Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 335 (2005) 53–66

Theoretical Computer Science

www.elsevier.com/locate/tcs

Exact matching of RNA secondary structure patterns

Ying Xu[a, b], Lusheng Wang[a, *], Hao Zhao[a], Jianping Li[a]

[a] *Department of Computer Science, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong, China*

[b] *Computer Science Department, Stanford University, USA*

Abstract

Many RNA structures are assembled from a collection of RNA motifs, which appear repeatedly and in various combinations. Identification of RNA structural motifs will enhance our understanding of RNA structures and functions. Searching for secondary structural patterns in sequence databases is the basic technique and fundamental problem for extracting and identifying such motifs. A number of algorithms and programs have been developed for this purpose.

In this paper, we adopt a representation of secondary structure called *secondary expressions*, and present two algorithms for finding all exact matches of a given secondary expression.

© 2005 Elsevier B.V. All rights reserved.

Keywords: RNA; Match; Algorithm

1. Introduction

RNA secondary structures play an important role in regulating gene expressions. Many of these RNA structures are assembled from a collection of RNA motifs. These basic patterns appear repeatedly and in various combinations to form different RNA types and define their unique structural and functional properties. Identification of RNA structural motifs

* Corresponding author.

E-mail addresses: xuying@stanford.edu (Ying Xu), lwang@cs.cityu.edu.hk (Lusheng Wang), zhaoh@cs.cityu.edu.hk (Hao Zhao), csjpli@cityu.edu.hk (Jianping Li).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.12.016

will therefore enhance our understanding of RNA structures and their association with functional and regulatory elements.

An important technique for extracting and identifying secondary motifs is to search patterns in sequence databases. A number of algorithms and software have been developed for this purpose. Early attempts in structural motif searching were designed for specific families, e.g. FAStRNA [5] for tRNAs, and CITRON [9] for group I introns. Tools for general secondary structures appear in [3,10,14]. In those general purpose tools, description of secondary structures is very flexible, but the major drawback is that such definitions do not allow efficient searching algorithm.

In [6], a representation which extends regular expressions with pairing operators, called *secondary expressions*, is developed for describing secondary structure. They designed algorithms for approximate matching of secondary structures. Unfortunately, the algorithms for approximate matching of secondary structures in [6] do not always give the optimal solution. Here we work on the following problem: given a secondary expression P and a string T, we want to find all (exact) occurrences of P in T.

If P is a string, there are many elegant linear or sub-linear algorithms [7]. If P is a regular expression of size m, one can convert the expression into a nondeterministic finite automaton (NFA) with $O(m)$ nodes and search in $O(mn)$ time, where n is the length of the text string [1]. Another choice is to convert the expression into a DFA and search in $O(n)$ time. However, the DFA might have 2^m states [1].

In this paper, we give algorithms for finding all exact matches of a given secondary expression in $O(nm^2)$ time, where m is the size of the secondary expression and n is the size of the text. The rest of this paper is organized as follows. Definitions and preliminaries are given in Section 2. In Section 3, we propose a dynamic programming algorithm to solve the problem. Section 4 describes a more efficient algorithm by keeping history links.

2. Preliminaries

We first give the definition of secondary expressions and the language they accept. We then discuss how to convert a secondary expression into a finite automaton.

2.1. Secondary expressions

A *network expression* over alphabet Σ is any expression built up with the operations, *concatenation* and *alternation* ($|$), using the symbols in $\Sigma \cup \{\varepsilon\}$, where ε denotes an empty string. Network expressions are, in fact, a subset of regular expressions by excluding the operator of Kleene closure. The motivation of this definition is that the Kleene closure operator is not so useful in most applications in molecular biology [12].

The *complement of network expression E* over alphabet Σ, denoted by E', is the network expression defined recursively by: (1) if $E = \varepsilon$, then $E' = \varepsilon$; (2) if $E = A(, U, G, C)$, then $E' = U(, A, C, G)$; (3) if $E = E_1E_2$, then $E' = E_2'E_1'$; (4) if $E = E_1|E_2$, then $E' = E_1'|E_2'$.

For example, the complement expression of $(A|AG)CC$ is $GG(U|CU)$.