Local vs. Global Optimization

- Local optimization
  - Use local information to construct solution
  - Greedy; exploitation
  - Fast but can become trapped in a local optimum
- Global optimization
  - Use global information to construct solution
  - Exploration
  - Slow but can find better solution for complex problems

ACO Algorithms

- Most global search algorithms combine global and local information to construct solutions
- Global information for TSP:
  - Pheromone
- Local information for TSP:
  - Distance to not-yet-visited neighboring cities
- Increase the “greediness” of an ACO algorithm seems to provide better solution.

Local optimization for TSP

- 2-opt heuristic:
  - Iteratively removing two edges and replacing them with two different edges to reconnect the broken fragments into a new and shorter tour

2-opt operation

Choose two edges that do not share a node at random
2-opt operation

Remove them

Reconnect the path in a different way (there is only one valid new way)

2-opt Algorithm

current_tour:=create_random_initial_tour()
repeat
-modified_tour:=apply_2opt_move(current_tour)
-if length(modified_tour) < length(current_tour)
-then
-current_tour:=modified_tour //hill climbing
until no further improvement or a specified number of iterations

• Demo: http://www-e.uni-magdeburg.de/mertens/TSP/node3.html

2-opt Algorithm

Exercise

Exercise

k-Opt algorithm

• 2-opt can be extended to 3-opt or k-opt.
• They are excellent local optimization algorithms for the TSP and similar problems.
• However, they do not always produce global optimal solution.
• Many global optimization methods, i.e. EA, PSO and ACO employ k-Opt to tune its final solution and obtain improved solutions.
The Lin-Kernighan (LK) Algorithm

- The algorithm is a generation of k-opt with variable k:
  - At each iteration, k number of edges \( X = \{x_1, \ldots, x_k\} \) are replaced by different set of k edges \( Y = \{y_1, \ldots, y_k\} \) to improve the solution.
  - The value of k can be different from one iteration to another.

LK Algorithm - Continue

- The two sets \( X \) and \( Y \) are constructed element by element adding a pair of edges, \( x_i \) and \( y_i \) to \( X \) and \( Y \), respectively.
- The sequence of \( (x_1, y_1, x_2, y_2, \ldots, x_k, y_k) \) constitutes a closed chain of adjoining edges.

LK Algorithm - Continue

- At each iteration, a pair of edges \( x_i \) and \( y_i \) are added to \( X \) and \( Y \) only when they will lead to positive cumulative gain:
  - \( g_i = c(x_i) - c(y_i) \), gain from exchanging \( x_i \) with \( y_i \).
  - \( G_i = \text{Sum}(g_1, g_2, g_3 \ldots g_k) > 0 \)
  - The new tour is a permutation (a closed tour)
- \( X \) and \( Y \) are disjoint.

Working Example

- \( F(adebea) = 38.551 \)
- \( F(acdsea) = 34.0206 \)
- Cycle(c,f,d,a,c)