Engineering Applications of Artificial Intelligence 25 (2012) 457–467

Contents lists available at SciVerse ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

An enhanced beam search algorithm for the Shortest Common Supersequence Problem

Sayyed Rasoul Mousavi*, Fateme Bahri, Farzaneh Sadat Tabataba

Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan 84156-83111, Iran

ARTICLE INFO

Article history:
Received 22 February 2011
Received in revised form
19 August 2011
Accepted 22 August 2011
Available online 21 September 2011

Keywords:
Heuristic
Optimization
Shortest common supersequence
Dynamic programming
Sequence analysis
Microarray production

ABSTRACT

The Shortest Common Supersequence Problem asks to obtain a shortest string that is a supersequence of every member of a given set of strings. It has applications, among others, in data compression and oligonucleotide microarray production. The problem is NP-hard, and the existing exact solutions are impractical for large instances. In this paper, a new beam search algorithm is proposed for the problem, which employs a probabilistic heuristic and uses the dominance property to further prune the search space. The proposed algorithm is compared with three recent algorithms proposed for the problem on both random and biological sequences, outperforming them all by quickly providing solutions of higher average quality in all the experimental cases. The Java source and binary files of the proposed IBS_SCS algorithm and our implementation of the DR algorithm and all the random and real datasets used in this paper are freely available upon request.

1. Introduction

The Shortest Common Supersequence (SCS) problem asks to obtain a shortest string that is a supersequence of every member of a given set of strings. A supersequence of a given string is a string that can be obtained by inserting zero or more characters anywhere in the given string. Among various applications of this problem are data compression (Storer, 1988; Timkovskii, 1989), AI planning (Foulser et al., 1992), query optimization in databases (Chaudhuri and Bruno, 2008; Sellis, 1988), and bioinformatics, particularly DNA oligonucleotide microarray production (Hubbell et al., 1996; Kasif et al., 2002; Ning et al., 2005; Rahmann, 2003; Sankoff and Kruskal, 1983). Microarrays are precious tools successfully used, among others, in gene clustering and identification, SNP detection, and fusion transcript detection(Ning et al., 2005; Rahmann, 2003; Skotheim et al., 2009). Two well-known types of microarrays are cDNA and oligonucleotide microarrays (Kasif et al., 2002; Ning et al., 2005), the latter known to be of higher sensitivity due to its lower cross-hybridization possibility (Kasif et al., 2002; Ning et al., 2005). Oligonucleotide microarrays are usually manufactured by the photolithographic method. This method involves several synthesis steps, each to append a same nucleotide, which corresponds to a letter in {A,T,C,G}, to several designated probes. Since the process is accomplished by means of light exposure, the other probes, which are not to receive the nucleotide, are protected by a mask. The sequence of the nucleotides used in the synthesis steps is called the *deposition string*, whose length determines the number of the synthesis steps. For several reasons, it is desirable to keep the deposition string as short as possible (Kasif et al., 2002; Ning et al., 2005; Rahmann, 2003). First, the masks and the synthesis steps are expensive. Even a small reduction in the length of the deposition string could lead to a significant reduction in the production cost (Rahmann, 2003). Second, the total manufacturing time is increased as the number of synthesis steps is raised. Third, there exist possibilities for errors in microarray fabrication, because the masking task is not perfect; the probability for a masked probe to be exposed to the light is nonzero. Consequently, the probability for fabrication errors is usually increased as the number of the synthesis steps is raised. Therefore, a shorter deposition sequence is desirable to reduce the manufacturing cost, time, and error. On the other hand, the deposition sequence is a common supersequence of the underlying probes. This motivates the design of high quality algorithms for the SCS problem. Fig. 1 illustrates how the use of a shorter deposition sequence can lead to fewer synthesis steps, hence reducing the production cost, time, and error.

The SCS problem can be optimally solved in polynomial time for a fixed number of input strings, but it is NP-hard in general (Maier, 1978). Consequently, it is highly unlikely to obtain a polynomial-time exact algorithm for the problem, unless $P=NP$ (Garey and Johnson, 1979). Exact algorithms proposed for the

* Corresponding author. Tel.: +98 3113915383; fax: +98 3113912451.
E-mail addresses: srm@cc.iut.ac.ir (S.R. Mousavi), f.bahri@ec.iut.ac.ir (F. Bahri), f.tabataba@ec.iut.ac.ir (F.S. Tabataba).

0952-1976/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2011.08.006

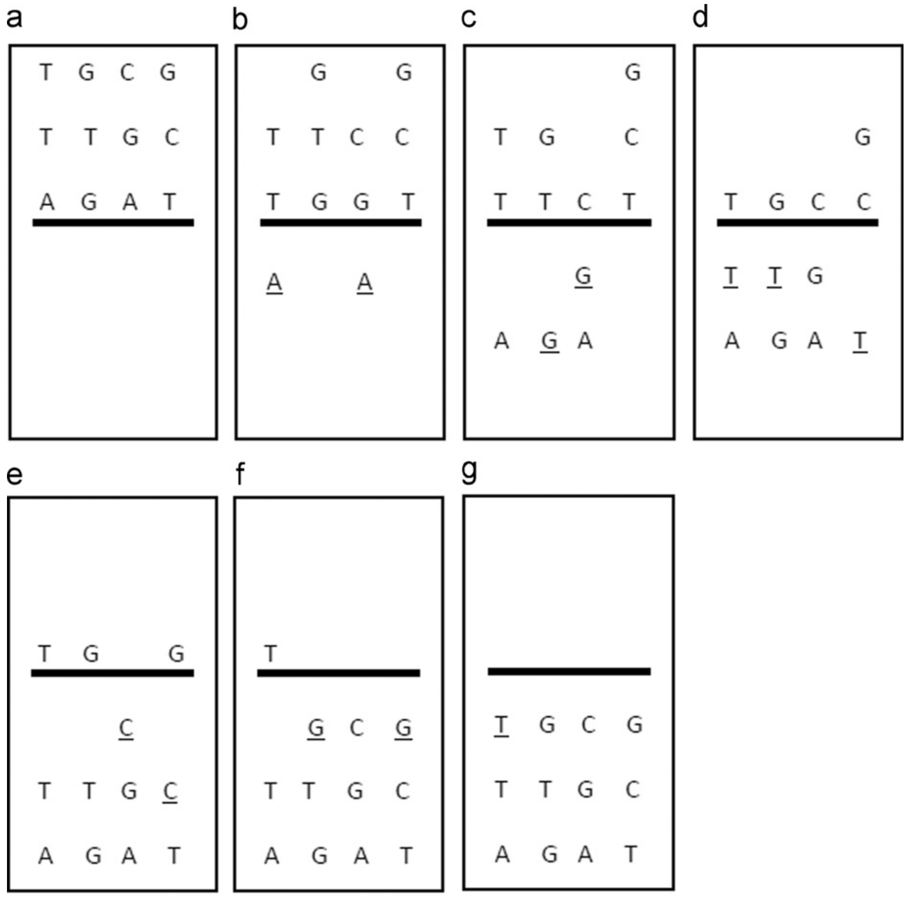

Fig.1. (a) a given set of 3mer oligonucleotides: TTA, GTG, CGA and GCT. (b–g) A step by step illustration of the synthesis process for these oligos. Partially constructed oligos are shown below the black line. The order of adding nucleotides is: AGTCGT (6 steps). If the alphabet method is used, it takes 8 steps (A,C,G,T,A,C,G,T) to build the complete oligos.

problem include a dynamic programming algorithm (Jiang and Li, 1995) and a branch and bound algorithm (Fraser, 1995), which are both exponential, the former in the number of strings and the latter in the size of the corresponding alphabet. Therefore, these algorithms are especially beneficial when, respectively, the number of strings or the alphabet size is restricted. Other research has aimed at devising approximation and (meta) heuristic algorithms, which achieve ‘good’, but not necessarily optimal, solutions in acceptable time. Approximation algorithms for the SCS include Alphabet (Barone et al., 2001), an approximate A* algorithm (Nicosia and Oriolo, 2003), Reduce_Expand (Barone et al., 2001), and Deposition and Reduction (DR) (Ning and Leong, 2006). The approximation ratio of the algorithms Alphabet, Reduce_Expand, and DR is $|\Sigma|$, which is not appealing. The algorithm DR is in fact a trivial combination of a heuristic mechanism with Alphabet, which, therefore, guarantees the approximation ratio of $|\Sigma|$. The approximate A* algorithm provides a $1+\varepsilon$ approximation ratio, for any fixed $\varepsilon>0$, particularly $\varepsilon=0.2$ in the experiments in Nicosia and Oriolo (2003). However, the algorithm is not efficient (i.e. is not of polynomial time complexity) and the size of the search tree can grow exponentially with the size of the given problem instance.

Among (meta) heuristic algorithms for the SCS are Tournament and Greedy (Irving and Fraser, 1993), Majority Merge (Branke et al., 1998), algorithms based on Genetic Algorithms (Branke and Middendorf, 1996; Branke et al., 1998), Ant System and Ant Colony Optimization (Michel and Middendorf, 1998; Michel and Middendorf, 1999), and Min_Height and Sum_Height (Kasif et al., 2002); the latter two specifically proposed for DNA sequences. More recent metaheuristic algorithms include a hybridization of Memetic Algorithms with Beam Search called Hybrid MA_BS (Gallardo et al., 2007), to which we simply refer as MA_BS, and a randomized Beam Search called Probabilistic Beam Search (PBS) (Blum et al., 2007). Another recent algorithm POEMS, together with its variants POEMS_f and POEMS_fw, was also proposed in Kubalik (2010). However, as reported in Kubalik (2010), it was outperformed by MA_BS in all the experimental cases. Based on the results reported in Blum et al. (2007), PBS outperforms MA_BS in most the experimental cases. On the other hand, DR outperforms Alphabet, Tournament, Greedy, and Majority merge in all the experimental cases as reported in Ning and Leong (2006). DR also outperforms Reduce_Expand for strings of length 50–100 (Ning and Leong, 2006). However, no comparison of DR and PBS has yet been made, leaving unclear which one is the state-of-the-art. The time complexity of DR, as specified in Ning and Leong (2006), is $O(|\Sigma|^3nm^2)$, where $|\Sigma|$, n and m are, respectively, the size of the alphabet, the number of strings and the maximum length of the strings. No complexity of PBS or Hybrid MA_BS was reported in their proposing papers (Gallardo et al., 2007; Blum et al., 2007).

In this paper, we provide an improved beam search algorithm called IBS_SCS for the SCS problem, which, on average, outperforms all the three recent algorithms, namely DR, MA_BS, and PBS, in *all* experimental cases. A similar approach has been successfully used for the Longest Common Subsequence (LCS) problem in Mousavi