Parameterized Complexity of Local Search

Jiong Guo
Local Search

Local search is a meta-heuristic widely applied in practice for solving (hard) combinatorial problems.

- Meta-heuristic = applicable to (almost) any problem.
Local Search

Local search is a meta-heuristic widely applied in practice for solving combinatorially hard problems.

- Meta-heuristic = applicable to (almost) any problem.

Extremely popular because:

- Easy to understand
- Easy to implement
- Generic
- Actually works (sometimes...)
  - Linear Programming (simplex)
  - Traveling Salesman
  - ...
Local Search

Basic idea:

1. Start with some solution $S$.

2. Local improvement:
   Search for a better solution $S'$ in the "neighborhood" $N_k(S)$, where $k$ is a predetermined "radius".

3. If such an $S'$ exists, repeat 1+2 with $S'$.

4. Otherwise, return $S'$. 
Analyzing Local Search

- How many local improvement steps are necessary?
  - PLS-completeness [Johnson, Papadimitriou, and Yannakakis, 1988]
Analyzing Local Search

- How many local improvement steps are necessary?
  - PLS-completeness [Johnson, Papadimitriou, and Yannakakis, 1988]

- How easy is the local improvement step?
  - Typically, the size of a $k$-radius neighborhood = $O(n^k)$.
  - Brute-force = $O(n^k)$ time.
  - Can we do better? — $f(k)n^{O(1)}$ time.
  - Parameterized Complexity
**Parameterized Complexity**

**Definition** Parameterized local improvement for optimization problems

**Input** An instance $G$, a solution $S^*$, a distance function $d$ of solutions, and an integer $k$.

**Output** A better solution $S$ and $d(S, S^*) \leq k$

fixed-parameter tractable—$f(k)|G|^{O(1)}$ vs. W[1]-hard
Parameterized Complexity

- Traveling Salesman  
  [E. Balas, 1999]  [D. Marx, 2008]  
  [Guo, Hartung, Niedermeier, Súchy, 2011]

- $r$-Center, Vertex Cover, Odd Cycle Transversal, Max Cut, Min Bisection  
  [Fellows, Fomin, Lokshtanov, Rosamond, Saurabh, Villanger, 2009]

- Feedback Arc Set on Tournaments  
  [Fomin, Lokshtanov, Raman, Saurabh, 2010]

- Stable Marriage  
  [Marx and Schlotter, 2001]

- Boolean Constraint Satisfiability  
  [Krokhin and Marx, 2012]

- Satisfiability  
  [S. Szeider, 2011]

- Cluster problems  
  [Dörnfelder, Guo Komusiewicz, Weller, 2011]

- String problems  
  [Guo, Hermelin, Komusiewicz, 2012]
Traveling Salesman

Definition Given two permutations $\pi, \pi'$ of $\{1, 2, \ldots, n\}$, the shift distance between $\pi$ and $\pi'$ is defined as

$$d(\pi, \pi') = \max_{1 \leq i \leq n} |\pi(i) - \pi'(i)|,$$

where $\pi(i)$ denotes the position of $i$ in $\pi$.

Local Improvement of Traveling Salesman

Input: A set of $n$ cities $\{1, \ldots, n\}$ with pairwise distance $c(i, j)$ between the cities, a tour $\pi^*$, and an integer $k$.

Output: A tour $\pi$ with $c(\pi) < c(\pi^*)$ and $d(\pi, \pi^*) \leq k$.

Assume $\pi^*$ starts and ends at 1 and is equal to the identity permutation.
Traveling Salesman

Theorem  [E. Balas, 1999]
Local Improvement of Traveling Salesman can be solved in $O(k^22^{2(k-1)}n)$ time.

Proof Reduction to the Shortest Path problem on a graph with $O(n \cdot (k + 1)2^{2(k-1)})$ vertices and a maximum out-degree of $2k$. 
Traveling Salesman

Definition Given a permutation $\pi$ of $\{1, 2, \ldots, n\}$ and $1 \leq i \leq n$, define

$$S^- (\pi, i) := \{ l \geq i \mid \pi(l) < i \}$$

and

$$S^+ (\pi, i) := \{ l < i \mid \pi(l) \geq i \}.$$
Observation For all feasible $\pi$ and all $1 \leq i \leq n$,

$$|S^- (\pi, i)| = |S^+ (\pi, i)| \leq k.$$
**Traveling Salesman**

Case: $j > i$ and $i \in s_{i}^{-}$:

- $s_{i+1}^{-} = s_{i}^{-} \cup \{j\} \setminus \{i\}$
- $s_{i+1}^{+} = s_{i}^{+}$
Traveling Salesman

Maximum out-degree $\leq 2k$. 
**Closest String**

**Definition** Given two length-$n$ strings $S$ and $S'$, the Hamming distance $d_H(S, S')$ is equal to the number of different positions in $S$ and $S'$.

**Closest String**

**Input** A set of length-$n$ strings $T_1, T_2, \ldots, T_m$

**Output** A length-$n$ string $S$ minimizing

$$\max_{1 \leq i \leq m} d_H(S, T_i).$$

**Example:**

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 0 0</td>
<td>1 1 1 0</td>
<td>0 0 0 1</td>
<td>1 1 0 0</td>
</tr>
</tbody>
</table>

$S := 1 1 0 1$
Hamming neighborhood: \( N_k(S) := \{S'|d_H(S, S') \leq k\}. \)

Local Improvement of Closest String

**Input**
A set of length-\(n\) strings \(T_1, T_2, \ldots, T_m\), a string \(S^*\), and an integer \(k\).

**Output**
A string \(S\) with \(S \in N_k(S^*)\) and 
\[\max_{1 \leq i \leq m} d_H(S, T_i) < \max_{1 \leq i \leq m} d_H(S^*, T_i)\]

**Example:** \(k = 1\)

\[
\begin{align*}
T_1 &: 0 1 0 0 \\
T_2 &: 1 1 1 0 \\
T_3 &: 0 0 0 1 \\
T_4 &: 1 1 0 0 \\
S^* &: 1 1 1 1 \\
S &: 1 1 0 1
\end{align*}
\]
Parameterized reductions:

\[(I, k) \in P \iff (I', k') \in Q\]

- \(f(k) \cdot |I|^{O(1)}\)
- \(k' = g(k)\)

\(P \text{ W[1]-hard} \implies Q \text{ W[1]-hard.}\)
Multicolored Hitting Set (MHS)

**Input** A set $V = \{v_1, v_2, \ldots, v_n\}$ colored by $k$ colors, and a family of subsets $E_1, \ldots, E_m$ of $V$

**Output** A subset $H \subseteq V$ of $k$ distinctly colored elements such that $H \cap E_i \neq \emptyset$ for all $i$'s.
Closest String

Multicolored Hitting Set (MHS)

Input A set $V = \{v_1, v_2, \ldots, v_n\}$ colored by $k$ colors, and a family of subsets $E_1, \ldots, E_m$ of $V$.

Output A subset $H \subseteq V$ of $k$ distinctly colored elements such that $H \cap E_i \neq \emptyset$ for all $i$'s.
Closest String

Warm up: $k + 2$ letters:

$$T_1 := \begin{bmatrix} G & R & 0 & 0 \\ \end{bmatrix}$$

$$T_2 := \begin{bmatrix} G & R & G & 0 \\ \end{bmatrix}$$

$$T_3 := \begin{bmatrix} 0 & R & G & R \\ \end{bmatrix}$$

$$T_4 := \begin{bmatrix} G & 0 & G & 0 \\ \end{bmatrix}$$

$$T_5 := \begin{bmatrix} 0 & 0 & 0 & R \\ \end{bmatrix}$$

$$T_6 := \begin{bmatrix} 0 & R & 0 & R \\ \end{bmatrix}$$

$$T_G := \begin{bmatrix} G & G & G & G \\ \end{bmatrix}$$

$$T_R := \begin{bmatrix} R & R & R & R \\ \end{bmatrix}$$

$$S^* := \begin{bmatrix} 1 & 1 & 1 & 1 \\ \end{bmatrix}$$
Closest String

Warm up: \( k + 2 \) letters:

\[
T_1 := \begin{array}{cccc}
G & R & 0 & 0
\end{array}
\]
\[
T_2 := \begin{array}{cccc}
G & R & G & 0
\end{array}
\]
\[
T_3 := \begin{array}{cccc}
0 & R & G & R
\end{array}
\]
\[
T_4 := \begin{array}{cccc}
G & 0 & G & 0
\end{array}
\]
\[
T_5 := \begin{array}{cccc}
0 & 0 & 0 & R
\end{array}
\]
\[
T_6 := \begin{array}{cccc}
0 & R & 0 & R
\end{array}
\]
\[
T_G := \begin{array}{cccc}
G & G & G & G
\end{array}
\]
\[
T_R := \begin{array}{cccc}
R & R & R & R
\end{array}
\]
\[
S^* := \begin{array}{cccc}
1 & 1 & 1 & 1
\end{array}
\]
\[
S := \begin{array}{cccc}
G & 1 & 1 & R
\end{array}
\]
Closest String

Binary strings

Using long strings with two parts: encoding and padding

- \( S^* = \) all-0 string,
- Any column in the padding part will contain at most one 1.
- Will force changes to be done only in the encoding part of \( S^* \).
Closest String

Binary strings

- $m$ strings to encode $E_i$'s.
- Pad so that each has exactly $n - k$ 1's.

$T_1 := 1 1 0 0 ...$
$T_2 := 1 1 1 0 ...$
$T_3 := 0 1 1 1 ...$
$T_4 := 1 0 1 0 ...$
$T_5 := 0 0 0 1 ...$
$T_6 := 0 1 0 1 ...$
Closest String

Binary strings

$E_1\ E_2\ E_3\ E_4\ E_5\ E_6$

$v_1\ v_2\ v_3\ v_4$

- $2^k - 1$ strings to encode each of the proper subset of colors.
- Pad so that $T_C$ has exactly $n - \left| C \right|$ 1’s.

$T_0 := 1\ 1\ 1\ 1\ \ldots$

$T_{\{G\}} := 0\ 1\ 0\ 1\ \ldots$

$T_{\{R\}} := 1\ 0\ 1\ 0\ \ldots$
Closest String

Binary strings

\[
\begin{align*}
T_1 & := 1 1 0 0 \\
T_2 & := 1 1 1 0 \\
T_3 & := 0 1 1 1 \\
T_4 & := 1 0 1 0 \\
T_5 & := 0 0 0 1 \\
T_6 & := 0 1 0 1 \\
T_0 & := 1 1 1 1 \\
T_{\{G\}} & := 0 1 0 1 \\
T_{\{R\}} & := 1 0 1 0 \\
S^* & := 0 0 0 0 
\end{align*}
\]
Closest String

Binary strings

$v_1$ $v_4$

$E_1$ $E_2$ $E_3$ $E_4$ $E_5$ $E_6$

$T_1 := 1 1 0 0 ...$
$T_2 := 1 1 1 0 ...$
$T_3 := 0 1 1 1 ...$
$T_4 := 1 0 1 0 ...$
$T_5 := 0 0 0 1 ...$
$T_6 := 0 1 0 1 ...$

$T_\emptyset := 1 1 1 1 ...$
$T_{\{G\}} := 0 1 0 1 ...$
$T_{\{R\}} := 1 0 1 0 ...$

$S^* := 0 0 0 0 ...$
$S := 1 0 0 1 ...$
Closest String

Binary strings

\begin{align*}
T_0 & := 1 \ 1 \ 1 \ 1 \ \ldots \\
T_{\{G\}} & := 0 \ 1 \ 0 \ 1 \ \ldots \\
T_{\{R\}} & := 1 \ 0 \ 1 \ 0 \ \ldots \\
S^* & := 0 \ 0 \ 0 \ 0 \ \ldots \\
S & := 1 \ 0 \ 0 \ 1 \ \ldots 
\end{align*}

- \( d_H(S, T_C) < n \implies \) selected vertices have a color not in \( C \).

- Selected vertices have exactly one of each color.
Closest String

Binary strings

- \( d_H(S, T_{E_i}) < n \implies \text{selected vertices hit } E_i \).
- Selected vertices form a multicolored hitting set.
The local improvement of Closest String is $W[2]$-hard even for binary alphabets.

**Exponential Time Hypothesis (ETH):** $n$-variable 3-SAT not solvable in $2^{o(n)}$ time.

**Corollary** The local improvement of Closest String cannot be solved in $n^{o(k)} \cdot \text{poly}(n, m)$, assuming ETH.

Brute-force is essentially optimal!
Conclusion

• Until now, the local improvement of most problems W[1]-hard
  – W[1]-hard
    * TSP with edge exchange/swap/reveral/..., string problems with Hamming distance, subgraph problems with set difference, clustering problem with KT-distance, ...
  – FPT
    * TSP with shift distance, subgraph problems on special graphs, ...

• Ways out of the dilemma:
  – the starting solution $S^*$
  – new distance measures
  – ...

• Experimental studies

[Simonetti and Balas, 1996]