Maksim Nikolaev

# All Instantiations of the Greedy Algorithm for the Shortest Common Superstring Problem are Equivalent

SPIRE 2021, October 5th

# Shortest Common Superstring Problem

Input: A set $\{s_1, \dots, s_n\}$ of n strings.

Output: A shortest string containing each $s_i$ as a substring.

Complexity: MAX-SNP-hard

Practical applications: data storage, data compression, genome assembly

Example

$\mathcal{S} = \{\text{aab, aaa, baa}\}$,

Solution: baaab

# Known approximation algorithms

| | | |
|---|---|---|
| 3.000 | Blum, Jiang, Li, Tromp, Yannakakis | 1991 |
| 2.889 | Teng, Yao | 1993 |
| 2.834 | Czumaj, Gasieniec, Piotrow, Rytter | 1994 |
| 2.794 | Kosaraju, Park, Stein | 1994 |
| 2.750 | Armen, Stein | 1994 |
| 2.725 | Armen, Stein | 1995 |
| 2.667 | Armen, Stein | 1996 |
| 2.596 | Breslauer, Jiang, Jiang | 1997 |
| 2.500 | Sweedyk | 1999 |
| 2.500 | Kaplan, Lewenstein, Shafrir, Sviridenko | 2005 |
| 2.500 | Paluch, Elbassioni, van Zuylen | 2012 |
| 2.479 | Mucha | 2013 |

# Known approximation algorithms

| | | |
|---|---|---|
| 3.000 | Blum, Jiang, Li, Tromp, Yannakakis | 1991 |
| 2.889 | Teng, Yao | 1993 |
| 2.834 | Czumaj, Gasieniec, Piotrow, Rytter | 1994 |
| 2.794 | Kosaraju, Park, Stein | 1994 |
| 2.750 | Armen, Stein | 1994 |
| 2.725 | Armen, Stein | 1995 |
| 2.667 | Armen, Stein | 1996 |
| 2.596 | Breslauer, Jiang, Jiang | 1997 |
| 2.500 | Sweedyk | 1999 |
| 2.500 | Kaplan, Lewenstein, Shafrir, Sviridenko | 2005 |
| 2.500 | Paluch, Elbassioni, van Zuylen | 2012 |
| 2.479 | Mucha | 2013 |
| 2.000 | greedy? | ??? |

# Greedy Algorithm

| | |
|---|---|
| $s_1$ | aabab |
| $s_2$ |  ababb |
| overlap$(s_1, s_2)$ |  abab |
| merge$(s_1, s_2)$ | aababb |

When there is more then one string: take two strings with the largest overlap; merge them; repeat.

# Greedy Algorithm

| | |
|---|---|
| $s_1$ | aabab |
| $s_2$ |  ababb |
| overlap($s_1, s_2$) |  abab |
| merge($s_1, s_2$) | aababb |

When there is more then one string: take two strings with the largest overlap; merge them; repeat.

Greedy conjecture: the greedy algorithm is factor 2 approximation [Storer 1987].

# Greedy Algorithm

| | |
|---|---|
| $s_1$ | aabab |
| $s_2$ | ababb |
| overlap$(s_1, s_2)$ | abab |
| merge$(s_1, s_2)$ | aababb |

When there is more then one string: take two strings with the largest overlap; merge them; repeat.

Greedy conjecture: the greedy algorithm is factor 2 approximation [Storer 1987].

Known to be factor 3.5 approximation [Kaplan and Shafrir 2004].

# Greedy is at least factor 2 approximation!

Dataset: $\{c(ab)^n, (ba)^n, (ab)^n c\}$

Greedy solution: $\{c(ab)^n, (ba)^n, (ab)^n c\} \rightarrow \{c(ab)^n c, (ba)^n\} \rightarrow \{c(ab)^n c(ba)^n\}$, length $= 4n + 2$

Optimal solution: $ca(ba)^n bc$, length $= 2n + 4$

# Greedy is non-deterministic!

Several pairs with the longest overlap ⇒ several possible merges ⇒ several possible superstrings.

Dataset: $\{ab^n, b^{n+1}, b^n a\}$

Greedy solution 1: $\{ab^n, b^{n+1}, b^n a\} \to \{ab^{n+1}, b^n a\} \to \{ab^{n+1}a\}$, length $= n + 3$

Greedy solution 2: $\{ab^n, b^{n+1}, b^n a\} \to \{ab^n a, b^{n+1}\} \to \{ab^n ab^{n+1}\}$, length $= 2n + 3$

# Maybe prove something weaker?

algorithm with specific tie-braking rule
||
To prove Greedy Conjecture, one needs to show that all instantiations of the Greedy Algorithm are factor 2 approximation.

Maybe it is easier to find at least one factor 2 approximation instantiation?

# Maybe prove something weaker?

algorithm with specific tie-braking rule

||

To prove Greedy Conjecture, one needs to show that all instantiations of the Greedy Algorithm are factor 2 approximation.

Maybe it is easier to find at least one factor 2 approximation instantiation?

**Main result: all instantiations of the Greedy Algorithm have the same approximation factor.**

# Idea behind the proof

**Perturbing Procedure**

Input: a dataset $\mathcal{S}$, an instantiation $A$ of the Greedy Algorithm ($A \in \mathrm{GA}$), $\varepsilon > 0$

Output: a dataset $\mathcal{S}'$ such that:

1. $\frac{|A(\mathcal{S})|}{|\mathrm{OPT}(\mathcal{S})|} - \varepsilon < \frac{|A(\mathcal{S}')|}{|\mathrm{OPT}(\mathcal{S}')|}$.
2. There is only one sequence of non-trivial greedy merges $\Rightarrow |A(\mathcal{S}')| = |B(\mathcal{S}')|, \forall\, B \in \mathrm{GA}$.

||

merge with non-empty overlap

# Perturbing procedure

$\mathcal{S} = \{\text{abb}, \text{bbb}, \text{bbc}\}$

How to make the merge $\{\text{abb}, \text{bbb}, \text{bbc}\} \rightarrow \{\text{abbc}, \text{bbb}\}$ the only greedy merge?

# Perturbing procedure

$\mathcal{S} = \{\text{abb}, \text{bbb}, \text{bbc}\}$

How to make the merge {abb, bbb, bbc} → {abbc, bbb} the only greedy merge?

Step1: {abb, bbb, bbc} → {$\$^{10}$a $\$^{10}$b $\$^{10}$b, $\$^{10}$b $\$^{10}$b $\$^{10}$b, $\$^{10}$b $\$^{10}$b $\$^{10}$c}

# Perturbing procedure

$\mathcal{S} = \{abb, bbb, bbc\}$

How to make the merge $\{abb, bbb, bbc\} \to \{abbc, bbb\}$ the only greedy merge?

Step1: $\{abb, bbb, bbc\} \to \{\$^{10}a\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}c\}$

Step2: $\{\$^{10}a\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}c\} \to$

$\to \{\$^{10}a\ \$^{10}b\ \$^{10}b\$,\ \$^{9}b\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}c\}$

$\text{overlap}(\$^{10}a\ \$^{10}b\ \$^{10}b\$,\ \$^{9}b\ \$^{10}b\ \$^{10}b) = 22$

$\text{overlap}(\$^{9}b\ \$^{10}b\ \$^{10}b,\ \$^{10}b\ \$^{10}b\ \$^{10}c) = 22$

$\text{overlap}(\$^{10}a\ \$^{10}b\ \$^{10}b\$,\ \$^{10}b\ \$^{10}b\ \$^{10}c) = 23$

# Perturbing procedure

For $\mathcal{S} = \{s_1, \dots, s_n\}$ and $A \in \text{GA}$ let $\big(l_A(1), r_A(1)\big), \big(l_A(2), r_A(2)\big), \dots, \big(l_A(n-1), r_A(n-1)\big)$ be the order of merges: strings $s_{l_A(i)}$ and $s_{r_A(i)}$ are merged at step $i$.

If $|\text{overlap}(s_{l_A(i)}, s_{r_A(i)})| = 0$ for some $i$, then the same holds for any $i' > i$. Let $T_A$ be the first such $i$. This is the first trivial merge. If there were no trivial merges, $T_A = n$.

# Perturbing procedure

Input: a dataset $\mathcal{S}$, an instantiation $A$ of the Greedy Algorithm ($A \in \mathrm{GA}$), $\varepsilon > 0$.

For every $s_i = c_1 c_2 \ldots c_{|s_i|} \in \mathcal{S}$ define a string

$$s_i' = \$^{m-\alpha_i}\, c_1\, \$^m\, c_2\, \$^m\, c_3\, \$^m \ldots \$^m\, c_{|s_i|}\, \$^{T_A-\beta_i},$$

where

- $\$$ is a sentinel — symbol which does not occur in $\mathcal{S}$,
- $m$ is a parameter that depends on $\varepsilon$,
- $\alpha_i$ is the number of step such that $r_A(\alpha_i) = i$, if such step exists and $< T_A$, and $\alpha_i = T_A$ otherwise;
- $\beta_i$ is the number of step such that $l_A(\beta_i) = i$, if such step exists and $< T_A$, and $\beta_i = T_A$ otherwise.

Order: $(1,5), (3,2), (5,4), (2,1), T_A = 3 \Rightarrow \beta_1 = \alpha_5 = 1, \beta_3 = \alpha_2 = 2, \beta_5 = \alpha_4 = \beta_2 = \alpha_1 = \beta_4 = \alpha_3 = 3.$

# Perturbing procedure

As $m \to \infty$:

$$\frac{1}{m}|\mathrm{OPT}(\mathcal{S}')| \to |\mathrm{OPT}(\mathcal{S})|,$$

$$\frac{1}{m}|A(\mathcal{S}')| \to |A(\mathcal{S})|,$$

so we can choose $m$ such that $\frac{|A(\mathcal{S})|}{|\mathrm{OPT}(\mathcal{S})|} - \varepsilon < \frac{|A(\mathcal{S}')|}{|\mathrm{OPT}(\mathcal{S}')|}$.

Since $|B(\mathcal{S}')| = |A(\mathcal{S}')|, \forall B \in \mathrm{GA}$, we have $\frac{|B(\mathcal{S}')|}{|\mathrm{OPT}(\mathcal{S}')|} = \frac{|A(\mathcal{S}')|}{|\mathrm{OPT}(\mathcal{S}')|}$.

# Corollaries

To prove (or disprove) the Greedy Conjecture, it is sufficient to consider datasets satisfying some of the following three properties:

1. there are no ties between non-empty overlaps, that is, datasets where all the instantiations of the greedy algorithm work the same;
2. there are no empty overlaps: overlap$(s_i, s_j) \neq \varepsilon, \forall, i \neq j$;
3. all non-empty overlaps are (pairwise) different: $|$overlap$(s_i, s_j)| \neq |$overlap$(s_k, s_l)|$, for all $i \neq j$, $k \neq l$, $(i, j) \neq (k, l)$.

# Corollaries

To prove (or disprove) the Greedy Conjecture, it is sufficient to consider datasets satisfying some of the following three properties:

1. there are no ties between non-empty overlaps, that is, datasets where all the instantiations of the greedy algorithm work the same;
2. there are no empty overlaps: overlap$(s_i, s_j) \neq \varepsilon, \forall, i \neq j$;
3. all non-empty overlaps are (pairwise) different: $|$overlap$(s_i, s_j)| \neq |$overlap$(s_k, s_l)|$, for all $i \neq j$, $k \neq l$, $(i, j) \neq (k, l)$.

**Thank you for your attention!**

Ask your questions: makc-nicko@yandex.ru