Available online at www.sciencedirect.com
ScienceDirect
Information Processing Letters 101 (2007) 53–54
Information Processing Letters
www.elsevier.com/locate/ipl

Simple deterministic wildcard matching

Peter Clifford [a], Raphaël Clifford [b,*]

[a] *Department of Statistics, University of Oxford, UK*
[b] *University of Bristol, Department of Computer Science, Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB, UK*

Received 1 June 2006; received in revised form 2 August 2006; accepted 2 August 2006

Available online 1 September 2006

Communicated by L.A. Hemaspaandra

Abstract

We present a simple and fast deterministic solution to the *string matching with don't cares* problem. The task is to determine all positions in a text where a pattern occurs, allowing both pattern and text to contain single character wildcards. Our algorithm takes $O(n \log m)$ time for a text of length n and a pattern of length m and in our view the algorithm is conceptually simpler than previous approaches.

Keywords: Algorithms; String matching; Wildcards

1. Introduction

The problem of determining the time complexity of exact matching with wildcards has been well studied. Fischer and Paterson [2] presented the first solution based on fast Fourier transforms (FFT) with an $O(n \log m \log |\Sigma|)$ time algorithm in 1974. Subsequently, the major challenge has been to remove the dependency on the alphabet size. Indyk [3] gave a randomised $O(n \log n)$ time algorithm which was followed by a simpler and slightly faster $O(n \log m)$ time randomised solution by Kalai [4]. In 2002 a deterministic $O(n \log m)$ time solution was given by Cole and Hariharan [1]. Section 2 presents a new deterministic $O(n \log m)$ solution, which we feel is conceptually simpler than previous approaches.

2. Problem and solution

Let text $t = t_0, \ldots, t_{n-1}$ and pattern $p = p_0, \ldots, p_{m-1}$. The pattern p is said to *occur* at location i in t if, for every position j in the pattern, either $p_j = t_{i+j}$ or at least one of p_j and t_{i+j} is the wildcard symbol.

The main idea of the algorithm is to calculate the sum of squared differences between the pattern and the text for every possible alignment. Suppose that the alphabet is a subset of non-zero integers. If there are no wildcards then for each location $0 \leqslant i \leqslant n - m$ we can calculate

$$\sum_{j=0}^{m-1} (p_j - t_{i+j})^2 = \sum_{j=0}^{m-1} \left(p_j^2 - 2p_j t_{i+j} + t_{i+j}^2\right)$$

in $O(n \log m)$ time using FFTs.[1] Wherever there is an exact match this sum will be exactly 0. When wildcards

* Corresponding author.
E-mail address: clifford@cs.bris.ac.uk (R. Clifford).

[1] We assume the RAM model in order to be consistent with previous work on wildcard matching.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2006.08.002

Input: Pattern p and text t
Output: $A[i] = 0$ iff p occurs at location i in t
Replace each symbol in the pattern and
text by a unique non-zero integer;
Replace all occurrences of the wildcard symbol $*$ with 0;
$A[i] = \sum_{j=0}^{m-1} p_j^3 t_{i+j} - 2\sum_{j=0}^{m-1} p_j^2 t_{i+j}^2 + \sum_{j=0}^{m-1} p_j t_{i+j}^3$;

Algorithm 1. Algorithm for exact matching with wildcards.

are allowed in the pattern and the text we replace the wildcard symbols by 0s and then modify this sum to be

$$\sum_{j=0}^{m-1} p_j t_{i+j}(p_j - t_{i+j})^2 = \sum_{j=0}^{m-1} \left(p_j^3 t_{i+j} - 2p_j^2 t_{i+j}^2 + p_j t_{i+j}^3\right)$$

which equals 0 if and only if there is an exact match with wildcards.

Theorem 1. *The problem of exact matching with wildcards can be solved in* $O(n \log m)$ *time.*

Proof. Consider Algorithm 1 and note that the quantities $\{A[i],\ i = 0, \ldots, n - m\}$ can be calculated in $O(n \log n)$ time by FFTs. Now consider p at location i in the text. $A[i] = 0$ iff for each $j = 0, \ldots, m - 1$, at least one of $p_j = t_{i+j}$, $p_j = 0$ or $t_{i+j} = 0$ is true. Therefore $A[i] = 0$ iff there is an exact wildcard match at location i. □

In order to reduce the time complexity from $O(n \log n)$ to $O(n \log m)$ we employ a standard trick. The text is partitioned into n/m overlapping substrings of length $2m$ with the first substring starting at the beginning of the text and each subsequent substring having an overlap of length m with the previous one. Therefore every position in the text, except for the first and last m positions is covered by exactly two substrings. The matching algorithm is then performed separately on each substring. Each iteration takes $O(m \log m)$ time giving an overall time complexity of $O((n/m)m \log m) = O(n \log m)$.

3. Discussion

Algorithm 1 requires 3 correlation/convolution calculations when wildcards are present in the pattern and text. The pattern and text must also be encoded as integers with each value cubed in the worst case. The accuracy of the calculations must be sufficient to distinguish 0 from other integer values. The deterministic algorithm of Cole and Hariharan encodes the pattern and text as strings of rational numbers of length $2m$ and $2n$, respectively. One correlation calculation is performed on these extended length strings and one on a pair of integer strings of length m and n. $O(\log m)$ bits of accuracy are needed in the representation of each rational value and the accuracy of the calculations must be sufficient to distinguish 0 from values as small as $1/m(m-1)$.

References

[1] R. Cole, R. Hariharan, Verifying candidate matches in sparse and wildcard matching, in: Proceedings of the Annual ACM Symposium on Theory of Computing, 2002, pp. 592–601.
[2] M. Fischer, M. Paterson, String matching and other products, in: R. Karp (Ed.), Proceedings of the 7th SIAM–AMS Complexity of Computation, 1974, pp. 113–125.
[3] P. Indyk, Faster algorithms for string matching problems: Matching the convolution bound, in: Proceedings of the 38th Annual Symposium on Foundations of Computer Science, 1998, pp. 166–173.
[4] A. Kalai, Efficient pattern-matching with don't cares, in: Proceedings of the 13th Annual ACM–SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002, pp. 655–656.