The Greedy Algorithm for the Minimum Common String Partition Problem

Marek Chrobak* Petr Kolman†* Jiří Sgall‡

July 23, 2004

Abstract

In the Minimum Common String Partition problem (MCSP) we are given two strings on input, and we wish to partition them into the same collection of substrings, minimizing the number of the substrings in the partition. This problem is NP-hard, even for a special case, denoted 2-MCSP, where each letter occurs at most twice in each input string. We study a greedy algorithm for MCSP that at each step extracts a longest common substring from the given strings. We show that the approximation ratio of this algorithm is between $\Omega(n^{0.43})$ and $O(n^{0.69})$. In case of 2-MCSP, we show that the approximation ratio is equal to 3. For 4-MCSP, we give a lower bound of $\Omega(\log n)$.

1 Introduction

By a *partition* of a string A we mean a sequence $\mathcal{P} = (P_1, P_2, \ldots, P_m)$ of strings whose concatenation is equal to A, that is $P_1 P_2 \ldots P_m = A$. The strings P_i are called the *blocks* of $\mathcal{P}$. If $\mathcal{P}$ is a partition of A and $\mathcal{Q}$ is a partition of B, then the pair $\pi = \langle \mathcal{P}, \mathcal{Q} \rangle$ is called a *common partition* of A, B, if $\mathcal{Q}$ is a permutation of $\mathcal{P}$. For example, $\pi = \langle (ab, bccad, cab), (bccad, cab, ab) \rangle$ is a common partition of strings $A = abbccadcab$ and $B = bccadcabab$.

The *minimum common string partition* problem (MCSP) is defined as follows: given two strings A, B, find a common partition of A, B with the minimal number of blocks, or report that no common partition exists. By k-MCSP we denote the version of MCSP where each letter occurs at most k times in each input string.

The necessary and sufficient condition for A, B to have a common partition is that each letter has the same number of occurrences in A and B. Strings with this property are called *related.* Verifying whether two strings are related can be done easily in linear time, and for the rest of the paper we assume, without loss of generality, that the input strings are related. In particular, A and B have the same length, that we denote by n.

In this article, we study the greedy algorithm for MCSP that constructs a common partition by iteratively extracting the longest common substring of the input strings. More precisely, the algorithm can be described in pseudo-code as follows:

*Department of Computer Science, University of California, Riverside, CA 92521. marek@cs.ucr.edu. Supported by NSF grant CCR-0208856.

†Institute for Theoretical Computer Science, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic. kolman@kam.mff.cuni.cz. Supported by project LN00A056 of MŠMT ČR and NSF grants CCR-0208856 and ACI-0085910.

‡Mathematical Institute, AS CR, Žitná 25, CZ-11567 Praha 1, Czech Republic. sgall@math.cas.cz. Supported by Inst. for Theor. Comp. Sci., Prague (project LN00A056 of MŠMT ČR) and by grant IAA1019401 of GA AV ČR.

Algorithm GREEDY

Let A and B be two related input strings

while there are symbols in A or B outside marked blocks **do**

$S \leftarrow$ longest common substring of A, B that does not overlap previously marked blocks

mark one occurrence of S in each of A and B as blocks

$(\mathcal{P}, \mathcal{Q}) \leftarrow$ sequence of consecutive marked blocks in A and B, respectively

For example, if $A = cdabcdabceab$, $B = abceabcdabcd$, then GREEDY first marks substring $abcdabc$, then ab, and then three single-letter substrings c, d, e, so the resulting partition is

$$\langle (c, d, abcdabc, e, ab), (ab, c, e, abcdabc, d) \rangle,$$

while the optimal partition is $\langle (cdabcd, abceab), (abceab, cdabcd) \rangle$. As illustrated by the above example, the common partition computed by GREEDY is not necessarily optimal. The question we study is what is the approximation ratio of GREEDY on MCSP and its variants. We prove the following results:

Theorem 1.1 (a) *The approximation ratio of* GREEDY *for* MCSP *is between* $\Omega(n^{0.43})$ *and* $O(n^{0.69})$.
(b) *For* 4-MCSP*, the approximation ratio of* GREEDY *is at least* $\Omega(\log n)$.
(c) *For* 2-MCSP*, the approximation ratio of* GREEDY *is equal to* 3.

Our results extend to the *signed* variation of the minimum common partition problem, where each letter has a plus or minus sign associated with it (cf. [1, 3]). In the signed MCSP, a block P from A may be matched with a block Q from B if either $P = Q$ (including signs), or $P^R = Q$ where P^R denotes the *reversal* of P, defined as the block P in the reverse order and with all signs switched. As in MCSP, we want to find a minimum common partition of A and B, under the above restrictions.

Related work. The minimum common string partition problem was introduced by Chen *et al.* [1]. They pointed out that MCSP is closely related to the well-known problem of sorting by reversals and they use MCSP for comparison of two DNA sequences. In this application, the letters in the alphabet represent different genes in the DNA sequences, and the cardinality of the minimum common partition measures the similarity of these sequences. The restricted case of k-MCSP is of particular interest here. Goldstein *et al.* [3] proved that 2-MCSP is NP-hard.

The size of the minimum partition of A and B can be thought of as a distance between A and B. The classical edit-distance of two strings is defined as the smallest number of insertions, deletions, and substitutions required to convert one string into another [5]. Kruskal and Sankoff [4], and Tichy [8] were the first consider block operations in string comparison, in addition to the character operations. Lopresti and Tomkins [6] investigated several different distance measures; one of them is identical to the MCSP measure.

Shapira and Storer [7] study the problem of *edit distance with moves* in which the allowed string operations are the following: insert a character, delete a character, move a substring. They observe that if the input strings A, B are related, then the minimum number of the above listed operations needed to convert A into B is within a constant factor of the minimum number of only substring movements needed to convert A into B; and the latter quantity is within a constant factor of the minimum common partition size. Shapira and Storer also considered a greedy algorithm nearly identical to ours and claimed an $O(\log n)$ upper bound on its approximation ratio; as it turns out, however, their analysis is flawed.

Cormode and Muthukrishnan [2] describe an $O(\log n \log^* n)$-approximation algorithm for the problem of edit distance with moves. As explained above, this result yields an $O(\log n \log^* n)$-approximation for MCSP. Better bounds for MCSP are known for some special cases. A 1.5-approximation algorithm for 2-MCSP was given by Chen *et al.* [1]; a 1.1037-approximation algorithm for 2-MCSP and a 4-approximation algorithm for 3-MCSP were given by Goldstein *et al.* [3]. All these algorithms are considerably more complicated than GREEDY. Due to its simplicity and ease of implementation, GREEDY is a likely choice for solving MCSP in many practical situations, and thus its analysis is of its own independent interest.

2 Preliminaries

By $A = a_1 a_2 \dots a_n$ and $B = b_1 b_2 \dots b_n$ we denote the two arbitrary, but fixed, input strings of GREEDY. Without loss of generality, we assume that A and B are related. If π is a common partition of A, B, then we use notation $\#blocks(\pi)$ for the number of blocks in π, that we refer to as the *size* of π. The size of a minimum partition of A, B is denoted by $dist(A, B)$.

We typically deal with occurrences of letters in strings, rather than with letters themselves. By a "substring" we mean (unless stated otherwise) a specific occurrence of one string in another. Thus we identify a substring $S = a_p a_{p+1} \dots a_{p+s}$ of A with the set of indices $\{p, p+1, \dots, p+s\}$ and we write $S = \{p, p+1, \dots, p+s\}$, where $|S| = s+1$ is the *length* of S. Of course, the same convention applies to substrings of B. If S is a *common* substring of A, B, we use notations S^A and S^B to distinguish between the occurrences of S in A and B.

Partitions as functions. Suppose that we are given a bijection $\xi : [n] \to [n]$ (where $[n] = \{1, 2, \dots, n\}$) that *preserves letters* of A and B, that is, $b_{\xi(i)} = a_i$ for all $i \in [n]$. A pair of consecutive positions $i, i+1 \in [n]$ is called a *break* of ξ if $\xi(i+1) \neq \xi(i) + 1$. Let $\#breaks(\xi)$ denote the number of breaks in ξ. For a common substring S of A, B, say $S = a_p a_{p+1} \dots a_{p+s} = b_q b_{q+1} \dots b_{q+s}$, we say that ξ *respects* S if it maps consecutive letters of S^A onto consecutive letters in S^B, that is, $\xi(i) = i + q - p$ for $i \in S^A$.

A letter-preserving bijection ξ induces a common partition (also denoted ξ, for simplicity) whose blocks are the maximum length substrings of A that do not contain breaks of ξ. The partition obtained in this way does not have any "unnecessary" blocks, that is, $\#blocks(\xi) = \#breaks(\xi) + 1$. And vice versa, if $\pi = \langle \mathcal{P}, \mathcal{Q} \rangle$ is a common partition of A, B, we can think of π as a letter-preserving bijection $\pi : [n] \to [n]$ that respects each block of the partition. Obviously, we then have $\#blocks(\pi) \geq \#breaks(\pi) + 1$. We use this relationship throughout the paper, identifying common partitions with their corresponding bijections.

Reference partitions. Let π be a minimum common partition of A and B. (This partition may not be unique, but for all A, B, we choose one minimum common partition in some arbitrary way.) In the first step, GREEDY is guaranteed to find a substring S_1 of length at least the maximum length of a block in π. For the analysis of GREEDY, we would like to have a similar estimate for all later steps, too. However, already in the second step there is no guarantee that GREEDY finds a substring as long as the second longest block in π, since this block might overlap S_1 and it may be now partially marked (in A or B). To get a lower estimate on $|S_t|$, for $t > 1$, we introduce a corresponding *reference common partition* of A, B that respects all the blocks $S_1, \dots, S_{t-1}$ selected by GREEDY in steps 1 to $t-1$. This partition may gradually "deteriorate" (in comparison to the minimum partition of A and B), that is, it may include more blocks and its blocks may get shorter. Furthermore, it may not include a minimum common partition of the unmarked segments. Nevertheless, reference partitions provide a useful estimate on the "damage" caused by GREEDY when it makes wrong choices (that is, when it marks strings which are not in the optimum partition).

Denote by g the number of steps of GREEDY on A, B. For $t = 0, 1, \ldots, g$, the *reference common partition* ρ_t is defined inductively as follows. Initially, $\rho_0 = \pi$. Consider any $t = 1, \ldots, g$. Suppose that $S_t^A = \{p, p+1, \ldots, p+s\}$ and $S_t^B = \{q, q+1, \ldots, q+s\}$. Define function $\delta : S_t^A \to S_t^B$ such that $\delta(i) = i + q - p$ for $i \in S_t^A$. Then ρ_t is defined by

$$\rho_t(i) \quad = \quad \begin{cases} \delta(i) & \text{for } i \in S_t^A \\ \rho_{t-1}(\delta^{-1}\rho_{t-1})^{\ell(i)}(i) & \text{for } i \in [n] - S_t^A \end{cases} \tag{1}$$

where $\ell(i) = \min\left\{\lambda \geq 0 : \rho_{t-1}(\delta^{-1}\rho_{t-1})^{\lambda}(i) \notin S_t^B\right\}$. We show that each ρ_t is well-defined and we also bound the increase of the number of breaks from ρ_{t-1} to ρ_t:

Lemma 2.1 *For each $t = 0, 1, \ldots, g$,* (a) *ρ_t is a common partition of A, B,* (b) *ρ_t respects $S_1, \ldots, S_t$, and* (c) *if $t > 0$ then $\#breaks(\rho_t) \leq \#breaks(\rho_{t-1}) + 4$.*

Proof: The proof of the lemma is by induction. For $t = 0$, (a) and (b) are trivially true. Suppose that $t > 0$ and that the lemma holds for $t - 1$. To simplify notation let $S = S_t$, $\rho = \rho_{t-1}$ and $\rho' = \rho_t$.

Consider a bipartite graph $G \subseteq [n] \times [n]$, with edges $(i, \rho(i))$, for $i \in [n]$, and $(i, \delta(i))$, for $i \in S^A$. These two types of edges are called *ρ-edges* and *δ-edges*, respectively.

Let $\bar{S}^A = [n] - S^A$ and $\bar{S}^B = [n] - S^B$. In this proof, to avoid introducing additional notation, we think of S^A and $\bar{S}^A$ as the sets of nodes on the "left-hand" side of G and S^B and $\bar{S}^B$ as the nodes on the "right-hand" side. Then, any node in $\bar{S}^A$ or $\bar{S}^B$ is incident to one ρ-edge, and each node in S^A or S^B is incident to one ρ-edge and one δ-edge. Thus, G is a collection of vertex disjoint paths and cycles whose edges alternate between ρ-edges and δ-edges. We call them *G-paths* and *G-cycles*. All G-cycles have even length and contain only nodes from S^A and S^B. All maximal G-paths have odd lengths, start in $\bar{S}^A$, end in $\bar{S}^B$, and their interior vertices are in S^A or S^B. The G-path starting at $i \in \bar{S}^A$ has the form

$$i, \rho(i), \delta^{-1}\rho(i), \rho\delta^{-1}\rho(i), \ldots, \rho(\delta^{-1}\rho)^{\ell(i)}(i).$$

Thus, for $i \in \bar{S}^A$, $\rho'(i)$ is simply the other endpoint of the G-path that starts at i. This implies that ρ' is 1-1 and letter-preserving, so it is indeed a common partition. Condition (b) follows immediately from the inductive assumption and the definition of ρ'. It remains to prove (c).

Lemma 2.2 *Suppose that $i, i+1$ is a break of ρ'. Then one of the following conditions holds:*
(B0) *Exactly one of $i, i+1$ is in S^A.*
(B1) *$i, i+1 \in \bar{S}^A$ and there is $\lambda \leq \min\{\ell(i), \ell(i+1)\}$ such that $(\delta^{-1}\rho)^{\lambda}(i)$, $(\delta^{-1}\rho)^{\lambda}(i+1)$ is a break of ρ.*
(B2) *$i, i+1 \in \bar{S}^A$ and there is $\lambda \leq \min\{\ell(i), \ell(i+1)\}$ such that exactly one of $\rho(\delta^{-1}\rho)^{\lambda}(i)$, $\rho(\delta^{-1}\rho)^{\lambda}(i+1)$ belongs to S^B.*

We refer to breaks of types (B0), (B1), (B2), respectively, as breaks induced by the endpoints of S^A, breaks induced by the breaks inside S^A (only if i,$i+1$ is a new break), and breaks induced by the endpoints of S^B.

Proof: If exactly one of i, $i+1$ is in S^A, the case (B0) holds. Since i, $i+1$ is never a break in ρ' if both i and $i+1$ are in S^A, we assume that $i, i+1 \in \bar{S}^A$ for the rest of the proof.

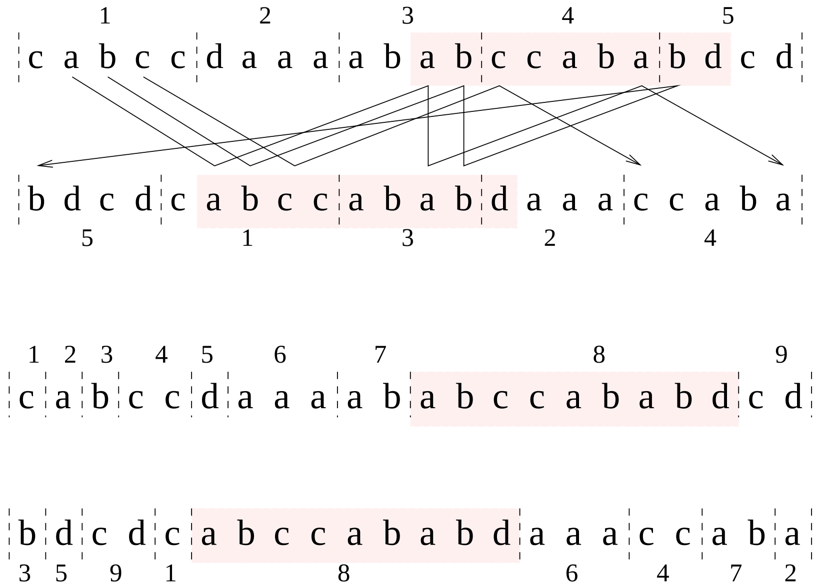

Figure 1: An example illustrating the construction of ρ'. The upper part shows ρ and some G-paths. The lower part shows ρ'. The strings in the partitions are numbered, and the common substring $S_t = abccababd$ is shaded.

Consider the largest integer $\lambda \leq \min\{\ell(i), \ell(i+1)\}$ for which $(\delta^{-1}\rho)^\lambda(i)$, $(\delta^{-1}\rho)^\lambda(i+1)$ are consecutive in S^A, that is $(\delta^{-1}\rho)^\lambda(i+1) = (\delta^{-1}\rho)^\lambda(i) + 1$. (We remark that it is not necessarily true that $(\delta^{-1}\rho)^h(i+1) = (\delta^{-1}\rho)^h(i) + 1$ for $h = 1, \ldots, \lambda$; these indices may diverge and then meet later, any number of times.) Let $j = (\delta^{-1}\rho)^\lambda(i)$. We have two sub-cases. If $\rho(j+1) \neq \rho(j)+1$, then $j, j+1$ is a break of ρ, so the condition (B1) is satisfied. If $\rho(j+1) = \rho(j)+1$, then at least one of $\rho(j)$, $\rho(j+1)$ must be in S^B, for otherwise $i, i+1$ would not be a break of ρ'. But we also cannot have both $\rho(j), \rho(j+1) \in S^B$, since then $(\delta^{-1}\rho)^{\lambda+1}(i)$, $(\delta^{-1}\rho)^{\lambda+1}(i+1)$ would be consecutive in S^A, violating the choice of λ. Therefore the case (B2) holds. $\square$

We now complete the proof of part (c) of Lemma 2.1. There are no breaks of ρ' inside S^A, and we have at most two breaks of type (B0) corresponding to the endpoints of S^A. By the disjointness of G-paths and cycles, there are at most two breaks of ρ' of type (B2), each corresponding to one endpoint of S^B. Similarly, each break of ρ (inside or outside S^A) induces at most one break of ρ' of type (B1). This implies (c), and the proof of the lemma is complete. $\square$

Note that we did not use the fact that S has maximum length. So our construction of ρ_t can be used to convert any common partition π into another partition π' that respects a given common substring S, and has at most four more breaks than π.

Lemma 2.1 implies that in every step t of the algorithm, every block in the reference partition ρ_t is either completely marked or completely unmarked.

3 Upper Bound for MCSP

In this section we show that GREEDY's approximation ratio is $O(n^{0.69})$. The proof uses reference common partitions introduced in Section 2 to keep track of the length of the common substrings selected by GREEDY.

For $p \geq q \geq 1$, we define $H(p,q)$ to be the smallest number h with the following property: for any input strings A, B, if at some step t of GREEDY there are at most p unmarked symbols in A and at most q unmarked blocks in the current reference partition ρ_t, then GREEDY makes at most h more steps until it stops (so its final common partition has at most $t+h$ blocks.) For convenience, we allow non-integral p and q in the definition. Note that $H(p,q)$ is non-decreasing in both variables.

Before proving the $O(n^{0.69})$ upper bound, we sketch a slightly weaker but simpler bound of $O(n^{0.75})$. Lemma 2.1 immediately gives a recurrence $H(p,q) \leq H(p(1-1/q), q+3)+1$ (whenever both values of H are defined), as in one step of GREEDY, the longest common substring has at least $\frac{p}{q}$ letters (which will be marked in the next partition), and the number of unmarked blocks in the reference partition increases by at most 3. We prove by induction on p that for $p \geq q$ and a sufficiently large constant C, we have $H(p,q) \leq Cp^{\frac{3}{4}}q^{\frac{1}{4}} - \frac{1}{3}q$. For $q=1$ this is trivial, as GREEDY finds the single unmarked block. We choose C such that for all $q \leq p < 5q$, the right-hand side is at least p, which is a trivial upper bound on $H(p,q)$. For $p \geq 5q \geq 10$, we have $p(1-1/q) \geq q+3$, thus we can use the inductive assumption and the recurrence to obtain

$$\begin{aligned} H(p,q) &\leq H(p(1-1/q), q+3)+1 \\ &\leq Cp^{\frac{3}{4}}(1-1/q)^{\frac{3}{4}}(q+3)^{\frac{1}{4}} - \tfrac{1}{3}(q+3)+1 \\ &\leq Cp^{\frac{3}{4}}q^{\frac{1}{4}} - \tfrac{1}{3}q. \end{aligned}$$

The last inequality follows from $(q-1)^{\frac{3}{4}}(q+3)^{\frac{1}{4}} \leq (q-1)^{\frac{1}{2}}[(q-1)^{\frac{1}{4}}(q+3)^{\frac{1}{4}}] \leq (q-1)^{\frac{1}{2}}(q+1)^{\frac{1}{2}} \leq q$. This completes the induction step and the proof. The bound we proved implies that $H(p,q) \leq O(p^{\frac{3}{4}})q$. Thus, if the input of GREEDY consists of two strings A, B of length n, the number of blocks in GREEDY's partition is at most $H(n, \mathit{dist}(A,B)) = O(n^{0.75})\,\mathit{dist}(A,B)$.

The idea of the proof of the improved bound is to consider, instead of one step of GREEDY, a number of steps proportional to the number of blocks in the original optimal partition, and show that during these steps GREEDY marks a constant fraction of the input string. This yields an improved recurrence for $H(p,q)$.

Lemma 3.1 *For all p, q satisfying $p \geq 9q/5+3$, we have $H(p,q) \leq H(5p/6, (3q+5)/2) + (q+5)/6$.*

Proof: Consider a computation of GREEDY on A, B, where, after some step t (i.e., with t blocks having already been marked), there are p unmarked symbols in A, and q unmarked reference blocks of ρ_t. We denote these blocks by $R_1, R_2, \ldots, R_q$, in the order of non-increasing length, that is $|R_z| \geq |R_{z+1}|$, for $z = 1, \ldots, q-1$. We analyze the computation of GREEDY starting at step $t+1$. Let g be the number of additional steps that GREEDY makes. Our goal is to show that

$$g \leq H(\tfrac{5}{6}p, \tfrac{3q+5}{2}) + \tfrac{q+5}{6} \tag{2}$$

(Since the bound is monotone in p and q, we do not need to consider the case of fewer than q unmarked blocks or fewer than p unmarked symbols.) If $g \leq (q+5)/6$, inequality (2) trivially holds, so in the rest of the proof we assume that $g > (q+5)/6$.

Let $T_i = S_{t+i}$ be the common substring selected by GREEDY in step $t+i$. We say that GREEDY *hits* R_z in step $t+i$ if T_i overlaps R_z, either in A or in B, that is, if either $T_i^A \cap R_z^A \neq \emptyset$ or $T_i^B \cap R_z^B \neq \emptyset$.

Claim A: For all $j = 1, \ldots, g$, the total length of those blocks $R_1, \ldots, R_q$ that are hit by GREEDY in A in steps $t+1, \ldots, t+j$ is at most $6\sum_{i=1}^{j}|T_i|$.

Proof: We estimate the total length of the blocks R_z that are hit at step $t+i$ in A but have not been hit in steps $t+1, \ldots, t+i-1$. The total length of the blocks that are contained in S_i^A and S_i^B is at most $2|T_i|$. There are up to four blocks that are hit partially, but by the greedy choice of T_i, each has length at most $|T_i|$, and the claim follows. □

Claim B: $6\sum_{i=1}^{\lfloor (q+5)/6 \rfloor}|T_i| \geq p$.

Proof: Let l be the minimum integer such that $6\sum_{i=1}^{l}|T_i| \geq p$. Since $\sum_{i=1}^{g}|T_i| = p$, l is well defined and $l \leq g$. For $j = 1, \ldots, l$, define χ_j as the maximal index for which $\sum_{x=1}^{\chi_j}|R_x| \leq 6\sum_{i=1}^{j-1}|T_i|$. Since $6\sum_{i=1}^{l-1}|T_i| < p = \sum_{x=1}^{q}|R_x|$, all χ_j are well defined, and $\chi_l < q$. We also note that $\chi_1 = 0$. For each $j = 1, \ldots, l$, Claim A implies that one of the blocks $R_1, \ldots, R_{\chi_j+1}$ is not hit by any of the blocks $T_1, \ldots, T_{j-1}$ and thus, by the definition of GREEDY and the ordering of the blocks R_z, $|T_j| \geq |R_{\chi_j+1}|$. Considering again the ordering of the blocks R_z, we have $6|T_j| \geq |R_{\chi_j+1}| + ... + |R_{\chi_j+6}|$. We conclude that $\chi_{j+1} \geq \chi_j + 6$, for $j = 1, \ldots, l-1$. This, in turn, implies that $q \geq \chi_l + 1 \geq 6l - 5$. Therefore $l \leq (q+5)/6$, and Claim B follows, by the choice of l and its integrality. □

By Claim B, after exactly $\lfloor (q+5)/6 \rfloor$ steps, GREEDY marks at least $p/6$ letters, so the number of remaining unmarked letters is at most $p' = 5p/6$. By Lemma 2.1, the number of unmarked blocks increases by at most 3 in each step (since one new block is marked), so the number of unmarked blocks induced by GREEDY in these $\lfloor (q+5)/6 \rfloor$ steps is at most $3\lfloor (q+5)/6 \rfloor \leq (q+5)/2$. Thus the total number of unmarked blocks after these steps is at most $q' = q + (q+5)/2 = (3q+5)/2$. The condition in the lemma guarantees that $H(p', q')$ is defined, so, by induction, the total number of steps is at most $H(p', q') + (q+5)/6$. This completes the proof if inequality (2) and the lemma. □

Finally, we prove the upper bound in Theorem 1.1(a).

Theorem 3.2 GREEDY *is an $O(n^\gamma)$-approximation algorithm for* MCSP, *where* $\gamma = \log\frac{3}{2}/\log\frac{9}{5} \approx 0.69$.

Proof: We prove by induction on p that for $p \geq q$ and a sufficiently large constant C,

$$H(p,q) \leq Cp^\gamma(q+5)^{1-\gamma} - \tfrac{1}{3}q.$$

We choose C so that for all $q \leq p < 9q/5 + 3$, the right-hand side is at least p and thus the inequality is valid. For $p \geq 9q/5 + 3$, by Lemma 3.1, the inductive assumption, and the choice of γ, we have

$$\begin{aligned} H(p,q) &\leq H(\tfrac{5}{6}p, \tfrac{3q+5}{2}) + \tfrac{q+5}{6} \\ &\leq C(\tfrac{5}{6}p)^\gamma(\tfrac{3}{2}(q+5))^{1-\gamma} - \tfrac{1}{3}\cdot\tfrac{3q+5}{2} + \tfrac{q+5}{6} \\ &= Cp^\gamma(q+5)^{1-\gamma} - \tfrac{1}{3}q. \end{aligned}$$

Let A, B be input strings of length n and with $dist(A,B) = m$. Then the number of blocks in GREEDY's partition is at most $H(n,m) = O(n^\gamma)m$, and the theorem follows. □

4 Lower Bound for MCSP

We show that the approximation ratio of GREEDY is $\Omega(n^{1/\log_2 5}) = \Omega(n^{0.43})$. We first construct strings C_i, D_i, E_i, F_i as follows. Initially, $C_0 = a$ and $D_0 = b$. Suppose we already have C_i and D_i, and let Σ_i be the set of letters used in C_i, D_i. Define a new alphabet Σ'_i that has a new letter, say a', for each $a \in \Sigma$. We first create strings E_i and F_i by replacing all letters $a \in \Sigma$ in C_i and D_i, respectively, by their corresponding letters $a' \in \Sigma'_i$. Then, let

$$C_{i+1} = C_i D_i E_i D_i C_i, \qquad \text{and} \qquad D_{i+1} = D_i E_i F_i E_i D_i.$$

For each i, we consider the instance of strings $A_i = C_i D_i$ and $B_i = D_i C_i$. For example, $E_0 = c$, $F_0 = d$, $A_0 = ab$, $B_0 = ba$, $C_1 = abcba$, $D_1 = bcdcb$, $A_1 = abcbabcdcb$, and $B_1 = bcdcbabcba$, etc.

Let $n = 2 \cdot 5^i$. We have $|A_i| = |B_i| = n$ and $dist(A_i, B_i) \le 2$. We claim that GREEDY's common partition of A_i and B_i has $2^{i+2} - 2 = \Omega(n^{1/\log_2 5})$ substrings. We assume here that GREEDY does not specify how the ties are broken, that is, whenever a longest substring can be chosen in two or more different ways, we can decide which choice GREEDY makes.

The proof is by induction. For $i = 0$, GREEDY produces two substrings, as claimed. For $i \ge 0$,

$$A_{i+1} = C_i\, D_i\, E_i\, D_i\, C_i\, D_i\, E_i\, F_i\, E_i\, D_i, \qquad B_{i+1} = D_i\, E_i\, F_i\, E_i\, D_i\, C_i\, D_i\, E_i\, D_i\, C_i.$$

There are three common substrings of length 5^{i+1}: $C_iD_iE_iD_iC_i$, $D_iE_iF_iE_iD_i$, and $E_iD_iC_iD_iE_i$, and no longer common substrings exist. To justify this, we use the fact that the alphabet of C_i, D_i is disjoint from the alphabet of E_i, F_i. Suppose that S is a common substring of length at least 5^{i+1}. To have this length, S must contain either the first or the second E_i from A_{i+1}. We now have some cases depending on which E_i is contained in S, and where it is mapped into B_{i+1} via the occurrence of S in B_{i+1}. If S contains the first E_i, then, by the assumption about the alphabets, this E_i must be mapped into either $E_iF_iE_i$ or into the last E_i in B_{i+1}. If it is mapped into $E_iF_iE_i$, then S must be $E_iD_iC_iD_iE_i$. If it is mapped into the last E_i in B_{i+1}, then S must be $C_iD_iE_iD_iC_i$. In the last case, S contains the second E_i in A_{i+1}. By the same considerations as in the first case, it is easy to show that then S must be either $D_iE_iF_iE_iD_i$ or $E_iD_iC_iD_iE_i$.

Breaking the tie, assume that GREEDY marks substring $E_iD_iC_iD_iE_i$. The modified strings are:

$$C_i\, D_i\, \overline{E_i\, D_i\, C_i\, D_i\, E_i}\, F_i\, E_i\, D_i, \qquad D_i\, E_i\, F_i\, \overline{E_i\, D_i\, C_i\, D_i\, E_i}\, D_i\, C_i,$$

where the overline indicates the marked substring. In the first string the unmarked segments are A_i, A'_iD_i, and in the second string the unmarked segments are B_i and $D_iB'_i$, where $A'_i = F_iE_i$ and $B'_i = E_iF_i$ are identical as A_i, B_i respectively, but with the letters renamed. The argument in the previous paragraph and the disjointness of the alphabets implies that the maximum length of a non-marked common substring is 5^i. We break the tie again, and make GREEDY match the two D_i's in $F_iE_iD_i$ and $D_iE_iF_i$, and the resulting strings have the form

$$A_i\, \overline{E_i\, D_i\, C_i\, D_i\, E_i}\, A'_i\, \overline{D_i}, \qquad \overline{D_i}\, B'_i\, \overline{E_i\, D_i\, C_i\, D_i\, E_i}\, B_i.$$

Now, we have two non-marked pairs of substrings $\{A_i, B_i\}$ and $\{A'_i, B'_i\}$. These two pairs of strings have disjoint alphabets and will be processed by GREEDY independently of each other. By induction, GREEDY produces $2^{i+2} - 2$ substrings from A_i, B_i, and the same number from A'_i and B'_i. So we get the total of $2(2^{i+2} - 2) + 2 = 2^{i+3} - 2$ strings.

5 Lower Bound for GREEDY on 4-MCSP

In this section we show that GREEDY's approximation ratio is $\Omega(\log n)$ even on 4-MCSP instances. To simplify the description, we allow the input instances $\mathcal{A}, \mathcal{B}$ to be *multisets* of equal number of strings, rather than single strings. It is quite easy to see that this does not significantly affect the performance of GREEDY, for we can always replace $\mathcal{A}, \mathcal{B}$ by two strings A, B, as follows: If $\mathcal{A} = \{A_1, \ldots, A_m\}$ and $\mathcal{B} = \{B_1, \ldots, B_m\}$, let $A = A_1 x_1 y_1 A_2 x_2 y_2 \ldots A_{m-1} x_{m-1} y_{m-1} A_m$ and $B = B_1 y_1 x_1 B_2 y_2 x_2 \ldots B_{m-1} y_{m-1} x_{m-1} B_m$, where $x_1, y_1, \ldots, x_{m-1}, y_{m-1}$ are new letters. Then both the optimal partition and the partition produced by GREEDY on A, B are the same as on $\mathcal{A}, \mathcal{B}$, except for the singletons $x_1, y_1, \ldots, x_{m-1}, y_{m-1}$. Since in our construction m is a constant, it is sufficient to show a lower bound of $\Omega(\log n)$ for multisets of m strings.

For $i = 1, 2, \ldots$, we fix strings q_i, q'_i, r_i, r'_i that we will refer to as *elementary strings.* Each elementary string q_i, q'_i, r_i, r'_i has length 3^{i-1} and consists of 3^{i-1} distinct and unique letters (that do not appear in any other elementary string.)

We recursively construct instances $\mathcal{A}^i$, $\mathcal{B}^i$ of 4-MCSP. The invariant of the construction is that $\mathcal{A}^i$, $\mathcal{B}^i$ have the form:

$$\begin{array}{lllllllll} \mathcal{A}^i\colon & P_1 q_i, & P_2 q_i r_i, & P_3 q_i, & P_4 q_i r'_i, & P_5 q'_i, & P_6 q'_i r_i, & P_7 q'_i, & P_8 q'_i r'_i \\ \mathcal{B}^i\colon & P_1 q_i r_i, & P_2 q_i, & P_3 q_i r'_i, & P_4 q_i, & P_5 q'_i r_i, & P_6 q'_i, & P_7 q'_i r'_i, & P_8 q'_i \end{array}$$

where $P_1, \ldots, P_8$ are some strings of length smaller than 3^{i-1} with letters distinct from q_i, q'_i, r_i, r'_i.

Initially, we set all $P_1, \ldots, P_8 = \epsilon$, and construct $\mathcal{A}^1$, $\mathcal{B}^1$ as described above. In this case q_i, q'_i, r_i, r'_i are unique, single letters.

To construct $\mathcal{A}^{i+1}$, $\mathcal{B}^{i+1}$, we append pairs of elementary strings to the strings from $\mathcal{A}^i$, $\mathcal{B}^i$. For convenience, we omit the subscripts for elementary substrings, writing $q = q_i$, $\bar{q} = q_{i+1}$, etc. After rearranging the strings, the new instance is

$$\begin{array}{lllllllll} \mathcal{A}^{i+1}\colon & P_1 qr\ \bar{q}, & P_4 qr'\ \bar{q}\bar{r}, & P_7 q'r'\ \bar{q}, & P_6 q'r\ \bar{q}\bar{r}', & P_3 qr'\ \bar{q}', & P_2 qr\ \bar{q}'\bar{r}, & P_5 q'r\ \bar{q}', & P_8 q'r'\ \bar{q}'\bar{r}' \\ \mathcal{B}^{i+1}\colon & P_1 qr\ \bar{q}\bar{r}, & P_4 qr'\ \bar{q}, & P_7 q'r'\ \bar{q}\bar{r}', & P_6 q'r\ \bar{q}, & P_3 qr'\ \bar{q}'\bar{r}, & P_2 qr\ \bar{q}', & P_5 q'r\ \bar{q}'\bar{r}', & P_8 q'r'\ \bar{q}' \end{array}$$

Note that this instance has the same structure as the previous one, since we can take $P'_1 = P_1 qr$, $P'_2 = P_4 qr'$, etc.. Thus we can continue this construction recursively. Each letter appears at most four times in $\mathcal{A}^i$ and $\mathcal{B}^i$, so this is indeed an instance of 4-MCSP; the claimed bound on the length of the P'_j's also follows easily.

Consider the i-th instance, $\mathcal{A}^i$ and $\mathcal{B}^i$. To estimate the optimal partition, we match the 8 pairs of strings as aligned above, adding the shorter string from each pair to the common partition. There are only 4 additional strings left, namely $\bar{r}$, $\bar{r}'$, $\bar{r}$, $\bar{r}'$, implying $dist(\mathcal{A}^i, \mathcal{B}^i) \leq 12$.

We show that GREEDY computes a partition with $\Theta(i) = \Theta(\log n)$ blocks. To this end, we claim that, starting from $\mathcal{A}^{i+1}, \mathcal{B}^{i+1}$, GREEDY first matches all suffixes that consist of two elementary strings as shown below ($\mathcal{A}^{i+1}$ and $\mathcal{B}^{i+1}$ are rearranged to show the matched strings aligned vertically):

$$\begin{array}{lllllllll} \mathcal{A}^{i+1}\colon & P_4 qr'\ \bar{q}\bar{r}, & P_6 q'r\ \bar{q}\bar{r}', & P_2 qr\ \bar{q}'\bar{r}, & P_8 q'r'\ \bar{q}'\bar{r}', & P_1 q\ r\bar{q}, & P_7 q'\ r'\bar{q}, & P_3 q\ r'\bar{q}', & P_5 q'\ r\bar{q}' \\ \mathcal{B}^{i+1}\colon & P_1 qr\ \bar{q}\bar{r}, & P_7 q'r'\ \bar{q}\bar{r}', & P_3 qr'\ \bar{q}'\bar{r}, & P_5 q'r\ \bar{q}'\bar{r}', & P_6 q'\ r\bar{q}, & P_4 q\ r'\bar{q}, & P_8 q'\ r'\bar{q}', & P_2 q\ r\bar{q}' \end{array}$$

Indeed, the instance has four common substrings of length $2 \cdot 3^i$, namely $\bar{q}\bar{r}$, $\bar{q}\bar{r}'$, $\bar{q}'\bar{r}$, $\bar{q}'\bar{r}'$, and, by the choices of the lengths of elementary strings and the bound on the lengths of the P_j's, all other common substrings are shorter. Thus, GREEDY starts by removing (marking) these four suffixes. Similarly, at this step, the new instance will have four common substrings of length $3^i + 3^{i-1}$, namely $r\bar{q}$, $r'\bar{q}$, $r'\bar{q}'$, $r\bar{q}'$, and all other common substrings are shorter. GREEDY will remove these four suffixes. The resulting instance is simply $\mathcal{A}^i$, $\mathcal{B}^i$ and we can continue recursively, getting $\Theta(i)$ blocks. If n is the length (total number of characters) of $\mathcal{A}^i$, we have $i = \Theta(\log n)$ and the proof of the lower bound is complete.

6 Upper Bound for GREEDY on 2-MCSP

In this section we prove that on 2-MCSP instances GREEDY's approximation ratio is at most 3. In the next section we will give a matching lower bound.

Consider two arbitrary, but fixed, related strings $A = a_1 a_2 \cdots a_n$ and $B = b_1 b_2 \cdots b_n$ in which each letter appears at most twice. Let π be a minimum common partition of A, B, and denote by g the number of steps of GREEDY on A, B. For each $t = 0, \ldots, g$, let S_t be the block marked by GREEDY in step t, and let ρ_t be the common reference partition of A, B at step t, as defined in Section 2. In particular, $\rho_0 = \pi$, and ρ_g is the is the final partition computed by GREEDY.

Our proof is based on amortized analysis. We show how to define a *potential* Φ_t of ρ_t that has the following three properties:

(P1) $\Phi_0 \leq 3 \cdot \#blocks(\rho_0) + 1$,

(P2) $\Phi_t \leq \Phi_{t-1}$ for $t = 1, \ldots, g$, and

(P3) $\Phi_g \geq \#blocks(\rho_g) + 1$.

If such Φ_t's exist, then, using the optimality of ρ_0 and conditions (P1), (P2), (P3), we obtain $\#blocks(\rho_g) \leq \Phi_g - 1 \leq \Phi_0 - 1 \leq 3 \cdot \#blocks(\rho_0) = 3 \cdot \#blocks(\pi) = 3 \cdot dist(A, B)$, and the 3-approximation of GREEDY follows immediately. It remains to define the potential and show that it has the desired properties.

Classification of breaks. Consider some step t. A break $i, i+1$ of ρ_t is called *original* if it is also a break of π; otherwise we call this break *induced.* Letters inside blocks marked by GREEDY are called *marked.* For any letter a_i in A, we say that a_i is *unique* in ρ_t if a_i is not marked, and there is no other non-marked appearance of a_i in A.

Suppose that $i, i+1$ is an original break in ρ_t. We say that this break is *left-mature* (resp. *right-mature*) if a_i (resp. a_{i+1}) is unique; otherwise it is called *left-immature* (resp. *right-immature*). If a break is both left- and right-mature, we call it *mature.* If it is neither, we call it *immature.* The intuition behind these terms is that, if, say, a break $i, i+1$ is left-mature, then the value of $\rho_t(i)$ does not change anymore as t grows. We extend this terminology to the endpoints of A. For the left endpoint, if a_1 is unique in ρ_t we call this endpoint *right-mature*, otherwise it is immature. Analogous definitions apply to the right endpoint.

Claim C: For any step t and an unmarked position i, if $\rho_t(i) \neq \pi(i)$ then a_i is unique.

Consider the first t for which $\rho_t(i) \neq \pi(i)$. Then, by the definition of ρ_t, S_t must contain the other occurrence of a_i, and thus in ρ_t this other occurrence is marked. We conclude that a_i is unique in ρ_t.

This fact immediately yields the following:

Claim D: For any step t, if a break $i, i+1$ of ρ_t is induced, then one of symbols a_i, a_{i+1} must be either marked or unique.

Claim E: Suppose that S_t^A contains a unique letter that belongs to a block R of ρ_{t-1}. Then the whole block R is contained in S_t^A.

This fact follows directly from the definition of the algorithm, for GREEDY will match this unique letter with its occurrence in B and then extend the match to at least the boundaries of R.

Potential. We first assign potentials to the breaks and endpoints of ρ_t:

(ϕ1) Each induced break has potential 1.

(ϕ2) The potential of an original break depends on the degree of maturity. If a break is immature it has potential 3. If it is left-mature and right-immature, or vice versa, it has potential 2. If it is mature, it has potential 1.

(ϕ3) The left endpoint has potential 2 or 1, depending on whether it is right-immature or right-mature, respectively. The potential of the right endpoint is defined in a symmetric fashion.

The potential Φ_t of ρ_t is defined as the sum of the potentials of the breaks and endpoints in ρ_t. For $t = 0$, all breaks in π have potential at most 3 and the endpoints have potential at most 2, yielding $\Phi_0 \leq 3 \cdot \#\mathit{breaks}(\pi) + 4 = 3 \cdot \#\mathit{blocks}(\pi) + 1$. For $t = g$, all letters in A are marked, therefore the potentials of all breaks and endpoints are equal 1 and $\Phi_g \geq \#\mathit{breaks}(\rho_g) + 2 = \#\mathit{blocks}(\rho_g) + 1$. Properties (P1) and (P3) hold.

Proof of property (P2). Some breaks $i, i+1$ of ρ_{t-1} could disappear in ρ_t, either because they are inside S_t^A, or because after changing the values of ρ_{t-1}, we might have $\rho_t(i+1) = \rho_t(i) + 1$. These changes do not increase the potential. The potentials of the breaks of ρ_{t-1} that remain in ρ_t also do not increase.

Thus only the new breaks (i.e., those in ρ_t but not in ρ_{t-1}) can contribute to the increase of the potential. According to Lemma 2.2, there are three types of new breaks in ρ_t:

(B0) Breaks induced by the endpoints of S_t^A.

(B1) Breaks induced by the breaks of ρ_{t-1} inside S_t^A.

(B2) Breaks induced by the endpoints of S_t^B.

All these new breaks have potential 1 in ρ_t. We consider the three types of breaks separately. We show that with each new break we can associate some old break (or an endpoint) of ρ_{t-1} that either disappears in ρ_t or whose potential decreases. This mapping is not necessarily one-to-one. However, the number of new breaks associated with each old break does not exceed the decrease of the potential of this old break. This way we can "pay" for the potential of the new breaks.

Breaks of type (B0). Each such new break is inside some block of ρ_{t-1}. If there are no such breaks, we are done. If there is one new break of type (B0), then there must be at least one break of ρ_{t-1} inside S_t^A (because S_t^A was chosen greedily), and we use one unit of its potential to pay for the new break.

If there are two new breaks of type (B0), we distinguish two subcases. If S_t^A contains two breaks of ρ_{t-1} inside it, then we use one unit of each of these breaks' potentials to pay for the endpoints of S_t^A. If S_t^A contains just one break inside, say $j, j+1$, then this break is immature, for otherwise S_t^A would have to contain a whole block, by Claim E, contradicting the assumptions of this case (as then S_t^A would also have to contain the two old breaks adjacent to this block). Then $j, j+1$

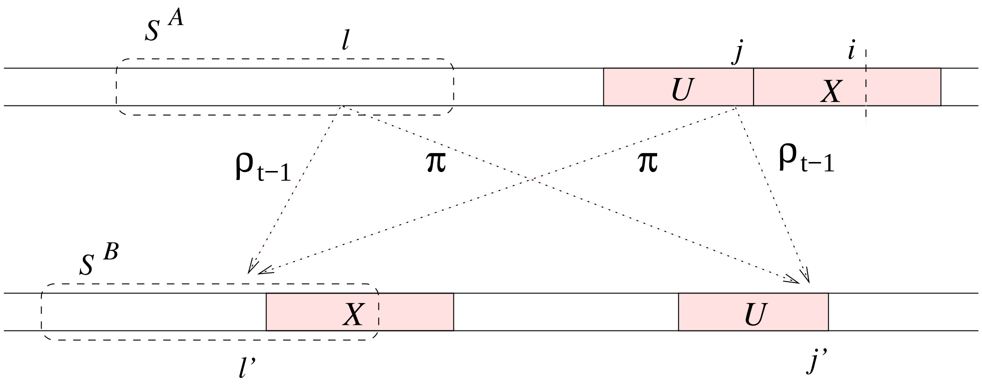

Figure 2: Charging of new breaks of type (B2)

has potential 3 in ρ_{t-1}, and we use two units from this potential to pay for the two breaks of type (B0).

Before proceeding further, we stress that at this point all breaks of ρ_{t-1} inside S_t^A whose previous potential was 2 or 3 still have at least one unit of potential left.

Breaks of type (B1). Let $i, i+1$ be a new break of type (B1). Since each letter appears in A at most twice, the exponent λ in the break classification in Lemma 2.2, Case (B1), must be equal 1. This means that $j, j+1$ is a break of ρ_{t-1}, where $j = \delta^{-1}\rho_{t-1}(i)$ (and $j+1 = \delta^{-1}\rho_{t-1}(i+1)$) and $j, j+1 \in S_t^A$. Further, a_j and a_{j+1} are not unique, so $j, j+1$ is an immature break, and thus it has potential 3. By the previous paragraph, at least one unit of this potential is still unused, so we can use it to pay for the new break $i, i+1$.

In the previous paragraph we used only the potentials of immature breaks contained in S_t^A. So the breaks whose previous potential was 2 (left mature and right immature, or vice versa) still have one unit of potential left.

Breaks of type (B2). The argument for these breaks is more tedious. Suppose that the right endpoint of S_t^B induces a break $i, i+1$ in ρ_t. Let X be a block of ρ_{t-1} that contains $i, i+1$, and let $j+1$ be the first position in X^A. The situation is depicted in Figure 2. (Block X^A can appear before or after S^A.) Assume that $j+1 \neq 1$, that is, X^A is not the first block in A (if X^A is the first block, then the left endpoint of A is right-immature and we charge the new break $i, i+1$ to this endpoint of A). We distinguish two cases.

Subcase B2.1: $j, j+1$ is an original break. Since a_{j+1} is not unique, $j, j+1$ is a right-immature break in ρ_{t-1}. This break becomes right-mature in ρ_t, its potential decreases by 1, and we use this unit of the potential to pay for the new break $i, i+1$.

Subcase B2.2: $j, j+1$ is an induced break. This case is harder, because the potential of $j, j+1$ will not decrease in step t. We are going to find an original, left-mature right-immature break $l, l+1$ in S^A. By the earlier discussion, one unit of potential of such a break can be used to pay for the new break $i, i+1$.

Let $l = \pi^{-1}(\rho_{t-1}(j))$. (Note that the assumption of the case implies $t \geq 2$.) By the definition of j, by the fact that $j, j+1$ is an induced break, and since a_{j+1} is not unique (and thus $\rho_{t-1}(j+1) = \pi(j+1)$), we have $\pi(j) \in S^B$. That is, one of the occurrences of the letter a_j in the string B is in the block S^B. The index l specifies the position of the other occurrence of the letter a_j in A, and since S^A and S^B contain the same letters, we have $l \in S^A$. Since $\pi(j)$ is not the last (rightmost) letter in S^B, we have also $l+1 \in S^A$.

Claim F: $l, l+1$ is an original break.

Let U^A be the block of ρ_{t-1} to the left of X^A and let $j' = \rho_{t-1}(j)$, $l' = \pi(j)$. In some previous step GREEDY matched (marked) U^A with U^B, and did not extend this match to a_{j+1}. There could be three reasons for this: either $j' = n$, or $a_{j+1} \neq b_{j'+1}$, or the block to the right of U^B had already been marked. If $j' = n$, Claim F is trivial. If $a_{j+1} \neq b_{j'+1}$, then, since $a_{l+1} = b_{l'+1} = a_{j+1}$, we have $a_{l+1} \neq b_{j'+1}$ and the claim follows. Finally, suppose that the block to the right of U^B is marked. Then, since there are still two unmarked copies of the letter a_{j+1} in A, we have $b_{j'+1} \neq a_{j+1}$ and the previous argument applies. Claim F is thus proven.

By Claim F, $l, l+1$ is an original break. Since $a_{l+1} = a_{j+1}$, this break is right-immature in ρ_{t-1}. Further, since $a_l = a_j$ and a_j is marked, this break is also left-mature. Therefore its potential in ρ_{t-1} was 2, and we still have one unit of its potential left to pay for the new break $i, i+1$.

We have shown how to charge the break of type (B2) induced by the right endpoint of S_t^B to some break whose potential decreases. In the same way we can charge the break of type (B2) (if any) induced by the left endpoint of S_t^B. It remains to show that the charges generated from these two cases do not conflict.

Indeed, for both endpoints, in case (B2.1) we charge to breaks outside S^A, while in case (B2.2) to breaks inside S^A. In case (B2.1), the right endpoint is charged to $j, j+1$, which is then an original right-immature break. It actually can happen that the left endpoint is charged to the same break $j, j+1$, but if so, $j, j+1$ would be also left-immature and start with potential 3. In case (B2.2), to charge the right endpoint, we identify some right-immature and left-mature break inside S^A, while for the left endpoint we would find a right-mature and left-immature break inside S^A. Thus no conflicts occur.

The above argument shows that we can pay for the potentials of the new breaks in ρ_t using the decrease of the potentials of some breaks of ρ_{t-1}. This completes the proof of property (P2).

Summarizing, we obtain the upper bound of 3 for 2-MCSP, the upper bound of Theorem 1.1(c).

7 Lower Bound for GREEDY on 2-MCSP

In this section we prove that the approximation ratio of GREEDY on 2-MCSP instances is not better than 3, matching the upper bound of the previous section.

For a large even integer l, let $A' = a_1 a_2 \ldots a_{l^2}$, where $a_1, a_2, \cdots, a_{l^2}$ are l^2 distinct letters. We also define a string $B' = b_1 b_2 \ldots b_{l^2}$, where the letters b_i are determined as follows. For $i \not\equiv 1 \pmod{l+1}$, let $b_i = a_i$. For all $i \equiv 1 \pmod{l+1}$, let b_i be new letters, distinct from each other and from all a_j. Define two more strings

$$\begin{aligned} A'' &= a_{l^2-l+2}a_{l^2-l+3}\ldots a_{l^2} \quad \ldots \quad a_{l+1}a_{l+2}\ldots a_{2l-1} \quad a_2a_3\ldots a_l \quad a_1 \\ B'' &= b_{l^2} \quad b_{l^2-l+1}b_{l^2-l+2}\ldots b_{l^2-1} \quad \ldots \quad b_lb_{l+1}\ldots b_{2l-2} \quad b_1b_2\ldots b_{l-1} \end{aligned}$$

Informally, A' and B' consist of the same $l-1$ substrings of length l, separated (and ended) by l *delimiters*, namely by the letters a_1, a_{l+2}, a_{2l+3}, ... , a_{l^2} in A', and b_1, b_{l+2}, b_{2l+3}, ... , b_{l^2} in B'. String A'' is obtained from A' by cutting it into a singleton a_1 and $l+1$ substrings of length $l-1$, each that we refer to as *A-slices*, and concatenating them in reverse order. Similarly, B'' consists of a singleton b_{l^2} and $l+1$ *B-slices*, concatenated in reverse order. Note that the A-slices and B-slices are not aligned with respect to each other.

For example, for $l = 4$, using notation $\dot{a}_i$ and $\ddot{a}_i$ to distinguish the delimiters from other letters, we have

$$A' = \dot{a}_{01}\, a_{02}\, a_{03}\, a_{04}\, a_{05}\, \dot{a}_{06}\, a_{07}\, a_{08}\, a_{09}\, a_{10}\, \dot{a}_{11}\, a_{12}\, a_{13}\, a_{14}\, a_{15}\, \dot{a}_{16}$$

$$
\begin{aligned}
B' &= \ddot{a}_{01}\, a_{02}\, a_{03}\, a_{04}\, a_{05}\, \ddot{a}_{06}\, a_{07}\, a_{08}\, a_{09}\, a_{10}\, \ddot{a}_{11}\, a_{12}\, a_{13}\, a_{14}\, a_{15}\, \ddot{a}_{16} \\
A'' &= a_{14}\, a_{15}\, \dot{a}_{16}\, \dot{a}_{11}\, a_{12}\, a_{13}\, a_{08}\, a_{09}\, a_{10}\, a_{05}\, \dot{a}_{06}\, a_{07}\, a_{02}\, a_{03}\, a_{04}\, \dot{a}_{01} \\
B'' &= \ddot{a}_{16}\, a_{13}\, a_{14}\, a_{15}\, a_{10}\, \ddot{a}_{11}\, a_{12}\, a_{07}\, a_{08}\, a_{09}\, a_{04}\, a_{05}\, \ddot{a}_{06}\, \ddot{a}_{01}\, a_{02}\, a_{03}
\end{aligned}
$$

In this example, the A-slices are $a_{02}a_{03}a_{04}$, $a_{05}\dot{a}_{06}a_{07}$, etc, and the B-slices are $\ddot{a}_{01}a_{02}a_{03}$, $a_{04}a_{05}\ddot{a}_{06}$, etc.

To prove the lower bound, we consider the instance A, B where

$$A = A'\$\#B'' \quad \text{and} \quad B = B'\#\$A'',$$

where \$ and # are two more new letters. As no letter occurs more than twice in A and in B, this is indeed an instance of 2-MCSP.

To obtain a partition, we can match the A-slices in A' and A'', and the B-slices in B' and B'' (as indicated by spaces in the definition of A'' and B''), and we will be left with 4 singletons a_1, b_{l^2}, \$, and #. This shows that $dist(A, B) \leq 2(l + 1) + 4 \leq 2l + 6$.

Now we estimate the number of blocks in the partition computed by Greedy. A' and B' have $l - 1$ common substrings of length l, namely the substrings between the delimiter symbols. We claim that A and B have no other common substrings of length l. Clearly, by the placement of the delimiters, A' and B' have no other common substrings of length l. The longest common substring of A' and A'' as well as of B' and B'' has length $l - 1$, because the A-slices and B-slices have length $l - 1$ and are listed in reverse order. The strings A'' and B'' also have no common substring of length l, since their corresponding slices are not aligned. By the choice of the boundaries \$# and #\$ in the middle of A and B, there is no common substring (of length more than 1) that overlaps these boundaries.

Consequently, Greedy starts by matching the $l - 1$ common substrings of A' and B' of length l. This gives the first $l - 1$ blocks. After this, each letter occurs in the non-marked parts of A', B' exactly once, and thus the bijection between the non-marked letters is unique. It remains to estimate the number of blocks (or breaks) in this bijection.

The remaining l delimiters a_i in A', as well as the two symbols \$ and #, will form another $l + 2$ single-letter blocks in the final partition.

We now bound the number of breaks in B''. Each delimiter b_i, for $i \equiv 1 \pmod{l + 1}$ will form a single-letter block (because of the initial matching of Greedy), and thus we will have a break to the left and right of it in B'' (except for b_{l^2} which appears at the beginning of B''); this gives $2l - O(1)$ breaks. There is also a break before and after each letter b_i, $i \equiv 1 \pmod{l - 1}$ as for such i, $b_i b_{i+1}$ is a consecutive pair in B'' but the possibly matching pair $a_i a_{i+1}$ is not consecutive in A''; similarly $a_{i-1}a_i$ is consecutive in A'' but $b_{i-1}b_i$ is not consecutive in B''. This gives $2l - O(1)$ breaks. Finally, since $l - 1$ and $l + 1$ are relatively prime, only $O(1)$ breaks may be counted twice, by the Chinese remainder theorem.

Altogether, Greedy produces $2l + 1$ blocks of $A'\$\#$ and $4l - O(1)$ blocks of B'', for the total of $6l - O(1)$ blocks. Since the optimal partition has at most $2l + 6$ blocks, the lower bound of 3 on the approximation ratio follows by taking l arbitrarily large.

8 Final Comments

We have established that Greedy's approximation ratio is $O(n^{0.69})$, but not better than $\Omega(n^{0.43})$. It would be interesting to determine the exact approximation ratio of this algorithm. In particular, is it below, above, or equal to $\Theta(\sqrt{n})$? Also, we have observed a difference between the performance of GREEDY on 2-MCSP instances and 4-MCSP instances: Whereas the approximation ratio for 2-MCSP is 3, for 4-MCSP it is not better than $\Omega(\log n)$. The reason for this is, roughly, that for 2-MCSP every new cut (i.e., an induced cut) is adjacent to a unique letter and, since GREEDY does not make mistakes on unique letters, these new cuts do not induce any further cuts. However, for $k > 2$, new cuts may induce yet more new cuts again. An intriguing question is whether for 4-MCSP the upper bound matches the $\Omega(\log n)$ lower bound, or whether it is higher? The question about the exact approximation ratio of GREEDY for k-MCSP remains open even for $k = 3$.

References

[1] X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, T. Jiang. Assignment of orthologous genes via genome rearrangement. Submitted. 2004.

[2] G. Cormode, J.A. Muthukrishnan, The string edit distance matching with moves. Proc. 13th Annual Symposium on Discrete Algorithms (SODA), pp. 667-676, 2002.

[3] A. Goldstein, P. Kolman, and J. Zheng: Minimum common string partitioning problem: Hardness and approximations. Manuscript. 2004.

[4] J. B. Kruskal and D. Sankoff. An anthology of algorithms and concepts for sequence comparison. In *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, Edited by David Sankoff and Joseph B. Kruskal, Addison-Wesley.* 1983.

[5] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals (in Russian). Doklady Akademii Nauk SSSR, 163(4):845–848, 1965.

[6] D. Lopresti, A. Tomkins. Block edit models for approximate string matching. Theoretical Computer Science 181 (159–179) 1997.

[7] D. Shapira, J.A. Storer. Edit distance with move operations. Proc. 13th Annual Symposium on Combinatorial Pattern Matching (CPM), pp. 85–98, 2002.

[8] W. F. Tichy. The string-to-string correction problem with block moves. ACM Trans. Computer Systems 2 (309–321) 1984.