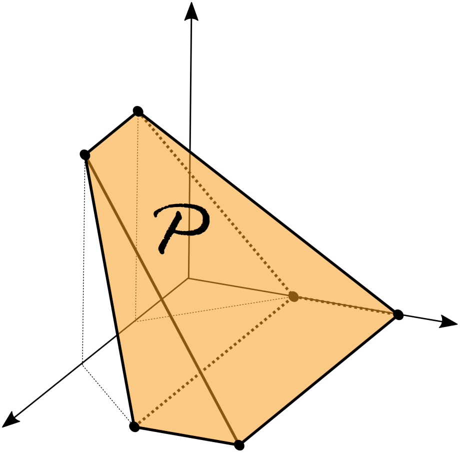
linear programming: standard form

When f and the constraints are all linear, an LP is posed in the **standard form** (minimization, equality constraints, non-negative variables):

$$\boxed{\begin{aligned} &\text{minimize}_{\vec{x}}\ \vec{c}^T\vec{x} \\ &\text{subject to: } \mathbf{A}\vec{x} = \vec{b} \\ &\qquad\qquad\quad \vec{x} \geq 0 \end{aligned}} \tag{2}$$

polyhedra

- A **hyperplane** is defined by the set
$$\{\vec{x} : \vec{a}^T \vec{x} = \vec{b}\}$$
- A **halfspace** is defined by the set
$$\{\vec{x} : \vec{a}^T \vec{x} \geq \vec{b}\}$$
- A **polyhedron** is constructed by the intersection of many halfspaces.
- The finite set of candidate solutions is the set of vertices of the **convex polyhedron** (*polytope*) defined by the linear constraints!
- Thus, solving any LP reduces to selecting a solution from a finite set of candidates ⇒ the problem is **combinatorial** in nature.



geometry of LP

Given a *polytope*

$$\mathcal{P} := \left\{\vec{x} : \mathbf{A}\vec{x} \le \vec{b}\right\}$$

- $\vec{x} \in \mathcal{P}$ is an **extreme point** of $\mathcal{P}$ if

$$\nexists \vec{y}, \vec{z} \in \mathcal{P}\,(\vec{y} \ne \vec{x}, \vec{z} \ne \vec{x}): \quad \vec{x} = \lambda\vec{y} + (1-\lambda)\,\vec{z},\ 0 < \lambda < 1$$

- $\vec{x} \in \mathcal{P}$ is a **vertex** of $\mathcal{P}$ if $\exists \vec{c} \in \mathbb{R}^n$ such that $\vec{x}$ is a unique optimum

$$\begin{array}{l} \text{minimize } \vec{c}^T\vec{y} \\ \text{subject to: } \vec{y} \in \mathcal{P} \end{array}$$

- $\vec{x} \ge \vec{0} \in \mathbb{R}^n$ is a **basic feasible solution** (**BFS**) iff $\mathbf{A}\vec{x} = \vec{b}$ and exist indices $\mathcal{B}_1, \ldots, \mathcal{B}_m$ such that:
  (i) the columns $\mathbf{A}_{\mathcal{B}_1}, \ldots, \mathbf{A}_{\mathcal{B}_m}$ are linearly independent
  (ii) if $\jmath \ne \mathcal{B}_1, \ldots, \mathcal{B}_m$ then $x_\jmath = 0$

polytopes and LP

"Corners" definitions: equivalence theorem

$\mathcal{P} := \left\{\vec{x} : \mathbf{A}\vec{x} \leq \vec{b}\right\}$; let $\vec{x} \in \mathcal{P}$.

$$\vec{x} \text{ is a vertex } \iff \vec{x} \text{ is an extreme point } \iff \vec{x} \text{ is a BFS}$$

See, e.g., [3] for the proof.

Conceptual LP search:

- begin at any "corner"
- **while "corner" is not optimal** hop to its neighbouring "corner" as long as it improves the objective function value

the basic simplex

$t \leftarrow 0$; *opt*, *unbounded* $\leftarrow$ `false`, `false`
$\vec{x}_t \leftarrow$ `constructBFS()`, $\mathbf{B} \leftarrow [\mathbf{A}_{\mathcal{B}_1}, \ldots, \mathbf{A}_{\mathcal{B}_m}]$
while *!opt && !unbounded* **do**
  if $\bar{c}_j := c_j - \vec{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j \geq 0 \;\; \forall j$ **then** *opt* $\leftarrow$ `true`
  else
    select any j such that $\bar{c}_j < 0$
    if $\vec{u} := \mathbf{B}^{-1} \mathbf{A}_j \leq \vec{0}$ **then** *unbounded* $\leftarrow$ `true`
    else
      $\vec{x}_{t+1} \leftarrow$ pivot on $\vec{x}_t$ `/* details omitted */`
      set new basis $\mathbf{A}_j$ `/* details omitted */`
      $t \leftarrow t + 1$
    end
  end
end

output: $\vec{x}_t$

duality

i. Every LP has an associated problem known as its **dual**; min turns into max, each constraint in the primal has an associated dual variable:

$$\begin{aligned} \text{minimize}_{\vec{x}} \quad & \vec{c}^T \vec{x} \\ \text{subject to: } & \mathbf{A}\vec{x} = \vec{b} \\ & \vec{x} \geq 0 \end{aligned} \qquad \begin{aligned} \text{maximize}_{\vec{p}} \quad & \vec{p}^T \vec{b} \\ \text{subject to: } & \vec{p}^T \mathbf{A} \leq \vec{c}^T \end{aligned}$$

$$\begin{aligned} \text{minimize}_{\vec{x}} \quad & \vec{c}^T \vec{x} \\ \text{subject to: } & \mathbf{A}\vec{x} \geq \vec{b} \end{aligned} \qquad \begin{aligned} \text{maximize}_{\vec{p}} \quad & \vec{p}^T \vec{b} \\ \text{subject to: } & \vec{p}^T \mathbf{A} = \vec{c}^T \\ & \vec{p} \geq 0 \end{aligned}$$

ii. The dual of the dual is the primal.

duality theorems [von Neumann, Tucker]

- **Weak duality theorem**
  If $\vec{x}$ is primal feasible and $\vec{p}$ is dual feasible then
  $$\vec{p}^T\vec{b} \leq \vec{c}^T\vec{x}$$
- Corollary: If $\vec{x}$ is primal feasible, $\vec{p}$ is dual feasible, and $\vec{p}^T\vec{b} = \vec{c}^T\vec{x}$, then $\vec{x}$ is optimal in the primal and $\vec{p}$ is optimal in the dual.
- **Strong duality theorem**
  Given an LP, if it has an optimal solution – then so does its dual – having equal objective functions' values.

$\Rightarrow$ **The dual provides a bound that in the best case equals the optimal solution to the primal – and thus can help solve difficult primal problems.**

dual simplex

- Simplex is a primal algorithm: maintaining primal feasibility while working on dual feasibility
- Dual-simplex: maintaining dual feasibility while working on primal feasibility –
  Implicitly use the dual to obtain an optimal solution to the primal as early as possible, regardless of feasibility; then hop from one vertex to another, while gradually decreasing the infeasibility while maintaining optimality
- **Dual-simplex is the first practical choice for most LPs**.

simplex: convergence

- Dantzig's simplex finds an optimal solution to any LP in a finite number of steps (avoiding cycles is easy, but not mentioned).
- Over half-century of improvements, its robust forms are very effective in treating very large LPs.
- However, simplex is not a polynomial-time algorithm, even if it is fast in practice over the majority of cases.
- *Pathological* LP-cases exist – where an **exponential number of steps** is needed for this algorithm to converge.
- An **ellipsoid algorithm**, guaranteed to solve every LP in a polynomial number of steps, was devised in the late 1970's by Soviet mathematicians.

"high-level" ellipsoid [Shor-Nemirovsky-Yudin]

input : a bounded convex set $\mathcal{P} \in \mathbb{R}^n$

$t \leftarrow 0$

$\mathcal{E}_t \leftarrow$ ellipsoid containing $\mathcal{P}$

while *center* $\vec{\xi}_t$ *of* $\mathcal{E}_t$ *is not in* $\mathcal{P}$ **do**

  let $\vec{c}^T\vec{x} \leq \vec{c}^T\vec{\xi}_t$ be such that $\left\{\vec{x} : \ \vec{c}^T\vec{x} \leq \vec{c}^T\vec{\xi}_t\right\} \supseteq \mathcal{P}$

  update to the ellipsoid with minimal volume containing the intersected subspace:

$$\mathcal{E}_{t+1} \leftarrow \mathcal{E}_t \cap \left\{\vec{x} : \ \vec{c}^T\vec{x} \leq \vec{c}^T\vec{\xi}_t\right\}$$

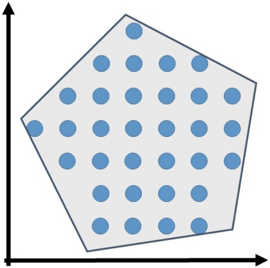
  $t \leftarrow t + 1$

end

output: *center* $\vec{\xi}_t \in \mathcal{P}$

ellipsoid aftermath

- Polynomial-time algorithm for obtaining $\vec{x}^*$ within any given bounded convex set
- Khachian first used it (1979) to show polynomial solvability of LPs
- **Theorem**: if there exists a polynomial-time algorithm for solving a strict linear inequalities problem, then there exists a polynomial-time algorithm for solving LPs (see [3] for the proof).
- Conceptual novelty: disregarding the combinatorial nature of LPs
- Unlike simplex, ellipsoid is slow and steady in practice.
- Yet, its theoretical "polynomiality" has strong implications also for discrete optimization.



discrete optimization

from LP to ILP

- The introduction of integer decision variables into a linear optimization problem yields a so-called (mixed)-integer linear program ((M)ILP) [4, 5].
- A powerful modeling framework with much flexibility in describing discrete optimization problems
- The general ILP is itself *NP-complete* — and yet, there are subsets of "very easy" versus "very hard" problems
- *p2p shortest path* over a graph with n nodes has an $\mathcal{O}(n^2)$ algorithm, versus the *traveling salesman problem*...
- Unlike "pure-LP", whose complexity is dictated by $n + m$ (variables+constraints), the choice of formulation in ILP is critical!

integer linear optimization

- Pure integer:

$$\boxed{\begin{aligned} \text{maximize}_{\vec{x}} \quad & \vec{c}^T \vec{x} \\ \text{subject to: } & \mathbf{A}\vec{x} \leq \vec{b} \\ & \vec{x} \in \mathbb{Z}_+^n \end{aligned}} \tag{3}$$

- Binary optimization (**important special case**):

$$(3) \text{ with } \vec{x} \in \{0, 1\}^n$$

- Mixed-integer:

$$\boxed{\begin{aligned} \text{maximize}_{\vec{x}} \quad & \vec{c}^T \vec{x} + \vec{h}^T \vec{y} \\ \text{subject to: } & \mathbf{A}\vec{x} + \mathbf{B}\vec{y} \leq \vec{b} \\ & \vec{x} \in \mathbb{Z}_+^n, \ \vec{y} \in \mathbb{R}_+^m \end{aligned}} \tag{4}$$

LP relaxations and the convex hull

- Given a discrete optimization problem, its consideration as a "*pure*" (continuous) LP is called its **LP relaxation**; e.g., each binary variable becomes continuous within the interval $[0, 1]$:

$$x_i \in \{0, 1\} \rightsquigarrow 0 \leq x_i \leq 1$$

- Formally, given a valid ILP formulation $\left\{\vec{x} \in \mathbb{Z}_+^n \mid \mathbf{A}\vec{x} \leq \vec{b}\right\}$, the polytope $\left\{\vec{x} \in \mathbb{R}^n \mid \mathbf{A}\vec{x} \leq \vec{b}\right\}$ constitutes its LP relaxation.
- The **convex hull** of a set of points is defined as the "smallest polytope" that contains all of the points in the set; given a finite set $S := \left\{p^{(1)}, \ldots, p^{(N)}\right\}$, it is defined as

$$\mathcal{C}(S) := \left\{ q \,\middle|\, q = \sum_k^N \lambda_k p^{(k)} \;,\; \sum_k^N \lambda_k = 1,\; \lambda_k \geq 0,\; p^{(k)} \in S \right\} \quad (5)$$

- The **integral hull** is the *convex hull of the set of integer solutions*:

$$\widetilde{\mathcal{P}} := \mathcal{C}(X), \quad X \subset \mathbb{Z}^n \text{ solution points}$$

quality of formulations

- The quality of an ILP formulation for a problem having a feasible solution set X, is governed by the **closeness** of the *feasible set of its LP relaxation* to $\mathcal{C}(X)$.
- Given an ILP with two valid formulations, $\{P_1,\ P_2\}$, let $\left\{P_1^{LR},\ P_2^{LR}\right\}$ denote the feasible sets of their LP relaxations: we state that P_1 **is as strong as** P_2 if $P_1^{LR} \subseteq P_2^{LR}$, or that P_1 **is better than** P_2 if $P_1^{LR} \subset P_2^{LR}$ (strictly).
- Explicit knowledge of $\mathcal{C}(X)$ is thus very valuable!
- If the *integral hull* is attainable as $\widetilde{\mathcal{P}} = \left\{\vec{x} \in \mathbb{R}^n \mid \widetilde{\mathbf{A}}\vec{x} \leq \widetilde{\vec{b}}\right\}$, the problem is polynomially solvable (all vertices are integers!) [4]
- "**Easy Polyhedra**": MILP with fully-understood integral hulls — *assignment, min-cost flow, matching, spanning tree, etc.*

branch-and-bound

One of the common approaches to address integer programming, relying on the ability to bound a given problem.

It is a tree-search, adhering to the principle of *divide-and-conquer*:

(i) **branch**: select an active subproblem $\hat{\mathcal{F}}$
(ii) **prune**: if $\hat{\mathcal{F}}$ is infeasible – discard it
(iii) **bound**: otherwise, compute its lower bound $L(\hat{\mathcal{F}})$
(iv) **prune**: if $L(\hat{\mathcal{F}}) \geq U$, the current best upper bound, discard $\hat{\mathcal{F}}$
(v) **partition**: if $L(\hat{\mathcal{F}}) < U$, either completely solve $\hat{\mathcal{F}}$, or further break it to subproblems added to the list of active problems

"high-level" LP-based branch-and-bound

input : a linear integer program $\mathcal{F}$

$\Omega \leftarrow \{\mathcal{F}\}$; $U \leftarrow \infty$ `/* active problems' set; global upper bound */`

while Ω *is not empty* **do**

  let $\hat{\mathcal{F}}$ be a active subproblem, $\hat{\mathcal{F}} \in \Omega$; $\Omega \leftarrow \Omega \backslash \left\{\hat{\mathcal{F}}\right\}$

  compute its lower bound $L(\hat{\mathcal{F}})$ **by solving its LP relaxation**

  if $L(\hat{\mathcal{F}}) < U$ **then**

    $U \leftarrow L(\hat{\mathcal{F}})$

    if *exists heuristic solution* $\vec{\psi}$ *for* $\hat{\mathcal{F}}$ **then** $\vec{x}^* \leftarrow \vec{\psi}$

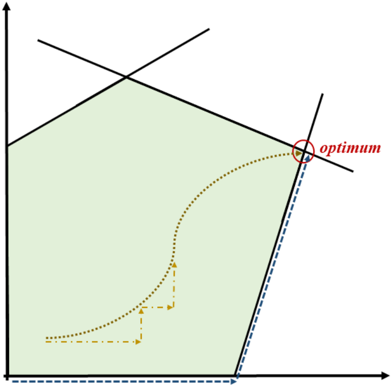
    else given the LP relaxation's optimizer, $\vec{\xi}$, if it contains a *fractional* decision variable ξ_i, construct 2 subproblems $\left\{\dot{\mathcal{F}}, \ddot{\mathcal{F}}\right\}$ by imposing either one of the new constraints $x_i \leq \lfloor \xi_i \rfloor$ or $x_i \geq \lceil \xi_i \rceil$ — and add them $\Omega \leftarrow \Omega \cup \left\{\dot{\mathcal{F}}, \ddot{\mathcal{F}}\right\}$

    `/* selection rules needed if #fractional` $\xi_i > 2$`*/`

  end

end

output: $\vec{x}^*$



MP in practice

obtaining an LP standard form

- LP's **standard form** (minimization, equality constraints, non-negative variables):

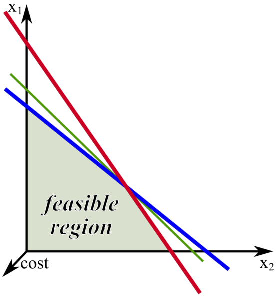
$$\boxed{\begin{aligned} &\text{minimize}_{\vec{x}}\ \vec{c}^T\vec{x} \\ &\text{subject to: } \mathbf{A}\vec{x} = \vec{b} \\ &\qquad\qquad\qquad \vec{x} \geq 0 \end{aligned}}$$

- Applicable transformations to obtain standard form (introducing *slack/surplus* variables and accounting for *unrestricted* variables):

$$\begin{array}{lll} \text{(a)} \quad \max\ \vec{c}^T\vec{x} & \Leftrightarrow & -\min\ \left(-\vec{c}^T\vec{x}\right) \\ \text{(b)} \quad \vec{a}_i^T\vec{x} \leq b_i & \Leftrightarrow & \vec{a}_i^T\vec{x} + s_i = b_i,\ s_i \geq 0 \\ \text{(c)} \quad \vec{a}_i^T\vec{x} \geq b_i & \Leftrightarrow & \vec{a}_i^T\vec{x} - s_i = b_i,\ s_i \geq 0 \\ \text{(d)} \quad -\infty < x_j < \infty & \Leftrightarrow & x_j := x_j^+ - x_j^-,\ \ x_j^+ \geq 0,\ x_j^- \geq 0 \end{array}$$

linear programming: solutions

$$\text{minimize} \ -x_1 - x_2$$
$$\text{subject to: } x_1 + 2x_2 \leq 3$$
$$2x_1 + x_2 \leq 3$$
$$x_1, x_2 \geq 0$$




```
dvar float+ x1,x2,s1,s2;
minimize
   -x1 - x2;
subject to {
    x1 + 2x2 + s1 == 3;
    2x1 + x2 + s2 == 3;
}
```

basic knapsack in OPL

```
// Data reading from external database (or sheet or flat file)
{int} N = ...;
{int} TOTAL = ...;

dvar int select_ind[N] in 0..1;
dvar float+ dev_plus;
dvar float+ dev_minus;

minimize
   dev_plus + dev_minus;

subject to {
      sum (n in N) (n * select_ind[n]) + dev_plus - dev_minus ==
          TOTAL ;

}
```

solver operations

- Modern solvers allow the user to choose/tune their core algorithms:

```
cplex.startalg = 1; //primal simplex; for LP relaxation
cplex.lpmethod = 2; //dual simplex
cplex.epgap = 0.001; //relative MIP optimality gap
cplex.IntSolLim = 100; //number of integer solutions to stop
cplex.polishtime = 1800; //polishing time; see text below
cplex.tilim = 1800; //computation time limit
```

- Some MILP solvers actually employ *evolutionary operators* in their heuristic components, such as CPLEX's `polish` subroutine [6].

quadratic programming (QP)

- The simplest formulation of a QP has a *quadratic* objective function and *linear* constraints:

$$\boxed{\begin{array}{ll} \text{minimize}_{\vec{x}} & \frac{1}{2}\vec{x}^T\mathbf{Q}\vec{x} + \vec{c}^T\vec{x} \\ \text{subject to:} & \mathbf{A}\vec{x} \leq \vec{b} \\ & \vec{\ell} \leq \vec{x} \leq \vec{u} \end{array}} \tag{6}$$

- Renowned QP: the Markowitz portfolio – minimizing risk while ensuring minimal ROI, subject to a bounded portfolio investment:

$$\begin{array}{l} \mathbf{Q}: \text{ portfolio's covariance matrix, representing RISK} \\ \vec{c} = \vec{0} \\ \vec{\rho}: \text{ stochastic return, representing ROI} \\ \text{constraints:} \quad \vec{\rho}^T\vec{x} \geq \text{ROI}_{min} \\ \qquad\qquad\qquad \sum_i x_i = \text{INVEST}_{total} \end{array} \tag{7}$$

QP (QCP) and MIQP (MIQCP)

- A Quadratically-Constrained Program (QCP) has quadratic terms in its constraints (possibly no quadratic terms in the objective)
- Mixed-integer QP and QCP involve also integer decision variables
- Renowned MIQP: the quadratic assignment problem (QAP)
- A basic QCP formulation:

```
dvar float x[0..2] in 0..40;
minimize
   0.5* (33*x[0]*x[0] + 22*x[1]*x[1] + 11*x[2]*x[2] -
       12*x[0]*x[1] - 23*x[1]*x[2]) - x[0] + 2*x[1] +
       3*x[2];
subject to {
    -x[0] + x[1] + x[2] <= 20;
     x[0] - 3*x[1] + x[2]<= 30;
     x[0]*x[0] + x[1]*x[1] + x[2]*x[2] <= 1.44;
}
```

the traveling salesman problem

- The *archetypical* Traveling Salesman Problem (TSP) is posed as finding a Hamilton circuit of minimal total cost. Explicitly, given a directed graph G, with a vertex set $V = \{1, \ldots, |V|\}$ and an edge set $E = \{\langle i, j \rangle\}$, each edge has cost information $c_{ij} \in \mathbb{R}^+$.

- **Black-box formulation: permutations**

$$\boxed{\begin{array}{l} \textbf{[TSP-perm]} \;\; \text{minimize} \displaystyle\sum_{i=0}^{n-1} c_{\pi(i),\pi((i+1)_{\mathrm{mod} n})} \\ \text{subject to:} \\ \qquad \pi \in P_{\pi}^{(n)} \end{array}} \tag{8}$$

- But this is clearly not an MP, since it does not adhere to the canonical form!

ILP formulation [Miller-Tucker-Zemlin]

TSP as an ILP utilizes n^2 binary decision variables $\mathtt{x}_{ij}$:

$$
\boxed{
\begin{aligned}
&\textbf{[TSP-ILP]} \quad \text{minimize} \sum_{\langle i,j\rangle \in E} c_{ij} \cdot \mathtt{x}_{ij} \\
&\text{subject to:} \\
&\qquad \sum_{j \in V} \mathtt{x}_{ij} = 1 \quad \forall i \in V \\
&\qquad \sum_{i \in V} \mathtt{x}_{ij} = 1 \quad \forall j \in V \\
&\qquad \mathtt{x}_{ij} \in \{0,1\} \quad \forall i,j \in V
\end{aligned}
}
\tag{9}
$$

But is this enough? What about inner-circles?

ILP formulation [Miller-Tucker-Zemlin]

TSP as an ILP utilizes n^2 binary decision variables $\mathtt{x}_{ij}$:

$$\boxed{\begin{aligned}&\textbf{[TSP-ILP]}\quad \text{minimize} \sum_{\langle i,j\rangle \in E} c_{ij}\cdot \mathtt{x}_{ij}\\ &\text{subject to:}\\ &\qquad \sum_{j\in V} \mathtt{x}_{ij} = 1 \quad \forall i \in V\\ &\qquad \sum_{i\in V} \mathtt{x}_{ij} = 1 \quad \forall j \in V\\ &\qquad \mathtt{x}_{ij} \in \{0,1\} \quad \forall i,j \in V\end{aligned}} \tag{9}$$

But is this enough? What about inner-circles?

n integers $\mathtt{u}_i$ are needed as decision variables to prevent inner-circles:

$$\boxed{\begin{aligned}&\ldots\\ &\qquad \mathtt{u}_i - \mathtt{u}_j + 1 \leq (|V|-1)(1-\mathtt{x}_{ij}) \quad \forall i,j \in 1\ldots|V|\\ &\qquad |V| \geq \mathtt{u}_i \geq 2 \quad \forall i \in \{2,3,\ldots,|V|\}\end{aligned}} \tag{10}$$

the EC perspective

- Unlike GAs, which require effective mutation and crossover operators for permutations, the challenge here is mostly about obtaining an effective formulation
- Perhaps *counter-intuitively*, increasing the order of magnitude of constraints does not necessarily render the problem harder to be solved as MP.
- The given MTZ formulation for TSP is itself of a polynomial size; an alternative formulation possesses $\mathcal{O}\left(2^{|V|}\right)$ *subtour elimination constraints*, though **impractical for large graphs**.
- In any case, TSP's *integral hull* is unknown; NP-hard problem.

- Note that EC researchers also started to look at TSP and other problems in a gray-box perspective: **Darrell Whitley's tutorial on "Next-Generation Genetic Algorithms"** !

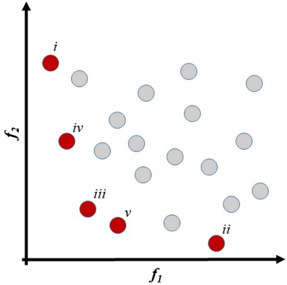
TSP on undirected graphs: OPL implementation

Addressing the undirected TSP by means of "node labeling" – assuming a single visit per node:

```
// Data preparation
tuple Raw_Edge {int point1; int point2; int dist; int active;}
{Raw_Edge} raw_edges = ...;

//Every edge is taken in both directions due to the graph
    nature, using 'union':
tuple Edge {int point1; int point2; int dist;}
{Edge} edges = {<e.point1, e.point2, e.dist> | e in raw_edges :
    e.active == 1}
      union {<e.point2, e.point1, e.dist> | e in raw_edges :
           e.active == 1};
{int} points = {e.point1 | e in edges};
int n = card (points); //set cardinality, i.e., number of cities
```

TSP in OPL continued: core model

```
dvar int edge_selector[edges] in 0..1;
dvar int label[points] in 0..n-1;

minimize sum (e in edges) edge_selector[e]*e.dist;

subject to {
  forall (p in points)
  ct_in_deg_equal_one:
    sum (e in edges : e.point2 == p) edge_selector[e] == 1;
  forall (p in points)
  ct_out_deg_equal_one:
    sum (e in edges : e.point1 == p)edge_selector[e] == 1;
  forall (e in edges : e.point2 != 1)
  ct_monotone_labeling:
    edge_selector [e] == 1 => label [e.point1] ==
        label[e.point2]-1;
}
```



extended topics

1. robust optimization

- In Stochastic Optimization, some numerical data is uncertain and associated with (partially-)known probability distributions; e.g.,

$$\min_{\vec{x},t}\left\{t:\ \mathrm{Prob}_{(\vec{c},\mathbf{A},\vec{b})\sim\Pi}\left\{\vec{c}^T\vec{x}\le t\wedge\mathbf{A}\vec{x}\le\vec{b}\right\}\ge 1-\epsilon\right\}$$

  with Π denoting the data distribution and $\epsilon \ll 1$ being the tolerance.

- In Robust Optimization [7], an uncertain LP is defined as a **collection**

$$\left\{\min_{\vec{x}}\left\{\vec{c}^T\vec{x}:\ \mathbf{A}\vec{x}\le\vec{b}\right\}\ :\ \left(\vec{c},\mathbf{A},\vec{b}\right)\in\mathcal{U}\right\}$$

  of LPs sharing a common structure and having the data varying in a given *uncertainty set* $\mathcal{U}$.

- A rich variety of MP techniques exist for robust/stochastic optimization; e.g., the Robust Stochastic Approximation Approach [8].

A. Ben-Tal, L. El Ghaoui, and A. Nemirovski: *Robust Optimization.* Princeton University Press, 2009.

2. multiobjective exact optimization

Diversity Maximization Approach (DMA) [9] key features:

- Iterative-exact nature: obtains a new **exact non-dominated solution** per each iteration
- Criteria exist for the attainment of the complete Pareto frontier
- Fine distribution of the existing set already found is guaranteed
- Optimality gap is provided – what may be gained by continuing constructing the Pareto frontier
- Solves any type of frontier (even if seems as a weighted sum)
- Importantly, DMA is **MILP if the original problem is MILP**

M. Masin and Y. Bukchin, 2008, "Diversity Maximization Approach for Multi-Objective Optimization", *Operations Research*, 56, 411-424.

"high-level" DMA for M-objectives linear problems

input : a linear program featuring M objectives

1. Find an optimal solution for a weighted sum of multiple objectives with any reasonable strictly positive weights. If there is no feasible solution – **Stop**.
2. Set the partial efficient frontier equal to the found optimal solution. Choose optimality gap tolerance and maximal number of iterations.
3. If the maximal number of iterations is reached – **Stop**, otherwise **add M binary variables and $(M+1)$ linear constraints to the previous MILP model**.
4. Maximize the proposed diversity measure. If the diversity measure is less than the optimality gap tolerance – **Stop**, otherwise add the optimal solution to the partial efficient frontier and go to Step 3.

output: Pareto set, Pareto frontier

3. hybrid metaheuristics

- Bridging between the “formal/OR” to “heuristic/SoftComp” and aiming to share expertise gained from each end.
- Hybrids are a trendy route which has proven powerful and has recently accomplished a great deal.
- MP-solvers occasionally “hit-a-wall” on discrete optimization problems – and that is when hybrids prove useful.
- A powerful hybrid theme that follows two principles: neighborhood search and solution construction

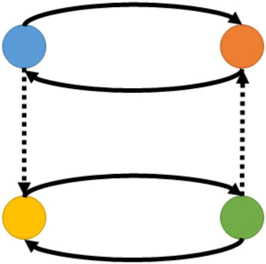
Ch. Blum and G. R. Raidl: *Hybrid Metaheuristics - Powerful Tools for Optimization*. Springer, 2016, ISBN: 978-3-319-30882-1.

a hybrid outperforming an MP-solver

MP formulation of the Multidimensional Knapsack Problem (MKP), utilizing n binary decision variables $\mathtt{x}_i$ for items' selection (relying on instance-specific data for the m knapsacks' capacities c_k, the profits of the n items, p_i, as well as the resources' consumptions $r_{i,k}$ of items per knapsacks):

$$\boxed{\begin{array}{l} \textbf{[MKP]} \;\; \text{maximize} \displaystyle\sum_{i=1}^{n} p_i \cdot \mathtt{x}_i \\ \text{subject to:} \\ \qquad \displaystyle\sum_{i=1}^{n} r_{i,k}\mathtt{x}_i \geq c_k \;\; \forall k \in 1 \ldots m \\ \qquad \mathtt{x}_i \in \{0,1\} \;\; \forall i \in 1 \ldots n \end{array}} \tag{11}$$

IBM's CPLEX was demonstrated to be outperformed when deployed alone on the complete problem, within a practical CPU time-limit – in comparison to a proposed hybrid [10].



discussion

quick summary

- MP is a well-established domain encompassing a variety of algorithms with underlying rigorous theory.
- Broad knowledge of MP is valuable for both EC theoreticians and practitioners
- Given convex problems, MP is most likely the fittest tool
- Given discrete optimization problems that may be formulated as MILP/MIQP – it makes sense to first try MP-solvers
- MP is inherently adjusted to constrained problems (unlike EC...)
- Effective MP formulation lies in the heart of practical problem-solving
- Robustness to uncertainty, Pareto optimization, and hybridization are solid extensions to classical MP

communities and resources

- INFORMS: The Institute for Operations Research and the Management Sciences; `https://www.informs.org/`
- COIN-OR: Computational Infrastructure for Operations Research – a project that aims to "create for mathematical software what the open literature is for mathematical theory"; `https://www.coin-or.org/`
- MATHEURISTICS: model-based metaheuristics, exploiting MP in a metaheuristic framework; `http://mh2018.sciencesconf.org/`

partial list of languages and solvers

- Modeling languages:
  GAMS
  AMPL
  OPL
  (`python` (Gurobi-Python, SciPy), MATLAB, ...)
- Environments and modeling systems:
  Google Optimization Tools (!)
  IBM ILOG CPLEX
  Gurobi
  sas
  YALMIP
- Third-party solvers (free and open-source):
  CBC (via Coin-OR)
  GLPK (GNU Linear Programming Kit)
  SoPlex
  LP_SOLVE

benchmarking and competitions

- MIPLIB: the Mixed Integer Programming LIBrary

  `http://miplib.zib.de/`

- CSPLib: a problem library for constraints

  `http://csplib.org/`

- SAT-LIB: the Satisfiability Library - Benchmark Problems

  `http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html`

- TSP-LIB: the Traveling Salesman Problem sample instances

  `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`

Acknowledgements go to
Dr. Yossi Shiloach (IBM-Research retired)
Dr. Michael Masin (IBM-Research)

obrigado

references

[1] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver, *Combinatorial Optimization*. New York, NY, USA: John Wiley and Sons, 2011.

[2] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York: Cambridge University Press, 2004.

[3] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Dover Books on Computer Science, Dover Publications, 1998.

[4] A. Schrijver, *Theory of Linear and Integer Programming*. John Wiley and Sons, 1998.

[5] E. R. Bixby, M. Fenelon, Z. Gu, E. Rothberg, and R. Wunderling, "MIP: Theory and practice — closing the gap," in *System Modelling and Optimization* (M. J. D. Powell and S. Scholtes, eds.), (Boston, MA), pp. 19–49, Springer US, 2000.

[6] E. Rothberg, "An Evolutionary Algorithm for Polishing Mixed Integer Programming Solutions," *INFORMS Journal on Computing*, vol. 19, no. 4, pp. 534–541, 2007.

[7] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton Series in Applied Mathematics, Princeton University Press, October 2009.

[8] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust Stochastic Approximation Approach to Stochastic Programming," *SIAM J. Optim.*, vol. 19, no. 4, pp. 1574–1609, 2009.

[9] M. Masin and Y. Bukchin, "Diversity maximization approach for multiobjective optimization," *Operations Research*, vol. 56, no. 2, pp. 411–424, 2008.

[10] C. Blum and G. R. Raidl, *Hybrid Metaheuristics: Powerful Tools for Optimization*. Artificial Intelligence: Foundations, Theory, and Algorithms, Springer International Publishing, 2016.