Permutation-Review

- A permutation of a finite set is an arrangement of its elements into a row (Knuth, 1973).
  - \{a, b, c, d, e\}, \{b, c, d, e, a\}, \{c, d, e, a, b\} …
- Permutation is a suitable coding for combinational optimization.
  - Scheduling, network optimization problems

Test Bank

- https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/
  - Symmetric travelling salesman problem (TSP)
  - Hamiltonian cycle problem (HCP)
  - Asymmetric travelling salesman problem (ATSP)
  - Sequential ordering problem (SOP)
  - Capacitated vehicle routing problem (CVRP)

Traveling Salesman Problem

- The salesman must visit every city in his territory exactly once and then return home covering the shortest distance.
- Variables: \(x_1, x_2, \ldots, x_n\) are the \(n\) city names
- Representation: permutation of \(n\) cities.
- Fitness function: minimize \(\sum \sum_{i < j} \text{dis}(x_i, x_j)\)
- 10 cities example:
  - 1 3 4 10 9 8 2 5 6 7
Characteristics of Permutation

• Given $n$ unique objects, $n!$ permutations of the objects exist. Searching the shortest path is an NP-hard problem.
• There are multiple equivalent solutions for some problems.
  – TSP: If starting point is not important, and the distance from city $i$ to $j$ is the same as that from city $j$ to $i$, each path $a$-$b$-$c$-$d$-$e$ can be represented in $2n$ ways and give the same distance.
• Search space: $n!/(2n) = (n-1)!/2$

Permutation Crossover Operators I

• Ordered crossover (Davis, 1985):

Permutation Crossover Operators II

• Partially matched crossover (Goldberg & Lingle, 1985):
  C swaps with A, D swaps with B, E swaps with I, F swaps with H

Permutation Mutation Operators

• Useful for problems where the order direction is important, i.e. $a$-$b$-$c$-$d$ $\neq b$-$c$-$d$-$a$
• Swap (Syswerda, 1991)
  – original: a b c d e f g h
  – mutated: a f c d e b g h
• Insert (Syswerda, 1991)
  – original: a b c d e f g h
  – mutated: a f b c d e g h
• Scramble (Davis, 1991)
  – original: a b c d e f g h
  – mutated: a f d c e b g h
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

Local optimization for TSP

• 2-opt heuristic:
  — iteratively removing two edges that do not share a common node and replacing them with two different edges to reconnect the broken fragments into a new and shorter tour

2-opt operation

Choose two edges at random

2-opt operation - continued

Choose two edges at random
2-opt Algorithm

\[
\text{current\_tour}:=\text{create\_random\_initial\_tour}() \\
\text{repeat} \\
\quad \text{modified\_tour}:=\text{apply\_2opt\_move}() \\
\quad \text{if length(modified\_tour) < length(current\_tour)} \\
\quad \text{then} \\
\quad \quad \text{current\_tour}:=\text{modified\_tour} //\text{hill climbing} \\
\text{until no further improvement or a specified number of iterations} \\
\text{• Demo: http://www-e.uni-magdeburg.de/mertens/TSP/node3.html}
\]

Exercise

f(acebd)=33.773765
ac -> pair with (be or bd)
Remove ac and be
Reconnect with bc and ae
New tour:
f(adbec)=32.42214
better, accept
ad -> pair with (ec or bc)
Exercise - Continued

- Remove ad and ec
- Reconnect with ac and bc
- New tour:
  - f(acbde)=28.81558
  - better, accept
- ac -> pair with (bd or ed)
- Remove ac and bd
- Reconnect with ab and cd
- New tour:
  - f(abcd)=21.82763
  - Done!!

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 \( X = \{x_1, ..., x_k\} \) are replaced by different set of k edges \( Y = \{y_1, ..., y_k\} \) to improve the solution.
  - The value of k can be different from one iteration to another.

LK Algorithm - Continued

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

- 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.