- 教材讨论
  - JH第3章第6节

问题1：local search的基本概念

- 你能解释这些术语的含义，并进而解释local search的基本思想吗？
  - feasible solution
  - specification
  - transformation
  - neighborhood
  - local optimum

问题1：local search的基本概念 (续)

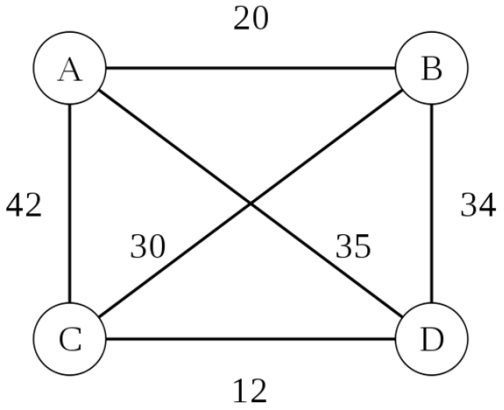
LSS($Neigh$)-Local Search Scheme according to a neighborhood $Neigh$

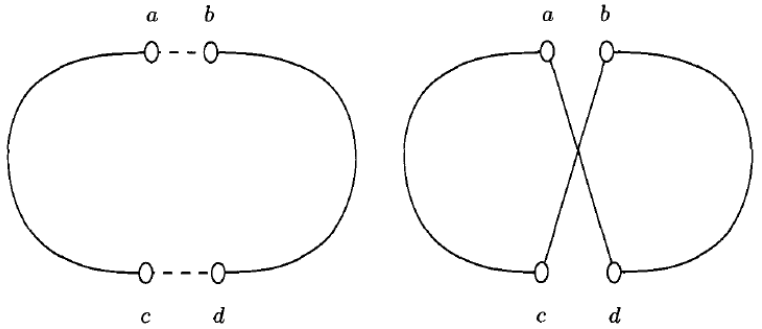
Input: An input instance x of an optimization problem U.
Step 1: Find a feasible solution $\alpha \in \mathcal{M}(x)$.
Step 2: **while** $\alpha \notin LocOPT_U(x, Neigh_x)$ **do**
begin find a $\beta \in Neigh_x(\alpha)$ such that
$cost(\beta) < cost(\alpha)$ if U is a minimization problem and
$cost(\beta) > cost(\alpha)$ if U is a maximization problem; $\alpha := \beta$
end
Output: **output**(α).

- 你能证明local search的total correctness吗？
- 决定local search能否找到并尽快找到全局最优解的因素是？
  - α
  - Neigh
  - β

问题2：hill climbing

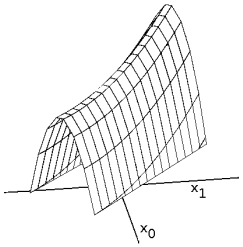
- In computer science, hill climbing is a mathematical optimization technique which belongs to the family of local search. It is an iterative algorithm that starts with an arbitrary solution to a problem, then attempts to find a better solution by incrementally changing a single element of the solution. If the change produces a better solution, an incremental change is made to the new solution, repeating until no further improvements can be found.
  - 这里的α、Neigh、β分别是怎么取的？
  - 你能以TSP为例，给出一个具体的算法吗？

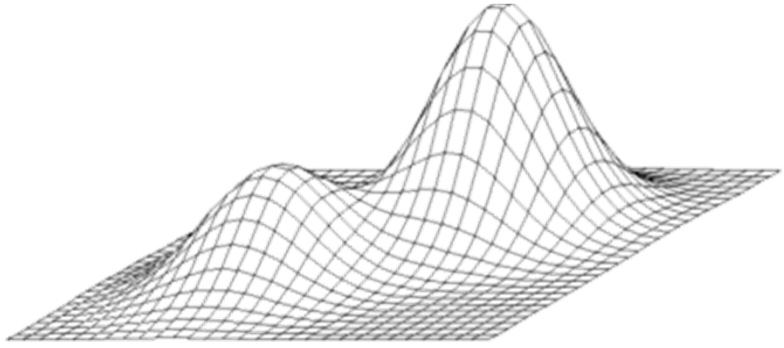




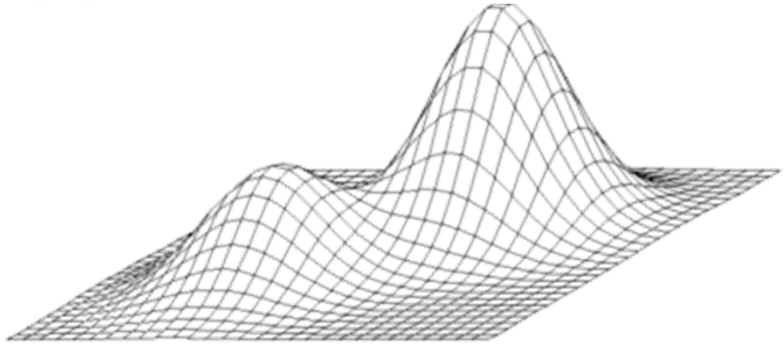
问题2： hill climbing (续)

- In computer science, hill climbing is a mathematical optimization technique which belongs to the family of local search. It is an iterative algorithm that starts with an arbitrary solution to a problem, then attempts to find a better solution by incrementally changing a single element of the solution. If the change produces a better solution, an incremental change is made to the new solution, repeating until no further improvements can be found.
  - 你认为hill climbing存在哪些问题？
    - 局部最优
    - 高原/肩膀
    - 缓升和山脊



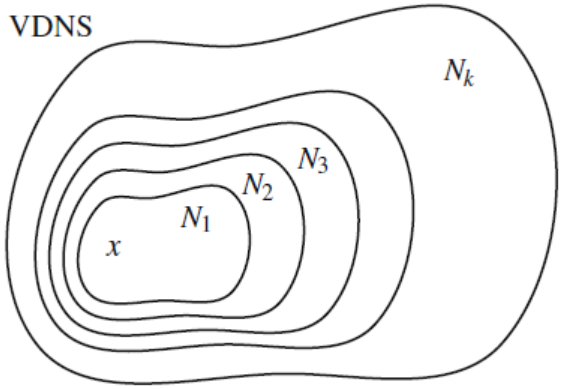


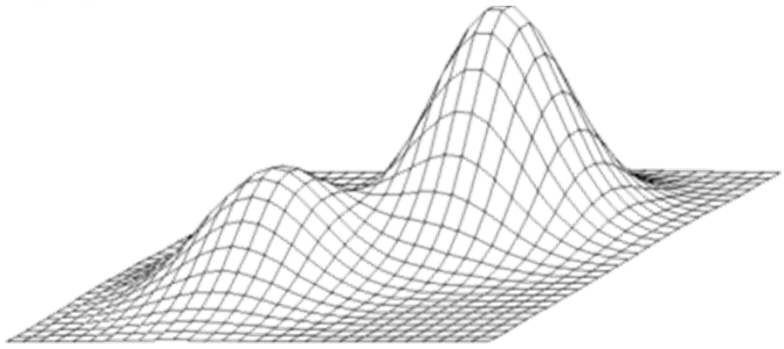
- 你能想到哪些策略来缓解这些问题？
  - α
  - Neigh
  - β



问题3：very large-scale neighborhood search

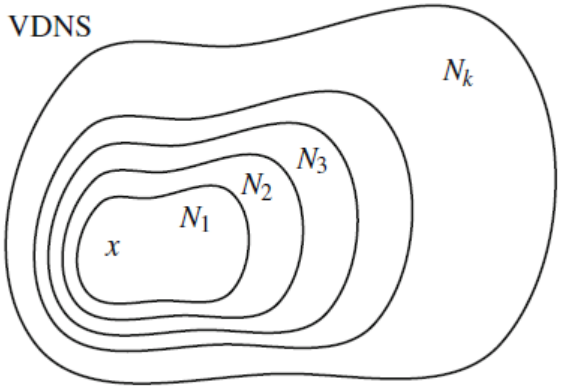
- A very large-scale neighborhood search is a local search algorithm which makes use of a neighborhood definition, which is large and possibly exponentially sized.
  - 和hill climbing相比，它改变了α、Neigh、β中的哪一个？
  - 这样做有什么好处？

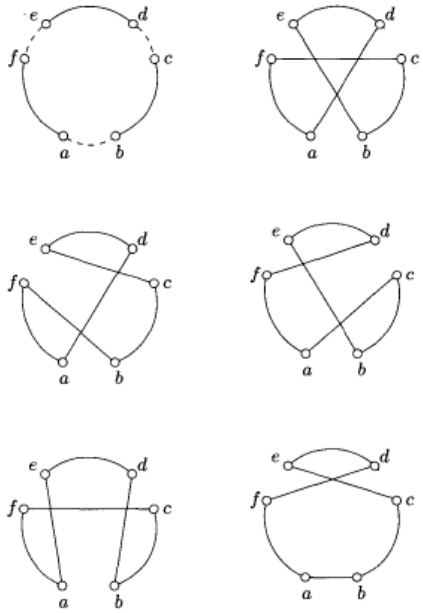




问题3：very large-scale neighborhood search (续)

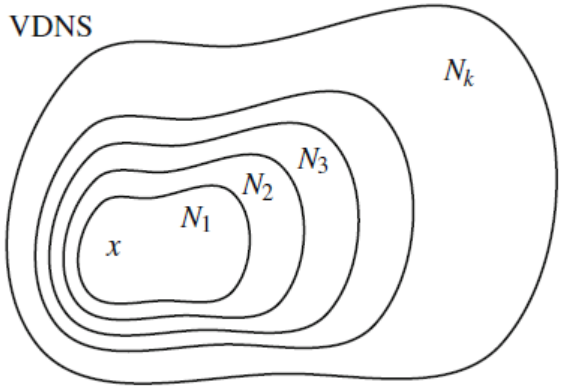
- A very large-scale neighborhood search is a local search algorithm which makes use of a neighborhood definition, which is large and possibly exponentially sized.
  - 你能以TSP为例，给出一个具体的算法吗？
  - 你给出的这个算法，存在什么问题？
  - 你有什么居中的策略来缓解这个问题？





问题3：very large-scale neighborhood search (续)

- Variable-depth search methods are techniques that search the *k*-exchange neighborhood partially, hence reducing the time used to search the neighborhood.
  - 你能想到什么方法来实现“partially”？



问题3：very large-scale neighborhood search (续)

- 你能用一两句话概述KL(Neigh)的基本思想吗？
  - Find an improvement a large distance but does not realize the exhaustive search of all feasible solutions.
  - Gradually extend the size of the neighborhood, each time the search gets trapped in a local minimum.

KL($Neigh$) Kernighan-Lin Variable-Depth Search Algorithm with respect to the neighborhood $Neigh$

Input: An input instance I of an optimization problem U.

Step 1: Generate a feasible solution $\alpha = (p_1, p_2, \ldots, p_n) \in \mathcal{M}(I)$ where $(p_1, p_2, \ldots, p_n)$ is such a parametric representation of α that the local transformation defining $Neigh$ can be viewed as an exchange of a few of these parameters.

Step 2: $IMPROVEMENT := TRUE$;
$EXCHANGE := \{1, 2, \ldots, n\}$; $J := 0$; $\alpha_J := \alpha$;
while $IMPROVEMENT = TRUE$ **do begin**
  while $EXCHANGE \neq \emptyset$ **do**
    begin $J := J + 1$;
      $\alpha_J :=$ a solution from $Neigh(\alpha_{J-1})$ such that $gain(\alpha_{J-1}, \alpha_J)$ is the maximum of
      $\{gain(\alpha_{J-1}, \delta) \mid \delta \in Neigh(\alpha_{J-1}) - \{\alpha_{J-1}\}$ and δ differs from α_{J-1} in the parameters of $EXCHANGE$ only$\}$;
      $EXCHANGE := EXCHANGE -$ {the parameters in which α_J and α_{J-1} differ}
  end;
  Compute $gain(\alpha, \alpha_i)$ for $i = 1, \ldots, J$;
  Compute $l \in \{1, \ldots, J\}$ such that

$$gain(\alpha, \alpha_l) = \max\{gain(\alpha, \alpha_i) \mid i \in \{1, 2, \ldots, J\}\};$$

  if $gain(\alpha, \alpha_l) > 0$ **then**
    begin $\alpha := \alpha_l$;
      $EXCHANGE := \{1, 2, \ldots, n\}$
    end
  else $IMPROVEMENT := FALSE$
end

Step 3: **output**(α).

问题3：very large-scale neighborhood search (续)

- 你能以TSP为例，给出一个具体的算法吗？

KL($Neigh$) Kernighan-Lin Variable-Depth Search Algorithm with respect to the neighborhood $Neigh$

Input: An input instance I of an optimization problem U.

Step 1: Generate a feasible solution $\alpha = (p_1, p_2, \ldots, p_n) \in \mathcal{M}(I)$ where $(p_1, p_2, \ldots, p_n)$ is such a parametric representation of α that the local transformation defining $Neigh$ can be viewed as an exchange of a few of these parameters.

Step 2: $IMPROVEMENT := TRUE$;
$EXCHANGE := \{1, 2, \ldots, n\}$; $J := 0$; $\alpha_J := \alpha$;
while $IMPROVEMENT = TRUE$ **do begin**
  while $EXCHANGE \neq \emptyset$ **do**
    begin $J := J + 1$;
      $\alpha_J :=$ a solution from $Neigh(\alpha_{J-1})$ such that $gain(\alpha_{J-1}, \alpha_J)$ is the maximum of
      $\{gain(\alpha_{J-1}, \delta) \mid \delta \in Neigh(\alpha_{J-1}) - \{\alpha_{J-1}\}$ and δ differs from α_{J-1} in the parameters of $EXCHANGE$ only$\}$;
      $EXCHANGE := EXCHANGE -$ {the parameters in which α_J and α_{J-1} differ}
  end;
  Compute $gain(\alpha, \alpha_i)$ for $i = 1, \ldots, J$;
  Compute $l \in \{1, \ldots, J\}$ such that

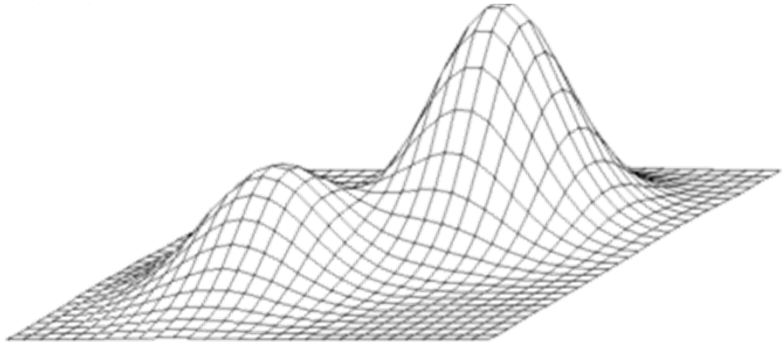
$$gain(\alpha, \alpha_l) = \max\{gain(\alpha, \alpha_i) \mid i \in \{1, 2, \ldots, J\}\};$$

  if $gain(\alpha, \alpha_l) > 0$ **then**
    begin $\alpha := \alpha_l$;
      $EXCHANGE := \{1, 2, \ldots, n\}$
    end
  else $IMPROVEMENT := FALSE$
end

Step 3: **output**(α).

问题4： Multi-start methods

- Re-start the procedure from a new solution once a region has been explored.
  - 和hill climbing相比，它改变了α、Neigh、β中的哪一个？
  - 这样做有什么好处？
  - 该选择什么样的new solution来re-start呢？



问题4： Multi-start methods (续)

- Greedy Randomized Adaptive Search Procedure (GRASP)
  - The GRASP metaheuristic is a multi-start or iterative process, in which each iteration consists of two phases: construction and local search. The construction phase builds a feasible solution, whose neighborhood is investigated until a local minimum is found during the local search phase. The best overall solution is kept as the result.

```
procedure GRASP(Max_Iterations,Seed)
1    Read_Input();
2    for k = 1,... ,Max_Iterations do
3        Solution ← Greedy_Randomized_Construction(Seed);
4        Solution ← Local_Search(Solution);
5        Update_Solution(Solution,Best_Solution);
6    end;
7    return Best_Solution;
end GRASP.
```

FIGURE 1. Pseudo-code of the GRASP metaheuristic.

```
procedure Greedy_Randomized_Construction(Seed)
1    Solution ← ∅;
2    Evaluate the incremental costs of the candidate elements;
3    while Solution is not a complete solution do
4        Build the restricted candidate list (RCL);
5        Select an element s from the RCL at random;
6        Solution ← Solution ∪ {s};
7        Reevaluate the incremental costs;
8    end;
9    return Solution;
end Greedy_Randomized_Construction.
```

FIGURE 2. Pseudo-code of the construction phase.

Greedy、Randomized、Adaptive
分别是怎么体现的？

```
procedure Local_Search(Solution)
1    while Solution is not locally optimal do
2        Find s' ∈ N(Solution) with f(s') < f(Solution);
3        Solution ← s';
4    end;
5    return Solution;
end Local_Search.
```

FIGURE 3. Pseudo-code of the local search phase.

问题4： Multi-start methods (续)

- 你能以TSP为例，给出一个具体的算法吗？

```
procedure GRASP(Max_Iterations,Seed)
1    Read_Input();
2    for k = 1,... ,Max_Iterations do
3        Solution ← Greedy_Randomized_Construction(Seed);
4        Solution ← Local_Search(Solution);
5        Update_Solution(Solution,Best_Solution);
6    end;
7    return Best_Solution;
end GRASP.
```

FIGURE 1. Pseudo-code of the GRASP metaheuristic.

```
procedure Greedy_Randomized_Construction(Seed)
1    Solution ← ∅;
2    Evaluate the incremental costs of the candidate elements;
3    while Solution is not a complete solution do
4        Build the restricted candidate list (RCL);
5        Select an element s from the RCL at random;
6        Solution ← Solution ∪ {s};
7        Reevaluate the incremental costs;
8    end;
9    return Solution;
end Greedy_Randomized_Construction.
```

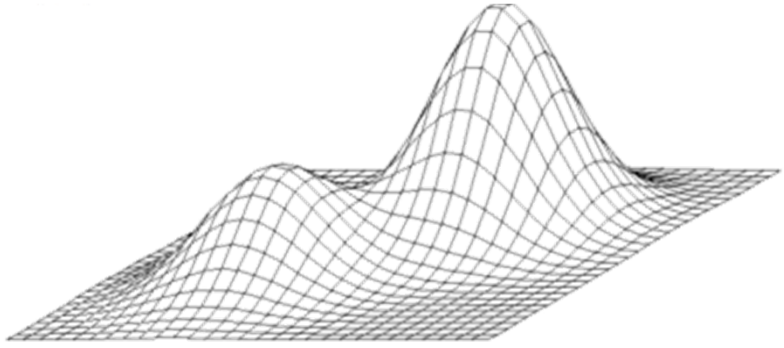
FIGURE 2. Pseudo-code of the construction phase.

```
procedure Local_Search(Solution)
1    while Solution is not locally optimal do
2        Find s' ∈ N(Solution) with f(s') < f(Solution);
3        Solution ← s';
4    end;
5    return Solution;
end Local_Search.
```

FIGURE 3. Pseudo-code of the local search phase.

问题5：Stochastic hill climbing

- Stochastic hill climbing chooses at random from among the uphill moves; the probability of selection can vary with the steepness of the uphill move.
  - 和hill climbing相比，它改变了α、Neigh、β中的哪一个？
  - 这样做有什么好处？
  - 你能以TSP为例，给出一个具体的算法吗？



- 你还能想到什么别的方法？
  - 提示：计算机除了“算”以外，还能做什么？

问题6：Tabu search

- Tabu search enhances the performance of local searches by using memory structures that describe the visited solutions or user-provided sets of rules.
  - Short-term: The list of solutions recently considered. If a potential solution appears on the tabu list, it cannot be revisited until it reaches an expiration point.
  - Intermediate-term: Intensification rules intended to bias the search towards promising areas of the search space.
  - Long-term: Diversification rules that drive the search into new regions (i.e. regarding resets when the search becomes stuck in a plateau or a suboptimal dead-end).

- 你能以TSP为例，给出一个具体的算法吗？

问题7：local search的性能

LSS($Neigh$)-Local Search Scheme according to a neighborhood $Neigh$

Input: An input instance x of an optimization problem U.
Step 1: Find a feasible solution $\alpha \in \mathcal{M}(x)$.
Step 2: **while** $\alpha \notin LocOPT_U(x, Neigh_x)$ **do**
begin find a $\beta \in Neigh_x(\alpha)$ such that
$cost(\beta) < cost(\alpha)$ if U is a minimization problem and
$cost(\beta) > cost(\alpha)$ if U is a maximization problem; $\alpha := \beta$
end
Output: **output**(α).

- local search的运算时间受哪些环节的影响？

问题7：local search的性能 (续)

- 什么叫exact polynomial-time searchable neighborhood？
- 为什么cost-bounded integer-valued TSP没有这一性质？

问题8：应用

- 你能基于hill climing、variable-depth search、GRASP、Tabu为下列问题设计一种local search算法吗
  - longest simple path
  - MAX-SAT
  - MAX-CL
  - MIN-VCP