Theoretical Computer Science 108 (1993) 365–369
Elsevier

Note

The shortest common nonsubsequence problem is NP-complete

M. Middendorf

Institut für Angewandte Informatik und Formale Sprachen, Postfach 6980, W-7500 Karlsruhe, Germany

Communicated by D. Perrin
Received September 1991
Revised January 1992

Abstract

Middendorf, M., The shortest common nonsubsequence problem is NP-complete, Theoretical Computer Science 108 (1993) 365–369.

The Shortest Common Nonsubsequence (SCNS) problem is: Given a finite set L of strings over an alphabet Σ and an integer $k \in \mathbb{N}$, is there a string of length $\leqslant k$ over Σ that is not a subsequence of any string in L? The SCNS problem is shown to be NP-complete for strings over an alphabet of size $\geqslant 2$.

1. Introduction

Given a string T over an alphabet Σ, a subsequence of T is any string that can be obtained by erasing zero or more symbols from T. A subsequence S of T is called substring of T if T is of the form $T'ST''$, where T' and T'' are strings over Σ.

The Shortest Common Nonsubsequence (SCNS) problem is to find, for a finite set of strings L over an alphabet Σ and an integer $k \in \mathbb{N}$, a string S of length $\leqslant k$ over Σ that is not a subsequence of any string in L. Analogously, the Shortest Common Nonsubstring problem is to find, for a finite set of strings L over an alphabet

Correspondence to: M. Middendorf, Institut für Angewandte Informatik und Formale Sprachen, Postfach 6980, W-7500 Karlsruhe, Germany. Email: mmi@aifb.uni-karlsruhe.de.

Σ and an integer $k \in \mathbb{N}$, a string S of length $\leqslant k$ over Σ that is not a substring of any string in L.

The SCNS problem and the Shortest Common Nonsubstring problem have been introduced by Timkovskii [5]. Both problems find applications in mechanical engineering (see [5]).

Timkovskii stated that the Shortest Common Nonsubstring problem is polynomial-time-solvable. He raises the question whether the SCNS problem is polynomial-time-solvable too.

In this paper we give a negative answer to this question (unless P = NP). We show that the SCNS problem is NP-complete for strings over an alphabet of size $\geqslant 2$.

Moreover, we consider a problem which is closely related to the SCNS problem. Let T and T' be two strings over an alphabet Σ and let $L = \{T_1, T_2, \ldots, T_l\}$ be a set of strings over Σ. We say that a string S distinguishes T and T' if S is a subsequence of T (T') and not a subsequence of T' (T). A string S distinguishes T and L if it distinguishes T and T_i for all $i \in [1:l]$.

Polynomial-time algorithms have been proposed to find the shortest string that distinguishes two strings T and T' (cf. Hebrard [2]).

We consider the problem of finding the shortest string that distinguishes a string T and a finite set L of strings. We show that this problem is NP-complete for strings over an alphabet of size $\geqslant 2$.

Our results complete those of Maier [3], Räihä and Ukkonen [4] and Timkovskii [5]. Maier has shown that the Shortest Common Supersequence problem is NP-complete for strings over an alphabet of size $\geqslant 5$. This result has been extended by Räihä and Ukkonen to strings over an alphabet of size $\geqslant 2$. The Shortest Common Supersequence problem is the decision version of the problem of finding, for a given finite set L of strings, a shortest string S such that every string in L is a subsequence of S. Furthermore, Maier has shown that the Longest Common Subsequence problem is NP-complete for strings over an alphabet of size $\geqslant 2$. The Longest Common Subsequence problem is the decision version of the problem of finding, for a given finite set L of strings, a longest string that is a subsequence of every string in L. Timkovskii investigated the Longest Common Nonsupersequence problem (see [5] for results). The Longest Common Nonsupersequence problem is to find, for a given finite set L of strings over an alphabet Σ, a longest string S over Σ such that there exists a string in L that is not a subsequence of S (such a string S does not necessarily exist).

2. Basic definitions and notations

An alphabet Σ is a finite set of symbols. A string over Σ is a finite sequence of the symbols of Σ. The concatenation of two strings S and T is denoted by ST.

For a string T and $k \in \mathbb{N}$, define iteratively $(T)^k = T(T)^{k-1}$. $(T)^0$ denotes the empty string. If T consists of a single symbol the brackets are omitted.

Let $T=t_1t_2\ldots t_k$ be a string over an alphabet Σ. Then t_h is *between* t_i and t_j if $i<h<j$ holds, $h, i, j\in[1:k]$. We say that t_h, $h\in[1:k]$ is the *nth a* in T for a symbol $a\in\Sigma$ if there are exactly n indices $i_1, i_2, \ldots, i_n\in[1:h]$, $i_1<i_2<\cdots<i_n=h$ such that $t_{i_j}=a$ for all $j\in[1:n]$.

Given a string T, a *subsequence of* T is any string S that can be obtained from T by erasing zero or more symbols from T. A string S is a *nonsubsequence of* T if S is not a subsequence of T. A string S is a *common nonsubsequence* of a set L of strings if S is a nonsubsequence of every string in L.

Let a string $S=s_1s_2\ldots s_l$ be a subsequence of a string $T=t_1t_2\ldots t_k$ over an alphabet Σ. An *embedding of* S *in* T is a strong growing function f from $[1:l]$ to $[1:k]$ such that $s_i=t_{f(i)}$ for all $i\in[1:l]$. We say that s_i *is mapped onto* $t_{f(i)}$ by f, $i\in[1:l]$. An embedding f of S in T is *leftmost* if, for every embedding g of S in T, we have that $f(i)\leqslant g(i)$ holds for all $i\in[1:l]$.

The reader is assumed to be familiar with the theory of NP-completeness (see [1]).

3. The Shortest Common Nonsubsequence problem

Problem: SHORTEST COMMON NONSUBSEQUENCE (SCNS)

Instance: A finite set L of strings over an alphabet Σ. An integer $k\in\mathbb{N}$.

Question: Does L have a common nonsubsequence of length $\leqslant k$ over Σ?

Theorem 3.1. *SCNS* is NP-*complete for strings over an alphabet of size* $\geqslant 2$.

Proof. SCNS is in NP because it can be tested in polynomial time whether a given string is a common nonsubsequence of a finite set L of strings.

To show that SCNS is NP-complete, we reduce the Vertex Cover problem to it. Let a graph $G=(V, E)$, with node set $V=\{v_1, v_2, \ldots, v_n\}$ and edge set $E=\{e_1, e_2, \ldots, e_m\}$ and an integer $k'\leqslant n$, be an instance of Vertex Cover. Recall that the Vertex Cover problem asks whether G has a vertex cover of size $\leqslant k'$, i.e. a subset $V'\subset V$ with $|V'|\leqslant k'$ such that, for each edge $\{v_i, v_j\}\in E$, at least one of v_i and v_j is in V'. We now construct a set L of strings over the alphabet $\Sigma=\{0, 1\}$ as follows.

Define

$$T=(0^{2n+3}1)^{k'-1}0^{2n+3}$$

and

$$T'=(1^{k'+1}0)^{2n+1}1^{k'+1}.$$

For every edge $e_l=\{v_i, v_j\}\in E$, $i<j$, $l\in[1:m]$, define

$$T_l=(1^{k'}0)^{2i}0(1^{k'}0)^{2(j-1-i)+1}0(1^{k'}0)^{2(n-j)+1}1^{k'}.$$

Now set $L=\{T_1, T_2 \ldots, T_m, T, T'\}$ and $k=2n+2+k'$.

We show that the following holds:

L has a common nonsubsequence of length $\leqslant k$ over $\{0, 1\}$ iff

G has a vertex cover of size $\leqslant k'$.

$\Rightarrow$: We need the following two claims.

Claim 3.2. *A string S of length $\leqslant 2n+2+k'$ over $\{0, 1\}$ is a nonsubsequence of T and T' iff S contains exactly $2n+2$ zeros and k' ones.*

Proof. The claim is a consequence of the following facts which are easy to obtain: Every string of length $\leqslant 2n+2+k'$ over $\{0, 1\}$ that contains at most $k'-1$ ones is a subsequence of T. Every string of length $\leqslant 2n+2+k'$ over $\{0, 1\}$ that contains at most $2n+1$ zeros is a subsequence of T'. Every string that contains $2n+2$ zeros and k' ones is a nonsubsequence of T and T'. □

Claim 3.3. *Let S be a string over $\{0, 1\}$ that contains exactly $2n+2$ zeros and k' ones, and $e_l=\{v_i, v_j\}$, $i<j, l\in[1:m]$, be an edge of G. S is a nonsubsequence of T_l iff $0^{2i}10^{2(n-i+1)}$ or $0^{2j}10^{2(n-j+1)}$ is a subsequence of S.*

Proof. At first observe that S and T_l both contain exactly $2n+2$ zeros. This implies that, if there exists an embedding f of S in T_l, then for all $p\in[1:2n+2]$ the pth zero in S is mapped onto the pth zero in T_l by f.

Assume first that S contains $0^{2i}10^{2(n-i+1)}$ or $0^{2j}10^{2(n-j+1)}$ as a subsequence. Then, we conclude from the observation just mentioned, that an embedding of S in T_l cannot exist, since there is a one between the $2i$th zero and the $(2i+1)$th zero in S or between the $2j$th zero and the $(2j+1)$th zero in S whereas there are no ones between the corresponding zeros in T_l. Hence, S is a nonsubsequence of T_l.

On the other hand, if S contains neither $0^{2i}10^{2(n-i+1)}$ nor $0^{2j}10^{2(n-j+1)}$ as a subsequence, there is no one between the $2i$th zero and the $(2i+1)$th zero in S and no one between the $2j$th zero and the $(2j+1)$th zero in S. Since S contains only k' ones, it is then easy to find an embedding of S in T_l. Hence, S is a subsequence of T_l. □

Proof of Theorem 3.1 (*conclusion*). Now let S be a common nonsubsequence of L of length $\leqslant k=2n+2+k'$ over $\{0, 1\}$.

From Claim 3.2 it follows that S contains exactly $2n+2$ zeros and k' ones. Let $\{i_1, i_2, \ldots, i_{k''}\}$ be the set of integers for which S contains $0^{2i_j}10^{2(n-i_j+1)}$ as a subsequence for all $j\in[1:k'']$. Then, $k''\leqslant k'$.

Set $V'=\{v_{i_1}, v_{i_2}, \ldots, v_{i_{k''}}\}$.

We conclude from Claim 3.3 that, for every edge $e_l=\{v_i, v_j\}$, $l\in[1:m]$, in G, the string S contains $0^{2i}10^{2(n-i+1)}$ or $0^{2j}10^{2(n-j+1)}$ as a subsequence. Hence, by the definition of V', for every edge $e_l=\{v_i, v_j\}$, $l\in[1:m]$, in G, one of the nodes v_i or v_j is in V'. This means that V' is a vertex cover of size $k''\leqslant k'$ of G.

$\Leftarrow$: Let G have a vertex cover of size $\leqslant k'$. Then, obviously, G has a vertex cover $V'=\{v_{i_1}, v_{i_2}, \ldots, v_{i_{k'}}\}$, $1\leqslant i_1<i_2<\cdots<i_{k'}\leqslant n$ of size k'.

Define

$$S=0^{2i_1}10^{2(i_2-i_1)}10^{2(i_3-i_2)}1\ldots0^{2(i_{k'}-i_{k'-1})}10^{2(n-i_{k'}+1)}.$$

Note that S contains exactly k' ones and $2n+2$ zeros. This implies that S is a nonsubsequence of T' and T''. Note, furthermore, that S contains $0^{2i_j}10^{2(n-i_j+1)}$ as a

subsequence for every $j \in [1:k']$. Now, since V' is a vertex cover of G, for every edge $e_l = \{v_i, v_j\}$, $l \in [1:m]$, in G, one of the nodes v_i or v_j is in V'. Hence, by the definition of S, for every edge $e_l = \{v_i, v_j\}$, $l \in [1:m]$, in G, the string S contains $0^{2i}10^{2(n-i+1)}$ or $0^{2j}10^{2(n-j+1)}$ as a subsequence. Then Claim 3.3 shows that S is a nonsubsequence of T_l for every $l \in [1:m]$. Altogether, we have that S is a common nonsubsequence of L of length k. □

4. The Shortest Distinguishing String problem

Problem: SHORTEST DISTINGUISHING STRING (SDS)

Instance: A string T and a finite set L of strings over an alphabet Σ. An integer $k \in \mathbb{N}$.

Question: Is there a string S of length $\leqslant k$ over Σ that distinguishes T and L?

Theorem 4.1. *Shortest Distinguishing String is* NP*-complete for strings over an alphabet of size* $\geqslant 2$.

Proof. Obviously SDS is in NP.

We reduce the SCNS problem for strings over an alphabet of size 2 to our problem. Let a finite set L of strings over the alphabet $\{0, 1\}$ and an integer $k \in \mathbb{N}$ be an instance of the SCNS problem.

Define $T = (01)^k$.

Now, since every string of length $\leqslant k$ over $\{0, 1\}$ is a subsequence of T, the following holds:

> There exists a string of length $\leqslant k$ over $\{0, 1\}$ that is a common nonsubsequence of L iff
> there exists a string S of length $\leqslant k$ over $\{0, 1\}$ that distinguishes T and L. □

Acknowledgment

We thank the referee for helpful remarks.

References

[1] M.R. Garey and D.S. Johnson, *Computers and Intractability* (W.H. Freeman, San Francisco, 1979).

[2] J.-J. Hebrard, An algorithm for distinguishing efficiently bit-strings by their subsequences, *Theoret. Comput. Sci.* **82** (1991) 35–49.

[3] D. Maier, The complexity of some problems on subsequences and supersequences, *J. ACM* **25** (1978) 322–336.

[4] K.-J. Räihä and E. Ukkonen, The shortest common supersequence problem over binary alphabet is NP-complete, *Theoret. Comput. Sci.* **16** (1981) 187–198.

[5] V.G. Timkovskii, Complexity of common subsequence and supersequence problems and related problems, *Cybernetics* **25** (1990) 565–580; translated from *Kibernetika* **25** (1989) 1–13.