Available online at www.sciencedirect.com
ScienceDirect
Journal of Discrete Algorithms 6 (2008) 37–50
JOURNAL OF DISCRETE ALGORITHMS
www.elsevier.com/locate/jda

Fast pattern-matching on indeterminate strings ☆

Jan Holub [a], W.F. Smyth [b,c,*], Shu Wang [b]

[a] *Department of Computer Science & Engineering, Czech Technical University, Karlovo náměstí 13, Praha 2, CZ-121 35, Czech Republic*
[b] *Algorithms Research Group, Department of Computing & Software, McMaster University, Hamilton, ON L8S 4K1, Canada*
[c] *Department of Computing, Curtin University, GPO Box U1987, Perth, WA 6845, Australia*

Received 5 December 2005; received in revised form 28 September 2006; accepted 3 October 2006

Available online 6 December 2006

Abstract

In a string $\boldsymbol{x}$ on an alphabet Σ, a position i is said to be *indeterminate* iff $\boldsymbol{x}[i]$ may be any one of a specified subset $\{\lambda_1, \lambda_2, \ldots, \lambda_j\}$ of Σ, $2 \leqslant j \leqslant |\Sigma|$. A string $\boldsymbol{x}$ containing indeterminate positions is therefore also said to be *indeterminate*. Indeterminate strings can arise in DNA and amino acid sequences as well as in cryptological applications and the analysis of musical texts. In this paper we describe fast algorithms for finding all occurrences of a pattern $\boldsymbol{p} = \boldsymbol{p}[1..m]$ in a given text $\boldsymbol{x} = \boldsymbol{x}[1..n]$, where either or both of $\boldsymbol{p}$ and $\boldsymbol{x}$ can be indeterminate. Our algorithms are based on the Sunday variant of the Boyer–Moore pattern-matching algorithm, one of the fastest exact pattern-matching algorithms known. The methodology we describe applies more generally to all variants of Boyer–Moore (such as Horspool's, for example) that depend only on calculation of the δ ("rightmost shift") array: our method therefore assumes that Σ is *indexed* (essentially, an integer alphabet), a requirement normally satisfied in practice.

Keywords: String; Word; Algorithm; Pattern-matching; Indeterminate; Approximate; Boyer–Moore; Sunday

1. Introduction

Driven by applications to computational biology, cryptanalysis, musicology, and other areas, there has been recent interest in strings that contain letters that are not uniquely defined. In computational biology, DNA sequences may still be considered to match each other if letter A (respectively, C) is juxtaposed with letter T (respectively, G); analogous juxtapositions may count as matches in protein sequences, and in fact the FASTA format [6] specifically includes indeterminate letters. In cryptanalysis, so far undecoded symbols may be known to match one of a specific set of letters in the alphabet. In music, single notes may match chords, or notes separated by an octave may match.

☆ Supported in part by grants from the Natural Sciences & Engineering Research Council of Canada, the Ministry of Education, Youth and Sports of Czech Republic and Czech Science Foundation. The authors express their gratitude to three anonymous referees, whose comments have materially improved the quality of this paper.

* Corresponding author.
E-mail addresses: holub@fel.cvut.cz (J. Holub), smyth@mcmaster.ca, smyth@computing.edu.au (W.F. Smyth), shuw@mcmaster.ca (S. Wang).
URL: http://www.cas.mcmaster.ca/cas/research/algorithms.htm (W.F. Smyth).

1570-8667/$ – see front matter
doi:10.1016/j.jda.2006.10.003

We refer to a letter $\boldsymbol{x}[i]$ in $\boldsymbol{x}$ that is not uniquely defined as *indeterminate*, and we use the same term to refer to the position i at which it occurs. A string $\boldsymbol{x}$ that may possibly contain indeterminate letters is also called *indeterminate*. The simplest form of indeterminate string is one in which indeterminate positions can contain only a *don't-care* letter—that is, a letter $*$ that matches any letter in the alphabet Σ on which $\boldsymbol{x}$ is defined. In 1974 an algorithm was described [7] for computing all occurrences of a pattern $\boldsymbol{p}$ in a text string $\boldsymbol{x}$, where both $\boldsymbol{p}$ and $\boldsymbol{x}$ are defined on the alphabet $\Sigma \cup \{*\}$, but although efficient in theory, the algorithm was not useful in practice. In 1987 [1] for the first time considered pattern-matching on indeterminate strings in our sense ("generalized pattern-matching"), but the algorithms again were not efficient in practice. In 1992 the bit-mapping technique for pattern-matching (often called the ShiftOr method) was reinvented [2,5,26] and applied (among several other applications) to finding matches for an indeterminate pattern $\boldsymbol{p}$ in a string $\boldsymbol{x}$. Since that time, the resulting `agrep` utility [25] has, with its variants [20–22], been virtually the only practical algorithm available for indeterminate pattern-matching.

Recently, an easily-implemented average-case $O(n)$ time algorithm was proposed [15] for computing all the periods of every prefix of a string $\boldsymbol{x} = \boldsymbol{x}[1..n]$ on $\Sigma \cup \{*\}$. In [11] this work was extended in two ways: first, by distinguishing two distinct forms of indeterminate match ("quantum" and "deterministic") on $\Sigma \cup \{*\}$; second, by refining the definition of indeterminate letters so that they can be restricted to matching only with specified subsets of Σ rather than with every letter of Σ (a case already considered in `agrep`). (Roughly speaking, a "quantum" match allows an indeterminate letter to match two or more distinct letters during a single matching process; a "determinate" match restricts each indeterminate letter to a single match.)

In this paper we present efficient practical algorithms for pattern-matching on indeterminate strings, where indeterminacy may be interpreted in any of the ways described in [11]. Since difficulties arise in efficiently implementing pattern-matching algorithms on indeterminate strings that are based on some form of period [16,19], we do not therefore pursue the ideas of [11,15], but turn rather to the variants of the Boyer–Moore algorithm [3], such as [12,24], that require only knowledge of the rightmost occurrence of each letter in the pattern $\boldsymbol{p}$ (the δ array). As documented in [14,17] and more recently in [8,18], these algorithms are in fact among the fastest available for exact pattern-matching on determinate strings, and as we shall see, these benefits extend also to indeterminate strings.

We begin with an alphabet $\Sigma = \{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ that is finite and moreover, as required by all Boyer–Moore-like algorithms, *indexed*—that is, with the property that each λ_j, $1 \leqslant j \leqslant k$, can be used as an index (or position) in an array, say $\delta = \delta[\lambda_1..\lambda_k]$, allowing $\delta[\lambda_j]$ to be accessed in constant time. Thus without loss of generality we suppose that $\lambda_j = j$, so that $\Sigma = \{1, 2, \ldots, k\}$ is an *integer* alphabet.

In order to model indeterminate letters, we define an *extended alphabet*

$$\Sigma' = \{1, 2, \ldots, k, k+1, \ldots, K\},$$

where to every $j \in k+1..K$ we associate a unique nonempty subset Σ_j of Σ, $|\Sigma_j| \geqslant 2$; for $j \in 1..k$ it is convenient to define $\Sigma_j = \{j\}$. (Throughout, for integers i, $j \geqslant i$, we use the notation $i..j$ to denote the *range* of integers $i, i+1, \ldots, j$.) Thus, in general, $K - k \in 0..2^k - k - 1$. Note that if all letters are determinate, then $K = k$ and $\Sigma' = \Sigma$; while if only the don't-care letter occurs, then $K = k+1$ and $\Sigma_K = \Sigma$. It should normally be true in practice that $K - k$ is "not too large"—perhaps

$$K - k \leqslant d \tag{1}$$

for some small fixed integer d. Note that (1) will always be satisfied for $k = 4$ (the DNA alphabet) if we merely choose $d = 11$.

Our problem in its most general form then becomes the following:

> Compute all occurrences of a pattern $\boldsymbol{p} = \boldsymbol{p}[1..m]$ in a text string $\boldsymbol{x} = \boldsymbol{x}[1..n]$, where both $\boldsymbol{p}$ and $\boldsymbol{x}$ are defined on Σ', and where every $j \in 1..K$ matches any element of Σ_j.

In fact we consider three pattern-matching models in increasing order of sophistication (and processing time requirements), of which only the third is the general form:

(M1) The only indeterminate letter is the don't-care $*$, whose occurrences may be in either $\boldsymbol{p}$ or $\boldsymbol{x}$, or both.
(M2) Arbitrary indeterminate letters can occur, but only in $\boldsymbol{p}$ (like `agrep`).
(M3) Indeterminate letters can occur in both $\boldsymbol{p}$ and $\boldsymbol{x}$.