Nature Inspired Search and Optimisation
Advanced Aspects of Nature Inspired Search and Optimisation

Lecture 4: Stochastic Local Search algorithms

Shan He

School for Computational Science
University of Birmingham

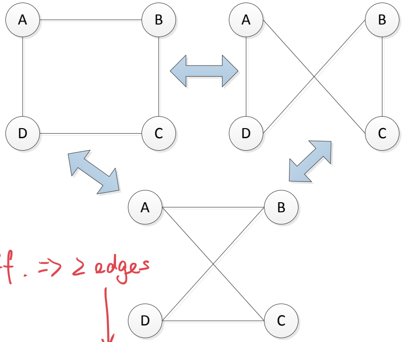
January 23, 2020

Outline of Topics

1. 2-Opt algorithm
2. Stochastic local search: basic ideas
   - Local search with random restart
   - Simulated Annealing
3. Conclusion

Let's take a look at some simple examples

- 2-3 cities: only one solutions
- 4 cities: 3 solutions
- **Question**: How those tours of the 4 cities TSP differ?
- Answer (Observation): **two immediate neighbour solutions can be two routes (cycles) only differ from two edges**
- Idea: Swapping two edges results in an **immediate** neighbour solution

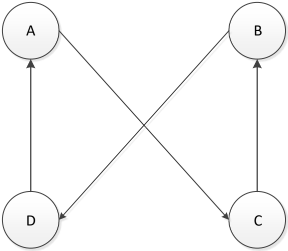


min diff. => 2 edges

immediate neighbour solutions

2-Opt algorithm

- **Question**: Given a route, e.g., A $\longrightarrow$ C $\longrightarrow$ B $\longrightarrow$ D $\longrightarrow$ A, how to swap two edges?
- Note: if you can answer this question, you invent 2-Opt algorithm, a famous local search operator first proposed by Croes in 1958 [1] for solving the travelling salesman problem.



[1] G.A. Croes. A method for solving traveling-salesman problems. Operations Research. 1958

2-Opt algorithm

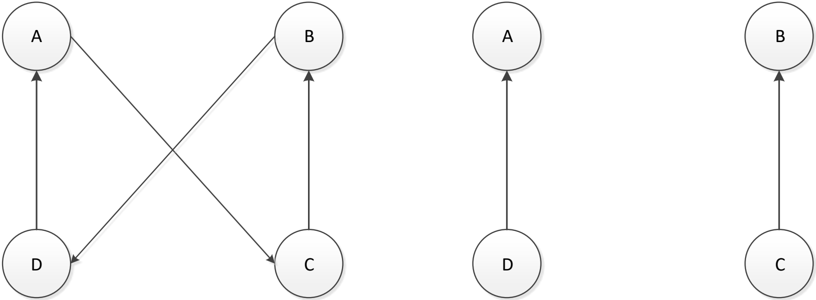
- Detailed swapping steps for swapping two edges, which result in an **immediate** neighbour solutions:
  - Step 1: removal of two edges from the current route, which results in two parts of the route.
  - Step 2: reconnect by two other edges to obtain a new solution

[1] G.A. Croes. A method for solving traveling-salesman problems. Operations Research. 1958

2-Opt algorithm

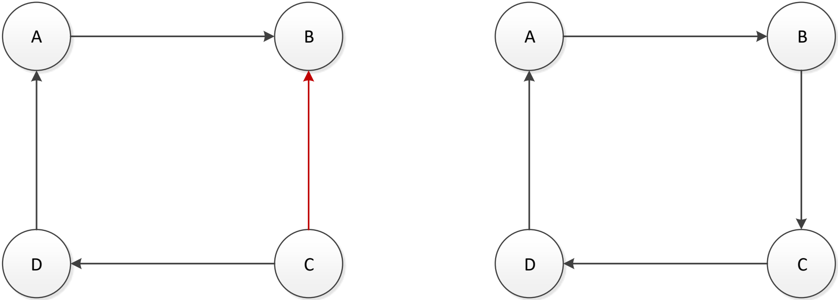
Suppose we have a route: A $\longrightarrow$ C $\longrightarrow$ B $\longrightarrow$ D $\longrightarrow$ A, let's see how to swap:

Step 1: removal of two edges from the current route, which results in two parts of the route



2-Opt algorithm

Step 2: reconnect by two other edges to obtain a new solution.

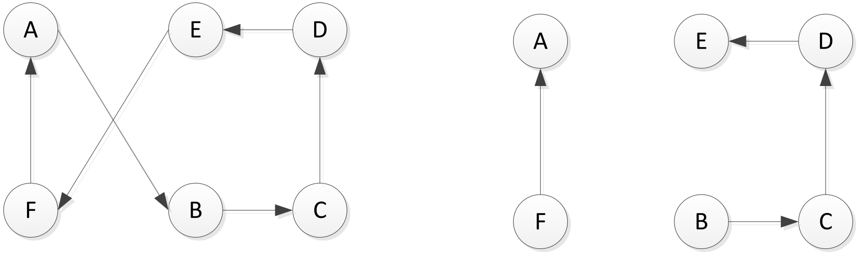


We need to reverse the order of one part of the solution, e.g., C $\longrightarrow$ B in order to get an optimal route: A $\longrightarrow$ B $\longrightarrow$ C $\longrightarrow$ D $\longrightarrow$ A

2-Opt algorithm for 6 cities TSP

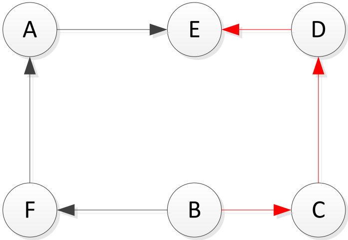
Suppose we have a route: A $\longrightarrow$ B $\longrightarrow$ C $\longrightarrow$ D $\longrightarrow$ E $\longrightarrow$ F $\longrightarrow$ A, let's see how to swap:

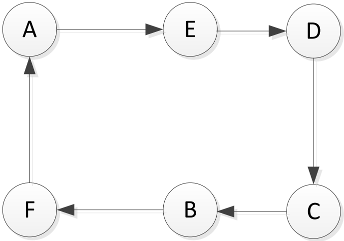
Step 1: removal of two edges from the current route, which results in two parts of the tour



2-Opt algorithm for 6 cities TSP

Step 2: reconnect by two other edges to obtain a new solution





We need to reverse the order of B $\Leftarrow$ C $\Leftarrow$ D $\Leftarrow$ E in order to get A $\longrightarrow$ E $\longrightarrow$ D $\longrightarrow$ C $\longrightarrow$ B $\longrightarrow$ F $\longrightarrow$ A

2-Opt algorithm: implementation

We observed 2-Opt essentially swap two cities, e.g., in the 6 cities TSP, 2-Opt swaps B and E, and then reverse the order of the route between them. We therefore have the following algorithm:

2-Opt algorithm

$route$:= initial TSP solution
i,j := two cities for swapping
Step 1: take $route[1]$ to $route[i-1]$ and add them in order to $newroute$
Step 2: take $route[i]$ to $route[j]$ and add them in reverse order to $newroute$
Step 3: take $route[j+1]$ to end and add them in order to new $newroute$
Output $newroute$

[a + b + c]

Code demonstration

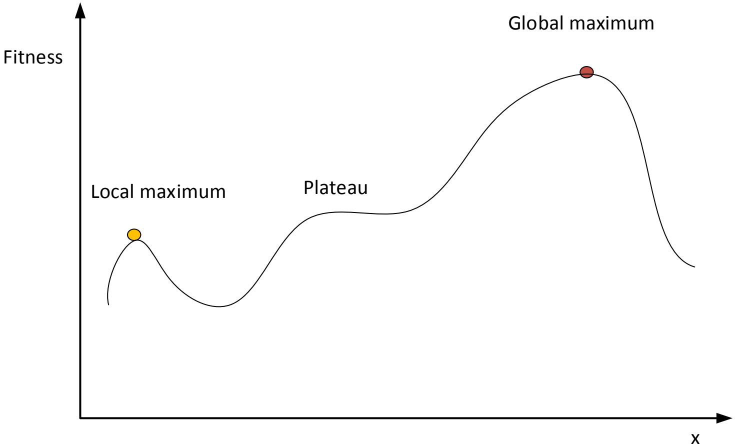
Let's run the hill-climbing with 2-opt algorithms

- **Simple hill climbing**: chooses the **first** better solution
- **Steepest ascent/descent hill climbing**: compares all neighbour solutions and chooses **the best** solution – greedy search

Why they cannot find the optimal solution of 10628?

Fitness landscape with global/local optima

Fitness landscape of a 1-dimensional optimisation (maximisation) problem



Randomised search vs Local search

- Randomised search:
  - Good at exploration, e.g., to search large unknown region of the search space
  - Not good at exploitation, e.g., to search small region around the current solution
  - Especially bad for problems where good solutions are just a small portion of all possible solutions
- Local search:
  - Good at exploitation: capable to find local optimum
  - Not good at exploration: **gets stuck into local optimum**
- Question: how to find global optimum?

Stochastic local search: Main idea

- Main idea: **escape** or **avoid** local optima
- How: Introduce randomness into local search algorithm to escape from local optima
- Escape strategies:
  - Random restart: simply restart the local search from a random initial solution
    - Applicable when:
      1. Number of local optima is small
      2. The cost for restarting the local search is cheap
  - Perform random non-improving step: randomly move to a less fit neighbour – Simulated Annealing (Explain later)

Hill climbing with random restart

Hill climbing with random restart

for $k := 0$ to $k_{\max}$

  $x_0 :=$ randomly generated initial solution

  terminationflag := false

  $x^i := x_0$

  while (terminationflag != true)

    Modify the current solution to a **immediate** neighbour one

    If $f(v) < f(x_i)$ then $x^i := v$

    If a termination criterion is met: terminationflag := true

  Store x^i

Output $x_{best} = \min(x^i, i = 1, \cdots, k_{\max})$

Simulated Annealing

- Simulated Annealing: a generic heuristic algorithm for optimisation problems proposed by Kirkpatrick in 1983 [1]
- Your first Nature inspired algorithm: inspired by annealing in metallurgy:
  - "*heat treatment that alters a material to increase its ductility and to make it more workable*"
  - The real annealing:
    - Step 1: heating a material to above its critical temperature
    - Step 2: maintaining a suitable temperature
    - Step 3: cooling
  - In annealing: find a state of lower thermodynamic free energy for metal
  - In optimisation: find a solution to get to the minimum of an objective function

[1] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi. Optimization by Simulated Annealing. Science. 1983

Generic Simulated Annealing algorithm for **minimisation**

$x := x_0$; $e := f(x)$ // Initial solution, objective function value (energy).
$x_{best} := x$; $x_{best} := x$ // Initial "best" solution
$k := 0$ // Count evaluation number.
while $(k < k_{max})$
    $T := temperature(t_0)$ // Temperature calculation.
    $x_{new} := neighbour(x)$ // Pick some neighbour.
    $e_{new} := f(x_{new})$ // Compute its objective function value.
    if $P(e, e_{new}, T) > R(0,1)$ then // Should we move to it?
        $x := x_{new}$; $e := e_{new}$ // Yes, change state.
    if $e_{new} < e_{best}$ then // Is this a new best?
        $x_{best} := x_{new}$; $e_{best} := e_{new}$ // Save as 'best found'.
    $k := k + 1$ // Increase evaluation
Output x_{best}

$P := 1$ if $e_{new} < e$, and
$P := \exp\left(\frac{e - e_{new}}{T}\right)$ otherwise
Annealing schedule $temperature()$ defines how to decreased temperature from an initial temperature t_0.

Simulated Annealing

+ - ↓ ↑

-5

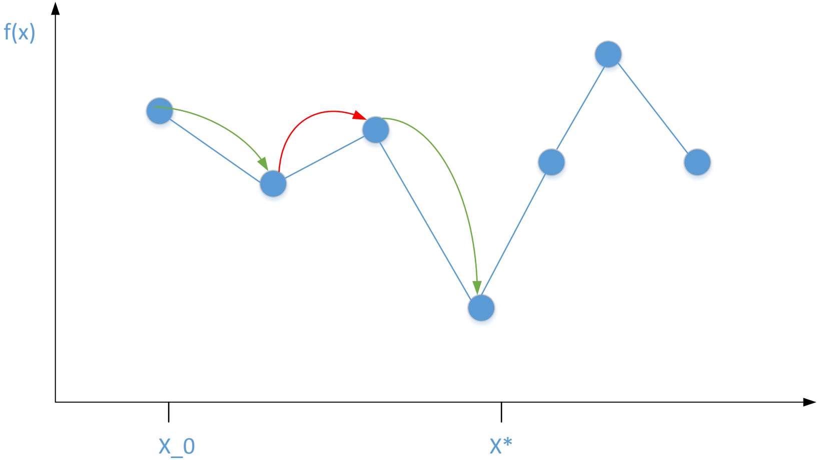
-500

- Without the analogue from metallurgy, SA algorithm is essentially a **stochastic local search** algorithm
- Main idea: Escape from local optima by a **random non-improving step**:
    - Accepting better solutions with the probability of 1: $P := 1$ if $e_{new} < e$, i.e., e_{new} is better than e -5 $\exp(\frac{-3}{100}) \sim 0. \rightarrow 1$
    - Accepting worse solutions with a certain probability: $P := \exp\left(\frac{e - e_{new}}{T}\right)$ <0 if $e_{new} \geq e$, i.e., e_{new} is worse e $e^{new} \uparrow$ $\exp\left(\frac{e - e^{new}}{T}\right) \rightarrow$
- This allows SA to explore more of the possible search space of solutions.

T starts high → T ↓ P ↓

- T, which is determined by the annealing schedule $temperature()$, will decrease, so the probability of accepting worse solutions will decrease.
- **Question**: what happen if $temperature()$ reaches 0?

Search trajectory of simulated annealing



Wait!

- Main idea: escape or **avoid** local optima
- Algorithm uses the 'avoiding' strategy: Tabu search

Conclusion

- Local search algorithms is simple but can generate good solutions for many problems
- However, local search algorithms only focus on exploitation, which will lead to local optima
- Different stochastic local search algorithms use different strategies to escape or avoid loclal optima
- Stochastic local search algorithms usually produce better results using default parameters than local search
- However, tuning stochastic local search algorithms to get better results sometimes is difficult