Complexity of Approximating Closest Substring Problems

Patricia A. Evans[1] and Andrew D. Smith[1,2]

[1] University of New Brunswick, P.O. Box 4400, Fredericton N.B., E3B 5A3, Canada
pevans@unb.ca
[2] Ontario Cancer Institute, University Health Network, Suite 703
620 University Avenue, Toronto, Ontario, M5G 2M9 Canada
fax: +1-506-453-3566
asmith@uhnres.utoronto.ca

Abstract. The CLOSEST SUBSTRING problem, where a short string is sought that minimizes the number of mismatches between it and each of a given set of strings, is a minimization problem with a polynomial time approximation scheme [6]. In this paper, both this problem and its maximization complement, where instead the number of matches is maximized, are examined and bounds on their hardness of approximation are proved. Related problems differing only in their objective functions, seeking either to maximize the number of strings covered by the substring or maximize the length of the substring, are also examined and bounds on their approximability proved. For this last problem of length maximization, the approximation bound of 2 is proved to be tight by presenting a 2-approximation algorithm.

Keywords: Approximation algorithms; Hardness of approximation; Closest Substring

1 Introduction

Given a set $\mathcal{F}$ of strings, the CLOSEST SUBSTRING problem seeks to find a string $\mathcal{C}$ of a desired length l that minimizes the maximum distance from $\mathcal{C}$ to a substring in each member of $\mathcal{F}$. We call such a short string $\mathcal{C}$ a *center* for $\mathcal{F}$. The corresponding substrings from each string in $\mathcal{F}$ are the *occurrences* of $\mathcal{C}$. If all strings in $\mathcal{F}$ are the same length n, and the center is also to be of length n, then this special case of the problem is known as CLOSEST STRING. We examine the complexity of approximating three problems related to CLOSEST SUBSTRING with different objective functions. A center is considered to be *optimal* in the context of the problem under discussion, in that it either maximized or minimizes the problem's objective function. This examination of the problems' approximability with respect to their differing objective functions reveals interesting differences between the optimization goals.

In [6], a polynomial time approximation scheme (PTAS) is given for CLOSEST SUBSTRING that has a performance ratio of $1 + \frac{1}{2r-1} + \epsilon$, for any $1 \leq r \leq m$ where $m = |\mathcal{F}|$, and $\epsilon > 0$.

While CLOSEST SUBSTRING minimizes the number of mismatches, MAX CLOSEST SUBSTRING maximizes the number of matches. We show that the MAX CLOSEST SUBSTRING problem cannot be approximated in polynomial time with ratio better than $(\log m)/4$, unless P=NP. As the maximization complement of the CLOSEST SUBSTRING problem, its reduction can also be applied to CLOSEST SUBSTRING. This application produces a similarly complementary result indicating the necessity of the $\frac{1}{O(m)}$ term in the PTAS [6]. While the hard ratio for CLOSEST SUBSTRING disappears asymptotically when m approaches infinity (as is to be expected given the PTAS [6]), it indicates a connection between the objective function and the number of strings given as input. This result supports the position that the term $\frac{1}{O(m)}$ in the PTAS performance ratio cannot be significantly improved by a polynomial time algorithm.

In [8], Sagot presents an exponential exact algorithm for the decision problem version of CLOSEST SUBSTRING, also known as COMMON APPROXIMATE SUBSTRING. Sagot also extends the problem to quorums, finding strings that are approximately present in at least a specified number of the input strings. This quorum size can be maximized as an alternate objective function, producing the MAXIMUM COVERAGE APPROXIMATE SUBSTRING problem. A restricted version of this problem was examined in [7], and erroneously claimed to be as hard to approximate as clique. We give a reduction from the MAXIMUM COVERAGE version of SET COVER, showing that the problem is hard to approximate within $e/(e-1)-\epsilon$ (where e is the base of the natural logarithm) for any $\epsilon > 0$.

The LONGEST COMMON APPROXIMATE SUBSTRING problem seeks to maximize the length of a center string that is within some specified distance d from every occurrence. We give a 2-approximation algorithm for this problem and show that 2 is optimal unless P=NP.

2 Preliminary Definitions

Definition 1. *Let x be an instance of optimization problem Π with optimal solution $opt(x)$. Let A be an algorithm solving Π, and $A(x)$ the solution value produced by A for x. The* performance ratio *of A with respect to x is*

$$\max\left\{\frac{A(x)}{opt(x)}, \frac{opt(x)}{A(x)}\right\} .$$

A is a ρ-approximation algorithm if and only if A always returns a solution with performance ratio less than or equal to ρ.

Definition 2. *Let Π and Π' be two minimization problems. A gap-preserving reduction (GP-reduction, $\leq_{GP}$) from Π to Π' with parameters (c,ρ),(c',ρ') is a polynomial-time algorithm f. For each instance I of Π, f produces an instance $I' = f(I)$ of Π'. The optima of I and I', say $opt(I)$ and $opt(I')$ respectively, satisfy the following properties:*

$$opt(I) \leq c \Rightarrow opt(I') \leq c' \ ,$$

$$opt(I) > c\rho \Rightarrow opt(I') > c'\rho',$$

where (c, ρ) *and* (c', ρ') *are functions of* $|I|$ *and* $|I'|$ *respectively, and* $\rho, \rho' > 1$.

Observe that the above definition of gap preserving reduction specifically refers to minimization problems, but can easily be adapted for maximization problems. Although it is implied by the name, *GP*-reductions do not require the size of the gap to be preserved, only that some gap remains [1].

We now formally specify the problems treated in this paper. All of these can be seen as variations on the CLOSEST SUBSTRING problem. Note that $d_H(x, y)$ represents the number of mismatches, or *Hamming distance*, between two strings x and y of equal length $|x| = |y|$.

MAX CLOSEST SUBSTRING

Instance: A set $\mathcal{F} = \{S_1, \dots, S_m\}$ of strings over alphabet Σ such that $\max_{1 \leq i \leq m} |S_i| = n$, integer l, $(1 \leq l \leq n)$.
Question: Maximize $\min_i(l - d_H(\mathcal{C}, s_i))$, such that $\mathcal{C} \in \Sigma^l$ and s_i is a substring of S_i, $(1 \leq i \leq m)$.

MAXIMUM COVERAGE APPROXIMATE SUBSTRING

Instance: A set $\mathcal{F} = \{S_1, \dots, S_m\}$ of strings over alphabet Σ such that $\max_{1 \leq i \leq m} |S_i| = n$, integers d and l, $(1 \leq d < l \leq n)$.
Question: Maximize $|\mathcal{F}'|$, $\mathcal{F}' \subseteq \mathcal{F}$, such that for some $\mathcal{C} \in \Sigma^l$ and for all $S_i \in \mathcal{F}'$, there exists a substring s_i of S_i such that $d_H(\mathcal{C}, s_i) \leq d$.

LONGEST COMMON APPROXIMATE SUBSTRING

Instance: A set $\mathcal{F} = \{S_1, \dots, S_m\}$ of strings over alphabet Σ such that $\max_{1 \leq i \leq m} |S_i| = n$, integer d, $(1 \leq d < n)$.
Question: Maximize $l = |\mathcal{C}|$, $\mathcal{C} \in \Sigma^*$, such that $d_H(\mathcal{C}, s_i) \leq d$ and s_i is a substring of S_i, $(1 \leq i \leq m)$.

Throughout this paper, when discussing different problems the values of d, l and m may refer to either the optimal values of objective functions or the values specified as part of the input. These symbols are used in accordance with their use in the formal statement of whatever problem is being discussed.

3 Max Closest Substring

3.1 Hardness of Approximating Max Closest Substring

In this section we use a gap preserving reduction from SET COVER to show inapproximability for MAX CLOSEST SUBSTRING. Lund and Yannakakis [2], with a reduction from LABEL COVER to SET COVER, showed that SET COVER could not be approximated in polynomial time with performance ratio better than

$(\log|\mathcal{B}|)/4$ (where $\mathcal{B}$ is the base set) unless NP = DTIME($2^{\text{poly}(\log n)}$). A result of Raz and Safra [3] indirectly strengthened the conjecture; SET COVER is now known to be NP-hard to approximate with ratio better than $(\log|\mathcal{B}|)/4$.

SET COVER

Instance: A set $\mathcal{B}$ of elements to be covered and a collection of sets $\mathcal{L}$ such that $\mathcal{L}_i \subseteq \mathcal{B}, (1 \leq i \leq |\mathcal{L}|)$.
Question: Minimize $|R|$, $R \subseteq \mathcal{L}$, such that $\cup_{j=1}^{|R|} R_j = \mathcal{B}$.

Let $I = \langle \mathcal{B}, \mathcal{L} \rangle$ be an instance of SET COVER. The reduction constructs, in polynomial time, a corresponding instance $I' = \langle \mathcal{F}, l \rangle$ of MAX CLOSEST SUBSTRING. For all $\rho > 1$, there exists a $\rho' > 1$ such that a solution for I with a ratio of ρ can be obtained in polynomial time from a solution to I' with ratio ρ'.

The Alphabet. The strings of $\mathcal{F}$ are composed of characters from the alphabet $\Sigma = \Sigma_1 \cup \Sigma_2$. The characters of Σ_1 are referred to as *set characters*, and identify sets in $\mathcal{L}$. The characters of Σ_2 are referred to as *element characters* and are in one-to-one correspondence with elements of the base set $\mathcal{B}$.

$$\Sigma_1 = \{p_i : 1 \leq i \leq |\mathcal{L}|\} ,$$
$$\Sigma_2 = \{u_i : 1 \leq i \leq |\mathcal{B}|\} .$$

Substring Gadgets. The strings of $\mathcal{F}$ are made up of two types of substring gadgets. We use the function f, defined below, to ensure that the substring gadgets are sufficiently large. The gadgets are defined as follows:

Subset Selectors: $\langle set(i) \rangle = p_i^{f(|\mathcal{B}|)}$
Separators: $\langle separator(j) \rangle = u_j^{f(|\mathcal{B}|)}$

The Reduction. The string set $\mathcal{F}$ contains $|\mathcal{B}|$ strings, corresponding to the elements of $\mathcal{B}$. For each $j \in \mathcal{B}$, let $L_j \subseteq \mathcal{L}$ be the subfamily of sets containing the element j. With product notation referring to concatenation, define the string

$$S_j = \prod_{q \in L_j} \langle set(q) \rangle \langle separator(j) \rangle .$$

The function $f : \mathbb{N} \mapsto \mathbb{N}$ must be defined. It is necessary for f to have the property that for all positive integers $x < |\mathcal{B}|$,

$$\left\lfloor \frac{f(|\mathcal{B}|)}{x} \right\rfloor > \left\lfloor \frac{f(|\mathcal{B}|)}{x+1} \right\rfloor .$$

It is straightforward to check that $f(y) = y^2$ has this property. The maximum length of any member of $\mathcal{F}$ is $n = 2|\mathcal{L}||\mathcal{B}|^2$, the size of $\mathcal{F}$ is $m = |\mathcal{B}|$, the length of the center is $l = f(|\mathcal{B}|) = |\mathcal{B}|^2$ and the alphabet size is $|\Sigma| = |\mathcal{L}| + |\mathcal{B}|$. We call any partition of $\mathcal{F}$ whose equivalence relation is the property of having an exact common substring a *substring induced partition.* For any two occurrences s, s' of a center, we call s and s' disjoint if for all $1 \leq q \leq |s|$, $s[q] \neq s'[q]$. Observe that the maximum distance to an optimal center, for any set of disjoint occurrences, increases with the size of the set.

Lemma 1. *Let F be a set of occurrences of an optimal center $\mathcal{C}$ such that $|F| = k$. If for each pair $s, s' \in F$, $d_H(s, s') = l$, then for every $s \in F$, $l - d_H(\mathcal{C}, s) \geq \lfloor l/k \rfloor$. Also, there is at least one $s \in \mathcal{F}$ such that $l - d_H(\mathcal{C}, s) = \lfloor l/k \rfloor$.*

Proof. There are l total positions and for any position p, there is a unique $s \in F$ such that $s[p] = \mathcal{C}[p]$. If some $s \in F$ had $l - d_H(\mathcal{C}, s) < \lfloor l/k \rfloor$, then the center $\mathcal{C}$ would not be optimal, as a better center can be constructed by taking position symbols evenly from the k occurrences. If all $s \in F$ have $l - d_h(\mathcal{C}, s) > \lfloor l/k \rfloor$, then the total number of matches exceeds l, some pair of matches would have the same position, and thus some pair $s, s' \in F$ have $d_H(s, s') < l$. □

The significance of our definition for f is apparent from the above proof. It is essential that, under the premise of Lemma 1, values of k (the number of distinct occurrences of a center) can be distinguished based on the maximum distance from any occurrence to the optimal center.

Lemma 2. SET COVER $\leq_{GP}$ MAX CLOSEST SUBSTRING.

Proof. Suppose the optimal cover R for $\langle \mathcal{B}, \mathcal{L} \rangle$ has size less than or equal to c. Construct string $\mathcal{C}$ of length $|\mathcal{B}|^2$ as follows. To the positions in $\mathcal{C}$, assign in equal amounts the set characters representing members of R. Then $\mathcal{C}$ is a center for $\mathcal{F}$ with maximum similarity $\lfloor |\mathcal{B}|^2/c \rfloor$.

Suppose $|R| > c$. Let $\mathcal{F}'$ be the largest subset of $\mathcal{F}$ having a substring induced c-partition. By the reduction, since $|R| > c$, $\mathcal{F}' \neq \mathcal{F}$. Let S be any string in $\mathcal{F} \backslash \mathcal{F}'$. By Lemma 1, any optimal center for $\mathcal{F}'$ must have minimum similarity $\lfloor |\mathcal{B}|^2/c \rfloor$, and therefore has at least $\lfloor |\mathcal{B}|^2/c \rfloor$ characters from a substring of every string in $\mathcal{F}'$. But the occurrence in S is disjoint from the occurrences in $\mathcal{F}'$, forcing the optimal center to match an equal number of positions in more than c disjoint occurrences. Hence, also by Lemma 1, the optimal center matches no more than $\lfloor |\mathcal{B}|^2/(c+1) \rfloor < \lfloor |\mathcal{B}|^2/c \rfloor$ characters in some occurrence. The gap preserving property of the reduction follows since $\lfloor |\mathcal{B}|^2/c \rfloor$ is a decreasing function of c. □

Theorem 1. MAX CLOSEST SUBSTRING *is not approximable within* $(\log m)/4$ *in polynomial time unless* P=NP.

Proof. The theorem follows from the fact that the NP-hard ratio for MAX CLOSEST SUBSTRING remains identical to that of the source problem SET COVER. □

As MAX CLOSEST SUBSTRING is the complementary maximization version of CLOSEST SUBSTRING, and there is a bijection between feasible solutions to the complementary problems that preserves the order of solution quality, this reduction also applies to CLOSEST SUBSTRING. The form of the hard performance ratio for CLOSEST SUBSTRING provides evidence that the two separate sources of error, $1/O(m)$ and ϵ, are necessary in the PTAS of [6].

Theorem 2. CLOSEST SUBSTRING *cannot be approximated with performance ratio* $1 + \frac{1}{\omega(m)}$ *in polynomial time unless* P=NP.

Proof. Since the NP-hard ratio for SET COVER is $\rho = (1/4)\log|\mathcal{B}|$, the NP-hard ratio obtained for CLOSEST SUBSTRING in the above reduction is

$$\begin{aligned}\rho' &= \tfrac{c\rho-1}{c\rho-\rho} \\ &= 1 + \left(\tfrac{\rho-1}{\rho}\right)\cdot\left(\tfrac{1}{c-1}\right) \\ &\geq 1 + \tfrac{1}{O(m)} \ .\end{aligned}$$

□

3.2 An Approximation Algorithm for Max Closest Substring

The preceding subsection showed that MAX CLOSEST SUBSTRING cannot be approximated within $(\log m)/4$. Here, we show that this bound is within a factor of $4 \cdot |\Sigma|$ of being tight, by presenting an approximation algorithm that achieves a bound of $|\Sigma| \log m$ for MAX CLOSEST SUBSTRING.

Due to the complementary relationship between MAX CLOSEST SUBSTRING and CLOSEST SUBSTRING, we start by presenting a greedy algorithm for CLOSEST STRING. The greedy nature of the algorithm is due to the fact that it commits to a local improvement at each iteration. The algorithm also uses a lazy strategy that bases each decision on information obtained by examining a restricted portion of the input. This is the most naive form of local search; the algorithm is not expected to perform well. The idea of the algorithm is to read the input strings column by column, and for each column i, assign a character to $\mathcal{C}[i]$ before looking at any column j such that $j > i$. Algorithm 1 describes this procedure, named GREEDYANDLAZY, in pseudocode.

Algorithm 1: Pseudocode for the GREEDYANDLAZY algorithm.
Input: A set of strings $\mathcal{F} = \{S_1, \ldots, S_m\}$.
Output: An $m(1 - |\Sigma|^{-1})$-approximate center $\mathcal{C}$ for $\mathcal{F}$.

```
GREEDYANDLAZY($\mathcal{F}$)
1.    $\mathcal{C} \leftarrow \lambda$
2.    for $i \leftarrow 1$ to $n$
3.        Let $F_i = \{S_1[1..i], \ldots, S_m[1..i]\}$
4.        for each $\alpha \in \Sigma$
5.            $X^\alpha \leftarrow \{S \in F_i : d_H(\mathcal{C}\alpha, S) = \max_{S' \in F_i} d_H(\mathcal{C}\alpha, S')\}$
6.        Let $\alpha = S_1[i]$
7.        for each $\beta \in \Sigma$
8.            if $|X^\beta| < |X^\alpha|$
9.                $\alpha = \beta$
10.       $\mathcal{C} \leftarrow \mathcal{C}\alpha$
11.   output($\mathcal{C}$)
```

Lemma 3. *The greedy and lazy algorithm for* CLOSEST STRING *produces a center string with radius within a factor of* $m(1 - \frac{1}{|\Sigma|})$ *of the optimal radius.*

Proof. Consider the number of iterations required to guarantee that each $S \in \mathcal{F}$ matches $\mathcal{C}$ in at least one position. Let J_i be the set of strings that do not match any position of $\mathcal{C}$ after the i^{th} iteration, then

$$J_{i+1} \leq \left(\frac{|\Sigma| - 1}{|\Sigma|}\right) J_i \leq \exp(-1/|\Sigma|) J_i \ .$$

This is because the algorithm always selects the column majority character of those strings in J_i. Let x be the number of iterations required before all members of $\mathcal{F}$ match $\mathcal{C}$ in at least one position. A bound on the value of x is given by the following inequality:

$$\frac{1}{m} > \exp\left(-\frac{x}{|\Sigma|}\right) \ .$$

Hence, for any strictly positive ϵ, after $x = |\Sigma| \ln m + \epsilon$ iterations, each member of $\mathcal{F}$ matches $\mathcal{C}$ in at least one position. After the final iteration, the total distance from $\mathcal{C}$ to any member of $\mathcal{F}$ is at most $n - n/(|\Sigma| \ln m)$. The optimal distance is at least n/m, otherwise some positions are identical in $\mathcal{F}$ (and thus should not be considered). Therefore the performance ratio of GREEDYANDLAZY is

$$\frac{n - n/(|\Sigma| \ln m)}{n/m} \leq m \left(1 - \frac{1}{|\Sigma|}\right) \ .$$

□

The running time of GREEDYANDLAZY, for m sequences of length n, is $O(|\Sigma| m n^2)$.

Now consider applying GREEDYANDLAZY to the MAX CLOSEST SUBSTRING problem by selecting an arbitrary set of substrings of length l to reduce the problem to a MAX CLOSEST STRING problem. The number of matches between any string in $\mathcal{F}$ and the constructed center will be at least $\Omega(l/(|\Sigma| \log m))$.

Corollary 1. GREEDYANDLAZY *is a* $O(|\Sigma| \log m)$*-approximation algorithm for* MAX CLOSEST SUBSTRING.

Since MAX CLOSEST SUBSTRING is hard to approximate with ratio better than $(\log m)/4$, this approximation algorithm is within $4 \cdot |\Sigma|$ of optimal.

4 Maximum Coverage Approximate Substring

The incorrect reduction given in [7] claimed an NP-hard ratio of $O(n^\epsilon)$, $\epsilon = \frac{1}{4}$, for MAXIMUM COVERAGE APPROXIMATE SUBSTRING when $l = n$ and $|\Sigma| = 2$. Its error resulted from applying Theorem 5 of [5], proven only for alphabet size at least three, to binary strings. Hardness of approximation for the general problem is shown here by a reduction from MAXIMUM COVERAGE.

MAXIMUM COVERAGE

Instance: A set $\mathcal{B}$ of elements to be covered and a collection of sets $\mathcal{L}$ such that $\mathcal{L}_i \subseteq \mathcal{B}, (1 \leq i \leq |\mathcal{L}|)$, a positive integer k.
Question: Maximize $|B|$, $B \subseteq \mathcal{B}$, such that $B = \cup_{j=1}^{k} \mathcal{L}_j$, where $\mathcal{L}_j \in \mathcal{L}$.

Given an instance $\langle \mathcal{B}, L, k \rangle$ of MAXIMUM COVERAGE, we construct an instance $\langle \mathcal{F}, l, d \rangle$ of MAXIMUM COVERAGE APPROXIMATE SUBSTRING where $m = |\mathcal{B}|$, $l = k$, $d = k - 1$ and $n \leq k|\mathcal{L}|$. The construction of $\mathcal{F}$ is similar to the construction used when reducing from SET COVER to CLOSEST SUBSTRING in Section 3; unnecessary parts are removed.

The Alphabet. The strings of $\mathcal{F}$ are composed of characters from the alphabet Σ. The characters of Σ correspond to the sets $\mathcal{L}_i \in \mathcal{L}$ that can be part of a cover, so $\Sigma = \{x_i : 1 \leq i \leq |\mathcal{L}|\}$.

The Reduction. The string set $\mathcal{F} = \{S_1, \ldots, S_{|\mathcal{B}|}\}$ will contain strings corresponding to the elements of $\mathcal{B}$. To construct these strings for each $j \in \mathcal{B}$, let $L_j \subseteq \mathcal{L}$ be the subfamily of sets containing the element j. For each $j \in \mathcal{B}$, define

$$S_j = \prod_{x_i \in L_j} x_i^k \ .$$

Set $d = k - 1$ and $l = k$. We seek to maximize the number of strings in $\mathcal{F}$ containing occurrences of some center $\mathcal{C}$.

Lemma 4. MAXIMUM COVERAGE $\leq_{GP}$ MAXIMUM COVERAGE APPROXIMATE SUBSTRING.

Proof. Suppose $\langle \mathcal{L}, \mathcal{B}, k \rangle$ is an instance of MAXIMUM COVERAGE with a solution set $R \subset \mathcal{L}$, such that $|R| = k$ and R covers $b \leq |\mathcal{B}|$ elements. Then there is a center $\mathcal{C}$ for $\mathcal{F}$ of length $l = k$ that has distance at most $d = k - 1$ from a substring of b strings in $\mathcal{F}$. Let the k positions in $\mathcal{C}$ be assigned characters representing the k sets in the cover, *i.e.* for each $x_i \in R$, there is a position p such that $\mathcal{C}[p] = x_i$. All b members of $\mathcal{F}$ corresponding to those covered elements in $\mathcal{B}$ contain a substring matching at least one character in $\mathcal{C}$, and mismatch at most $k - 1$ characters. Suppose one cannot obtain a k cover with ratio better than ρ. Then one cannot obtain a center for $\mathcal{F}$ that occurs in more than b/ρ strings of $\mathcal{F}$, so the hard ratio is $\rho' = \frac{b}{b/\rho} = \rho$. □

Theorem 3. MAXIMUM COVERAGE APPROXIMATE SUBSTRING *cannot be approximated with performance ratio* $e/(e-1) - \epsilon$*, for any* $\epsilon > 0$*, unless* P=NP.

Proof. It was shown in [4] that the NP-hard ratio for MAXIMUM COVERAGE is $e/(e-1) - \epsilon$. This result combined with Lemma 4 proves the theorem. □

Note that this reduction shows hardness for the general version of the problem, and leaves open the restricted case of $l = n$ with $|\Sigma| = 2$. No approximation algorithms with nontrivial ratios are known.

5 Longest Common Approximate Substring

The LONGEST COMMON APPROXIMATE SUBSTRING problem seeks to maximize the length of a center that is within a given distance from each string in the problem instance. That a feasible solution always exists can be seen by considering the case of a single character, since the problem is defined with $d > 0$. This problem is useful in finding seeds of high similarity for sequence comparisons.

Here we show that a simple algorithm always produces a valid center that is at least half the optimal length. A valid center is any string that has distance at most d from at least one substring of each string in $\mathcal{F}$. The algorithm simply evaluates each substring of members of $\mathcal{F}$ and tests them as centers. The following procedure EXTEND accomplishes this with a time complexity of $\Theta(m^2n^3)$.

Algorithm 2: Pseudocode for the EXTEND algorithm.
Input: A set of strings $\mathcal{F} = \{S_1, \ldots, S_m\}$ and an integer d.
Output: A valid center $\mathcal{C}$ for $\mathcal{F}$.

```
EXTEND($\mathcal{F}$, $d$)
1.    Let $\mathcal{C}$ be $\lambda$, the empty string
2.    for each string $S_i$ in $\mathcal{F}$
3.        for each substring $c$ of $S_i$
4.            if $c$ is a valid center for $\mathcal{F}$ and $|c| > |\mathcal{C}|$ then let $\mathcal{C}$ be $c$
5.    output($\mathcal{C}$)
```

Theorem 4. EXTEND *is a 2-approximation algorithm for* LONGEST COMMON APPROXIMATE SUBSTRING.

Proof. Let $\mathcal{C}$ be the optimal center for $\mathcal{F}$. For each $S_i \in \mathcal{F}$, let s_i be the occurrence of $\mathcal{C}$ from S_i; observe that $|s_i| = |\mathcal{C}|$. Define $s_{i,1}$ as the substring of s_i consisting of the first $|\mathcal{C}|/2$ positions of s_i, and $s_{i,2}$ as the substring consisting of the remaining positions. Similarly, define $\mathcal{C}_1$ and $\mathcal{C}_2$ as the first and last half of $\mathcal{C}$. For $x \in \{1, 2\}$, let c_x be equal to the string $s_{i,x}$ that satisfies

$$d_H(s_{i,x}, \mathcal{C}_x) \leq \min_{s_{j,x}, j \neq i} d_H(s_{j,x}, \mathcal{C}_x) \ .$$

Define c such that

$$c = \begin{cases} c_1 & \text{if } d_H(c_1, \mathcal{C}_1) \leq d_H(c_2, \mathcal{C}_2), \\ c_2 & \text{otherwise.} \end{cases}$$

Note that $d_H(c, \mathcal{C}_x) \leq d/2$, for some $x \in \{1, 2\}$. Suppose, for contradiction, that c is not a valid center. Assume, without loss of generality, that $c = s_{i,1}$ for some i. Then there is some $s_{i,1}$ such that $d_H(c, s_{i,1}) > d$. Since $d_H(c, \mathcal{C}_1) = d/2 - y$ for some $1 \leq y \leq d/2$, by the triangle inequality $d_H(s_{i,1}, \mathcal{C}_1) \geq d/2 + y + 1$. This implies that $d_H(s_{i,2}, \mathcal{C}_2) \leq d/2 - y - 1 < d_H(c, \mathcal{C}_1)$, contradicting the definition of c. Hence c is a valid center. Since c is a substring of one of the input strings, it will be found by EXTEND. It is half the length of the optimal length center $\mathcal{C}$, so a center will be found that is at least half the length of the longest center. □

The performance ratio of 2 is optimal unless P=NP. We use a transformation from the VERTEX COVER decision problem that introduces a gap in the objective function.

VERTEX COVER

Instance: A graph $G = (V, E)$ and a positive integer k.
Question: Does G have a vertex cover of size at most k, *i.e.*, a set of vertices $V' \subseteq V$, $|V'| \leq k$, such that for each edge $(u, v) \in E$, at least one of u and v belongs to V'?

Suppose for some graph G, we seek to determine if G contains a vertex cover of size k. We construct an instance of LONGEST COMMON APPROXIMATE SUBSTRING with $|E|$ strings corresponding to the edges of G. The intuition behind the reduction is that an occurrence of the center in each string corresponds to the occurrence of a cover vertex in the corresponding edge. Before giving values of n and d, we describe the gadgets used in the reduction.

The Alphabet. The string alphabet is $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \{A\}$. We refer to these as vertex characters (Σ_1), unique characters (Σ_2), and the alignment character (A), where $\Sigma_1 = \{v_i : 1 \leq i \leq |V|\}$ and $\Sigma_2 = \{u_{ij} : (i, j) \in E\}$.

Substring Gadgets. We next describe the two "high level" component substrings used in the construction. The function f is any arbitrarily large polynomial function of $|G|$.

Vertex Selectors: $\langle vertex(x, i, j, z)\rangle = A^{f(k)} u_{ij}^{(z-1)} v_x u_{ij}^{(k-z)} A^{f(k)}$
Separators: $\langle separator(i, j)\rangle = u_{ij}^{3f(k)}$

The Reduction. We construct $\mathcal{F}$ as follows. For any edge $(i, j) \in E$:

$$S_{ij} = \prod_{1 \leq z \leq k} \langle vertex(i, i, j, z)\rangle \langle separator(i, j)\rangle \langle vertex(j, i, j, z)\rangle \langle separator(i, j)\rangle$$

The length of each string is then $n = k(10f(k) + 2k)$. The threshold distance is $d = k - 1$.

Theorem 5. LONGEST COMMON APPROXIMATE SUBSTRING *cannot be approximated in polynomial time with performance ratio better than $2-\epsilon$, for any $\epsilon > 0$, unless* P=NP.

Proof. For any set of strings $\mathcal{F}$ so constructed, there is an exact common substring of length $f(k)$ corresponding to the $f(k)$ repeats of the alignment character A. Suppose there is a size k cover for the source instance of VERTEX COVER. Construct a center $\mathcal{C}$ for $\mathcal{F}$ as follows. Assign the alignment character A to the first $f(k)$ positions in $\mathcal{C}$. To positions $f(k)+1$ through $f(k)+k$, assign the characters corresponding to the vertices in the vertex cover. These may be assigned in any order. Finally, assign the alignment character A to the remaining $f(k)$ positions of $\mathcal{C}$. Each string in $\mathcal{F}$ contains a substring that matches $2f(k)+1$ positions in $\mathcal{C}$, so $\mathcal{C}$ is a valid center.

If there is no k cover for the source instance of VERTEX COVER, then for any length $f(k)+k$ string there will be some $S \in \mathcal{F}$ that mismatches k positions. As f can be any arbitrarily large polynomial function of k, the NP-hard performance ratio is

$$\frac{2f(k)+k}{f(k)+k} \geq 2-\epsilon \ ,$$

for any constant $\epsilon > 0$.

To show hardness for $2-\epsilon$, where ϵ is not a constant (it can be a function of l), consider that we can manipulate the hard ratio into the form

$$2 - \frac{k}{f(k)+k} \ .$$

Since l is the optimal length and $l = 2f(k)+k$, substitute $f(k) = l/2 - k/2$ in the performance ratio:

$$2 - \frac{k}{l/2 - k/2 + k} = 2 - \frac{2k}{l+k} \ .$$

Suppose we select $l = k^c$ during the reduction, where c is any arbitrarily large constant. Then we have shown a hard performance ratio of

$$2 - \frac{2l^{1/c}}{l + l^{1/c}} \geq 2 - \frac{2l^{1/c}}{l} = 2 - \frac{2}{l^{(c-1)/c}} = 2\left(1 - \frac{1}{l^{(c-1)/c}}\right) \ .$$

□

6 Conclusion

These results show that, unless P=NP, the MAX CLOSEST SUBSTRING, MAXIMUM COVERAGE APPROXIMATE SUBSTRING, and LONGEST COMMON APPROXIMATE SUBSTRING problems all have limitations on their approximability.

The relationships between the different objective functions produce an interesting interplay between the approximability of minimizing d with l fixed, maximizing l with d fixed, and maximizing their difference $l-d$. While this last variant, the MAX CLOSEST SUBSTRING problem, has a hard performance ratio directly related to the number of strings m, the two variants that fix one parameter and attempt to maximize the difference by optimizing the other parameter have lower ratios of approximability. It is NP-hard to approximate MAX CLOSEST SUBSTRING with a performance ratio better than $(\log m)/4$, and we have provided a $(|\Sigma|\log m)$-approximation. For LONGEST COMMON APPROXIMATE SUBSTRING, with d fixed, the length can be approximately maximized with a ratio of 2, and it is NP-hard to approximate for any smaller ratio. The best ratio of approximation is for CLOSEST SUBSTRING, where l is fixed and d is minimized; the PTAS of [6] achieves a ratio of $(1+\frac{1}{2r-1}+\epsilon)$, for any $1 \leq r \leq m$, and we have now shown that unless P=NP it cannot be approximated closer than $1+\frac{1}{O(m)}$.

For the quorum variant of CLOSEST SUBSTRING, where the number of strings covered is instead the objective function to be maximized, then it is NP-hard to obtain a performance ratio better than $e/(e-1)$. The restricted variant with $l=n$ and $|\Sigma|=2$ once thought to be proven hard by [7] is still open, without either hardness or a nontrivial approximation algorithm.

Our reductions use alphabets whose size will increase. The complexity of variants of these new problems where the alphabet size is treated as a constant is open, except as they relate to known results for constant alphabets [6,7].

References

1. Sanjeev Arora. *Probabilistic checking of proofs and the hardness of approximation problems.* PhD thesis, UC Berkeley, 1994.
2. Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5), 1994.
3. Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, 475–484, 1997.
4. Uriel Feige. A threshold of $\log n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.
5. J. K. Lanctot, M. Li, B. Ma, S. Wang, and L. Zhang. Distinguishing string selection problems. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, 633–642. ACM Press, 1999.
6. Ming Li, Bin Ma, and Lusheng Wang. On the closest string and substring problems. *Journal of the ACM*, 49(2):157–171, 2002.
7. Bin Ma. A polynomial time approximation scheme for the closest substring problem. In *Combinatorial Pattern Matching (CPM 2000)*, *Lecture Notes in Computer Science 1848*, 99–107. Springer, 2000.
8. Marie-France Sagot. Spelling approximate repeated or common motifs using a suffix tree. In *LATIN'98*, *Lecture Notes in Computer Science 1380*, 374–390. Springer, 1998.