MVE165/MMG631
Linear and integer optimization with applications
Lecture 10
Combinatorial optimization theory and algorithms

Ann-Brith Strömberg

2019–05–03

Assignment information

- Don't forget that Assignment 2 shall be handed in twice:
  1. in "Assignment 2 - project report"
     Friday, May 10, before 9:30
  2. individually in "Ass2 - opposition"
     Friday, May 10, between 12:00 and 17:00
  3. Also: the individual "Assignment 2 - cooperation report"
     Monday, May 13, before 23:55
  4. Then you will receive a report for peer review (via "Ass2 - opposition"), which should be submitted by
     Tuesday, May 14, before 23:55
- A "doodle" from which you should choose *either* Assignment 3a *or* Assignment 3b, *as well as* a time slot for its presentation, will be published on the course homepage in the middle of next week (i.e., week 19). The specific time for the publication will be pre-announced in an email/PIM.

Overview

Convexity

- Local and global optima

Heuristics

I Constructive heuristics

II Local search methods

III Approximation algorithms

IV Meta-heuristics

Recall: Convex sets

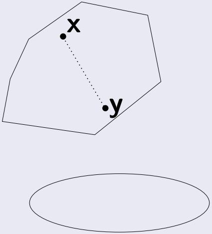
Convex set – definition

A set S is convex if, for any elements $\mathbf{x}, \mathbf{y} \in S$ it holds that

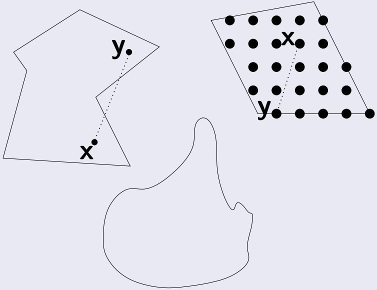
$$\alpha \mathbf{x} + (1 - \alpha)\mathbf{y} \in S \quad \text{for all} \quad 0 \leq \alpha \leq 1$$

Examples:

Convex sets



Non-convex sets



Local vs. global optima

Consider a minimization problem

$$\min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

- Global optimum:
  A solution $\mathbf{x}^* \in X$ such that $\mathbf{c}^\top \mathbf{x}^* \leq \mathbf{c}^\top \mathbf{x}$ for all $\mathbf{x} \in X$
- ε-neighbourhood of $\bar{\mathbf{x}}$: $N_\varepsilon(\bar{\mathbf{x}}) = \left\{ \mathbf{x} \in X \,\middle|\, \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \varepsilon \right\}$
  - The distance measure $\|\mathbf{x} - \bar{\mathbf{x}}\|$ may be "freely" defined
  - E.g., # arcs differing (Hamming distance), Euclidean, Manhattan, 2-interchange, 3-interchange, ...
  - $\varepsilon \geq 0$
- Local optimum:
  A solution $\bar{\mathbf{x}} \in X$ such that $\mathbf{c}^\top \bar{\mathbf{x}} \leq \mathbf{c}^\top \mathbf{x}$ for all $\mathbf{x} \in N_\varepsilon(\bar{\mathbf{x}})$ for some $\varepsilon > 0$
- Global optimum of a convex optimization problem:
  For a convex optimization problem, any local optimum is also a global optimum

Heuristic algorithms

- Optimization problems with high complexity may be too time consuming to solve to optimality
- Heuristic algorithms can be utilized
- But: Only local optimality can then be guaranteed

Consider a minimization problem

$$\min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

- Start by an "empty set" and "add" elements according to some (simple) rule
- Sometimes no guarantee that even a feasible solution will be found
- No measure of how "close" to a global optimum a solution is
- Special rules for structured problems
- E.g. the greedy algorithm is a constructive heuristic (finds, however, optimal solution to *minimum spanning tree*)
- For TSP: nearest neighbour, cheapest insertion, farthest insertion, etc
- EXAMPLE!

Heuristics II: Local search (Ch. 16.4)

Consider a minimization problem

$$\min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

- Start at a feasible solution, which is iteratively improved by limited modifications
- Finds a local optimum
- No measure on how close to a global optimum a solution is
- Specialized for structured problems, but also general (see Ch. 16.2)
- For TSP: e.g. 2-interchange, 3-interchange,
- EXAMPLE!

Local search heuristic algorithm (Ch. 16.4)

Consider a minimization problem

$$\min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

A general local search algorithm

0. Initialization: Choose a feasible solution $\mathbf{x}^0 \in X$. Let $k = 0$.
1. Find all feasible points in an ε-neighbourhood $N_\varepsilon(\mathbf{x}^k)$ of $\mathbf{x}^k$
2. If $\mathbf{c}^\top \mathbf{x} \geq \mathbf{c}^\top \mathbf{x}^k$ for all $\mathbf{x} \in N_\varepsilon(\mathbf{x}^k) \Rightarrow$ Stop; $\mathbf{x}^k$ is a local optimum (w.r.t. N_ε)
3. Choose $\mathbf{x}^{k+1} \in N_\varepsilon(\mathbf{x}^k)$ such that $\mathbf{c}^\top \mathbf{x}^{k+1} < \mathbf{c}^\top \mathbf{x}^k$
4. Let $k := k + 1$ and go to step 1

Heuristics III: Approximation algorithms (Ch. 16.6)

Consider a minimization problem

$$z^* := \min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

Properties of approximations algorithms

- Let $\bar{z} := \mathbf{c}^\top \bar{\mathbf{x}}$ for some $\bar{\mathbf{x}} \in X$ be computed by an *approximation algorithm*
- Performance guarantee: $\dfrac{\bar{z} - z^*}{z^*} \leq \alpha$ for some $0 < \alpha \leq 1$
- Specialized algorithms for structured problems

Example of an approximation algorithm

- The spanning tree approximation algorithm for the TSP
- First: We need some more definitions for this: *Spanning trees* and *greedy algorithms*

The minimum spanning tree (MST) problem (Ch. 8.3)

- Given an undirected graph $G = (N, E)$ with nodes N, edges E and distances d_{ij} for each edge $(i, j) \in E$
- Find a subset of the edges that connects all nodes at minimum total distance
- The number of edges in a spanning tree is $|N| - 1$
- A (spanning) tree contains *no cycles*
- MST is a very simple problem (a matroid) that can be solved to optimality by *greedy algorithms*

Greedy algorithms for MST (Ch. 8.3)

Prim's algorithm

1. Start at an arbitrary node
2. Among the nodes that are not yet connected, choose the one that can be connected at minimum cost
3. Stop when all nodes are connected

Solve an example!

Kruskal's algorithm

1. Sort the edges by increasing distances
2. Choose edges starting from the beginning of the list; skip any edge that would result in a cycle
3. Stop when all nodes are connected

Solve an example!

Spanning tree approximation algorithm for the TSP (Ch. 16.6.1)

A TSP on an undirected graph $G = (N, E, \mathbf{c})$

Assume

- G complete $\Leftrightarrow$ edges between all pairs of nodes $[(i, j) \equiv (j, i)]$
- Δ-inequality: $c_{ij} \leq c_{ik} + c_{kj}$ for all $i, j, k \in N$ DRAW!

Algorithm

1. Find a minimum spanning tree $T \subset E$ on G
2. Create a *multigraph* G' using *two copies* of each edge in T
3. Find an Eulerian walk of G' and an embedded TSP-tour

Not longer than twice the optimal tour:

- Guarantee: $\dfrac{\bar{z} - z^*}{z^*} \leq 1$
- EXAMPLE!

Performance guarantee for the spanning tree approximation for TSP

Theorem

$$\frac{\bar{z} - z^*}{z^*} \leq 1$$

Bevis.

- Let $c(\text{TSP}) = z^*$ and $c(\text{tour}) = \bar{z}$
- A spanning tree is a relaxation of a TSP:
  All subtour elimination constraints are fulfilled, but not the node valence (2 edges incident to each node)

$\Rightarrow$ $c(\text{MST}) \leq c(\text{TSP})$

- Two copies of each edge $\Rightarrow c(\text{tour}) \leq 2c(\text{MST}) \leq 2c(\text{TSP})$

$$\Rightarrow \quad \frac{c(\text{tour}) - c(\text{TSP})}{c(\text{TSP})} \leq 1$$

□

Heuristics IV: Metaheuristics (Ch. 16.5)

Consider a minimization problem

$$\min_{\mathbf{x} \in X} \quad \mathbf{c}^\top \mathbf{x}$$

- Metaheuristics
  - intend to be more efficient than plain local search methods;
  - aim at guiding local search methods in a systematic and efficient way;
  - include tabu search, simulated annealing

More about heuristics

A useful combination of heuristics

1. Start using a constructive heuristic $\Rightarrow$ feasible solution
   - The choice of neighbourhood definition is model-specific (e.g. Euclidean distance, # arcs differing, ...)
2. Apply a local search algorithm
   - Finds a *locally* optimal solution
   - *No guarantee* to find global optimal solutions

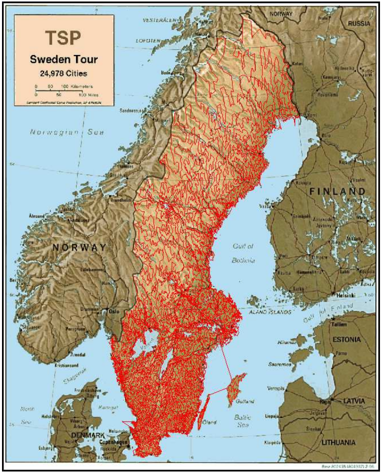
Variants and computational properties

- Extensions (e.g. tabu search): Temporarily allow worse solutions $\Rightarrow$ “move away” from a local optimum (Ch. 16.5)
- Larger neighbourhoods yield better local optima, but takes more computation time to explore

The historical development of TSP solution

Optimal solutions to TSP's of different sizes found

year	n
1954	49
1962	33
1977	120
1987	532
1987	666
1987	2392
1994	7397
1998	13509
2001	15112
2004	24978
2005/06	85900



The worlds largest TSP solved “so far” (2004) ...

- A TSP of 24 978 cities and villages (red houses) in Sweden
- Optimal tour: $\approx$ 72 500 km (855597 TSP LIB units)
- The tour of length 855 597 was found in March 2003 (Lin-Kernighan's TSP heuristic)
- It was proven in May 2004 that no shorter tour exists
- A variety of heuristics, B&B, and cut generation algorithms
- The final stages that improved the lower bound from 855 595 up to 855 597 required $\approx$ 8 CPU years (running in parallel on a network of Linux workstations)

  “Without knowledge of the 855 597 tour we would not have made the decision to carry out this final computation”
- New record in 2005/06: 85 900 locations in a VLSI application `www.math.uwaterloo.ca/tsp/pla85900`
- iPhone/iPad App: Concorde TSP `www.math.uwaterloo.ca/tsp/iphone`