Rotations of Periodic Strings and Short Superstrings

Tao Jiang
University of California - Riverside

Joint work with

Dany Breslauer
MPI für Informatik

Zhigen Jiang
McMaster University

The Shortest Superstring Problem (I)

Given a set of strings $S = \{s_1, \ldots, s_m\}$, find a shortest superstring s that contains all s_i as substrings.

Example. “Alf ate half lethal alpha alfalfa.”

$$S = \{\text{alf}, \text{ate}, \text{half}, \text{lethal}, \text{alpha}, \text{alfalfa}\}$$

$w_1 =$ atehalflethalalphaalfalfa $|w_1| = 25$

$w_2 =$ lethalfalfalphate $|w_2| = 17$

$w_3 =$ lethalphalfalfate $|w_3| = 17$

Asume the input set S is *substring-free*.

The Shortest Superstring Problem (II)

Applications:

1. Data compression. A set of strings can be represented by a superstring and positions of the strings in the superstring and their lengths.
2. DNA sequencing. State-of-the-art biochemistry can sequence a fragment of about 500 nucleotides. Longer DNA molecules are “cut” into short overlapping fragments that are sequenced separately. These fragments are then “assembled” by a shortest superstring algorithm.

The Shortest Superstring Problem (III)

The shortest superstring problem is MAX-SNP hard [GMS80,B

Try to approximate!

Denote as opt(S) the *length* of a shortest superstring and as maxov(S),

$$\text{maxov}(S) = \sum_{s_i \in S} |s_i| - \text{opt}(S),$$

the *compression* or total *overlap* between strings in a shortest superstring.

maxov(S) is the maximum overlap in *any* superstring!

Overlap approximation seems to be easier than length approximation.

[TU88] and [T89] prove that “the GREEDY” algorithm $\frac{1}{2}$-approximates the overlap.

Conjecture: “The GREEDY” algorithm 2-approximates the length. [BJLTY94] prove that it 4-approximates the length.

The Shortest Superstring Problem (IV)

Length Approximation Algorithms		
Blum, Jiang, Li, Tromp and Yannakakis	91	3
Teng and Yao	93	2.89
Czumaj, Gąsieniec Piotrow and Rytter	94	2.83
Kosaraju, Park and Stein	94	2.79
Armen and Stein	95	2.75
Armen and Stein	96	2.67
This work	96	2.67
This work	96	2.596
Sweedyk	96	2.5 ???
Overlap Approximation Algorithm		
Kosaraju, Park and Stein	94	0.603

Basic Notations and Facts

Given strings s and t, let y be the *longest* string such that $s = xy$ and $t = yz$, for some *non-empty* x and z.

$$\begin{array}{rcl} s & = & \text{lethal} \\ & & \quad\;\;\text{half} \;=\; t \end{array}$$

$$\begin{aligned} ov(s,t) &= 3 \\ pref(s,t) &= \text{let} \\ d(s,t) &= 3 \\ \langle s,t\rangle &= \text{lethalf} \end{aligned}$$

$\langle s_{i_1},\ldots,s_{i_r}\rangle$ is the shortest string containing the strings in the specified order:

$$\langle s_{i_1},\ldots,s_{i_r}\rangle = pref(s_{i_1},s_{i_2})\cdots pref(s_{i_{r-1}},s_{i_r})\ s_{i_r}$$

$$\langle \text{lethal}, \text{half}, \text{alfalfa}\rangle = \text{lethalfalfa}$$

Claim: The shortest superstring for $S = \{s_1,\ldots,s_m\}$ is $\langle s_{\pi(1)},\ldots,s_{\pi(m)}\rangle$ for some permutation π.

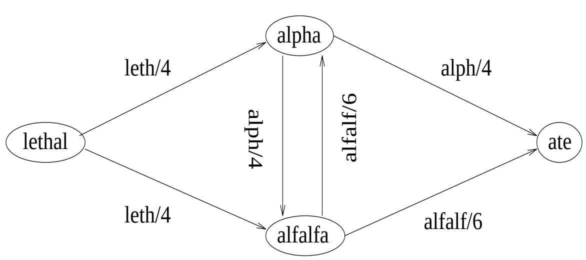
$$\text{maxov}(S) = \sum_{i=1}^{m} |s_i| - \text{opt}(S) = \sum_{i=1}^{m-1} ov(s_{\pi(i)}, s_{\pi(i+1)})$$

The Distance Graph

Given a set of strings $S = \{s_1,\ldots,s_m\}$, the distance graph G_S has m vertices $s_1,\ldots,s_m$.

The edge (s_i,s_j), $i \neq j$, has weight $d(s_i,s_j)$.

Example: G_S for $S = \{\text{ate}, \text{lethal}, \text{alpha}, \text{alfalfa}\}$:



TSP(G_S) is the minimum weight of any Hamiltonian cycle in G_S (*i.e.* optimal TSP solution).

Then, for each $s_i \in S$,

$$\text{TSP}(G_S) \le \text{opt}(S) \le \text{TSP}(G_S) + |s_i|.$$

$$\text{opt}(S) \Leftrightarrow pref(s_{\pi(1)}, s_{\pi(2)})\cdots pref(s_{\pi(k-1)}, s_{\pi(m)})\ s_{\pi(m)}$$

TSP(S) $\Leftrightarrow$

$$pref(s_{\pi(1)}, s_{\pi(2)})\cdots pref(s_{\pi(k-1)}, s_{\pi(m)}) pref(s_{\pi(m)}, s_{\pi(1)})$$

Cycle Covers

A cycle cover of a graph is a collection of disjoint cycles that cover all vertices.

CYC(G_S) is the minimum weight of any cycle cover of G_S.

Clearly CYC(G_S) $\le$ TSP(G_S) $\le$ opt(S).

opt(S) $\le$?CYC(G_S)?.

Good news:

We can compute CYC(G_S) in polynomial time.

This will be taken for granted in this talk.

Periods and Rotations of Strings

A string s has *a factor* x if $s = x^i y$, for some integer i and prefix y of x.

The factor of s, $f(s)$, is the shortest factor of s.

Denote the *period* of s as $p(s) = |f(s)|$.

Two strings s and t are equivalent if $f(s)$ is a rotation of $f(t)$. Namely if $f(s) = xy$ and $f(t) = yx$.

The factor $x = f(s)$ of a semi-infinite string s is the shortest string such that $s = x^\infty$.

Examples:

The string $aabaabaa$ has factors aab, $aabaab$, $aabaaba$, and $aabaabaa$.

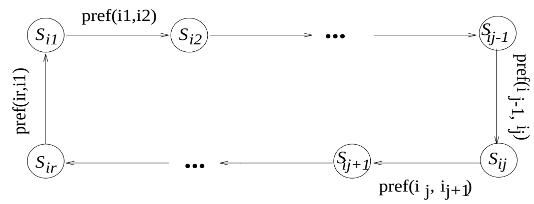
The factor $f(aabaabaa) = aab$.

The strings aba and baa are rotations of aab.

The string $(baa)^\infty$ is equivalent to $aabaabaa$.

Periods and Cycles

Let C be a minimum weight cycle cover of G_S and $c = s_{i_1}, \ldots, s_{i_r}, s_{i_1} \in C$.



Opening the cycle at i_j gives $\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_{j-1}} \rangle$.

$$f(\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_{j-1}} \rangle) = pref(s_{i_j}, s_{i_{j+1}}) \cdots pref(s_{i_{r-1}}, s_{i_r}) pref(s_{i_1}, s_{i_2}) \cdots pref(s_{i_{j-1}}, s_{i_j})$$

Lemma. These strings are all equivalent and have period $w(c) = d(s_{i_1}, s_{i_2}) + \cdots + d(s_{i_{r-1}}, s_{i_r}) + d(s_{i_r}, s_{i_1})$.

Lemma: Let $\hat{c} = s_{h_1}, \ldots, s_{h_l} \in C$ be another cycle. Then, $\langle s_{h_p}, \ldots, s_{h_l}, s_{h_1}, \ldots, s_{h_{p-1}} \rangle$ and $\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_{j-1}} \rangle$ are inequivalent.

The Generic Superstring Algorithm (I)

First part:

1. Construct the distance graph G_S.
2. Find a minimum weight cycle cover C in G_S.
3. Choose a *representative* string t_c for each cycle $c \in C$, such that for some j:
   (a) t_c contains $\langle s_{i_{j+1}}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle$, and
   (b) t_c is contained in $\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle$.
4. Let T be the set of representatives above.

Remark: The strings in T are pairwise *inequivalent*.

Lemma: opt$(T) \le$ opt$(S) +$ CYC$(S) \le$ 2opt(S).

Proof.

$$\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle = f(\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_{j-1}} \rangle) s_{i_j}$$

$$\text{opt}(\bigcup \{s_{i_j}\}) \le \text{opt}(S)$$

The Generic Superstring Algorithm (II)

Second part:

1. Construct the distance graph G_T.
2. Find a minimum weight cycle cover CC in G_T.
3. Open each cycle of CC arbitrarily.
4. Let R be the set of the strings obtained above.
5. Concatenate the strings in R to produce a superstring $\hat{s}$ of S.

Overlap Lemma: [FW65] For inequivalent strings s and t,

$$ov(s,t) \le p(s) + p(t).$$

Let OV be the overlap on the broken edges in all cycles of CC. Then,

$$OV \le \sum_{c \in C} p(t_c) = \sum_{c \in C} w(c) = \text{CYC}(G_S).$$

Conclusion: Recalling CYC$(G_T) \le$ opt$(T) \le$ 2opt(S).

$$|\hat{s}| = \text{CYC}(G_T) + OV \le \text{2opt}(S) + \text{opt}(S) \le \text{3opt}(S).$$

The Overlap-Rotation Lemma

Let $\alpha = a_1 a_2 \cdots$ be a semi-infinite string.

Denote a *rotation* $\alpha[k] = a_k a_{k+1} \cdots$.

There exists an integer k, such that for any finite string s that is *inequivalent* to α and satisfies $p(s) \leq p(\alpha)$,

$$ov(s, \alpha[k]) \leq \frac{2}{3}(p(s) + p(\alpha)).$$

Remarks:

The bound above is roughly tight as demonstrated by the string $\alpha = (0^n 1 0^{n+1} 1)^\infty$.

α is semi-infinite for convenience.

The Improved Algorithm (I)

First part:

Choose the representatives t_c *with care*.

Second part:

Break each cycle of CC by deleting an edge that goes from a string to another with equal or larger period.

Now,

$$OV \leq \frac{2}{3}\sum_{c \in C} p(t_c) = \frac{2}{3}\mathrm{CYC}(G_S) \leq \frac{2}{3}\mathrm{opt}(S).$$

And,

$$|\hat{s}| = \mathrm{CYC}(G_T) + OV \leq 2\frac{2}{3}\mathrm{opt}(S).$$

The Improved Algorithm (II)

How to choose the representative t_c, for the cycle $c = s_{i_1}, \ldots, s_{i_r}$.

1. Take $\alpha = (f(\langle s_{i_1}, \ldots, s_{i_r} \rangle))^\infty$.
2. Let β be the rotation of α with the properties of the overlap-rotation lemma.
3. Let $\langle s_{i_{j+1}}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle$ be the first such string that appears in β.
4. Take t_c to be the shortest prefix of β containing $\langle s_{i_{j+1}}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle$.

Clearly t_c is contained in $\langle s_{i_j}, \ldots, s_{i_r}, s_{i_1}, \ldots, s_{i_j} \rangle$.

The string t_c can be found in polynomial time.

The Second Improvement

O-R Lemma II: $\forall \alpha \exists k \forall s \;\; ov(s, \alpha[k]) \leq p(s) + \frac{1}{2}p(\alpha)$.

Lemma: If $\mathrm{apx}(T)$ is the length of the superstring of T produced by some δ overlap approximation algorithm, then,

$$\begin{aligned} \mathrm{apx}(T) &\leq \sum_{t_i \in T} |t_i| - \delta \; \mathrm{maxov}(T) \\ &= \mathrm{opt}(T) + (1 - \delta)\mathrm{maxov}(T). \end{aligned}$$

Lemma: Assume that a shortest superstring of T is $\langle t_1, \ldots, t_r \rangle$. Then,

$$\mathrm{maxov}(T) = \sum_{i=1}^{r-1} ov(t_i, t_{i+1}) \leq \sum_{i=1}^{r} \frac{3}{2}p(t_i) = \frac{3}{2}\mathrm{CYC}(G_S).$$

Consequence: Using the $\frac{38}{63}$ overlap approximation algorithm of [KPS94]:

$$\begin{aligned} \mathrm{apx}(T) &\leq \mathrm{opt}(T) + (1 - \frac{38}{63})\mathrm{maxov}(T) \\ &\leq 2\mathrm{opt}(S) + \frac{25}{63} \cdot \frac{3}{2}\mathrm{CYC}(G_S) \\ &\leq 2\frac{25}{42}\mathrm{opt}(S) \approx 2.596\mathrm{opt}(S). \end{aligned}$$

Proof of the O-R Lemma (I)

$\forall \alpha \exists k \forall s$, s inequivalent to α and $p(s) \leq p(\alpha)$,

$$ov(s, \alpha[k]) \leq \frac{2}{3}(p(s) + p(\alpha)).$$

A string w is *unbordered* if it has no proper prefix that is also a suffix. Namely, $f(w) = w$. *E.g.*, **ababb**.

A *non-trivial factorization* (u, v) of w is a non-empty prefix u and suffix v of $w = uv$.

The local factor of a factorization (u, v) of $w = uv$ is the shortest string x that is consistent with both sides of the factorization (u, v).

Example:

a | b a a a b a	a b | a a a b a	a b a | a a b a
ba ba	**aaab aaab**	**a a**
(a)	**(b)**	**(c)**

A factorization (u, v) of $w = uv$ is *critical* if its local factor x has length $p(w)$.

The Critical Factorization Theorem [CV78]:

Given any $p(w) - 1$ consecutive non-trivial factorizations of w, at least one is a critical factorization.

Proof of the O-R Lemma (II)

Lemma:

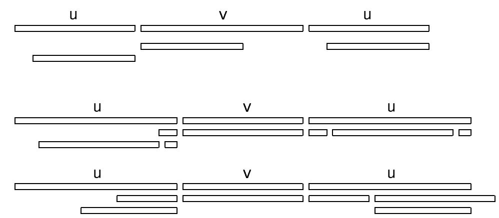
Let w be unbordered and have critical factorization (u, v). Then,

1. the rotation $w' = vu$ is also unbordered; and
2. (v, u) is a critical factorization of w'.

Proof: by contradiction.

1.

2.



Proof of the O-R Lemma (III)

Let α' be a rotation of α with $w = f(\alpha')$ unbordered.

Let $w = uv$ be a critical factorization.

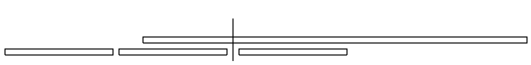
1. If $|u| \leq \frac{p(\alpha)}{2}$, $\beta = (uv)^\infty$ is the required rotation.
2. Otherwise $|v| \leq \frac{p(\alpha)}{2}$ and $\beta = (vu)^\infty$ is the required rotation.

Since the rotation β starts with an unbordered factor,

$$ov(s, \beta) < p(\alpha).$$

Since β has a critical factorization $\beta = x\beta'$ with $|x| \leq \frac{p(\alpha)}{2}$,

$$ov(s, \beta) < p(s) + |x| \leq p(s) + \frac{p(\alpha)}{2}.$$



Putting all this together we get:

$$ov(s, \beta) \leq \min(p(\alpha), p(s) + \frac{p(\alpha)}{2}) \leq \frac{2}{3}(p(s) + p(\alpha)).$$

Open Problems

1. Is GREEDY a 2-approximation?
2. Find better polynomial time approximation algorithms for length or for overlap.