Efficient Pattern-Matching with Don’t Cares

Adam Kalai*

Abstract
We present a randomized algorithm for the string matching with don’t cares problem. Based on the simple fingerprint method of Karé and Rabin for ordinary string matching [4], our algorithm runs in time $O(n \log m)$ for a text of length $n$ and a pattern of length $m$ and is simpler and slightly faster than the previous algorithms [3, 5, 1].

1 Introduction.
We extend the simple randomized fingerprinting algorithm of Karé and Rabin [4] to the problem of string matching with don’t cares. Our algorithm uses a single, simple convolution. This is optimal in the sense that the string matching with don’t cares problem is at least as hard as the boolean convolution problem [6]. Thus, to improve our run time of $O(n \log m)$ on text of length $n$ and pattern of length $m$, one would have to improve on the Fast Fourier Transform.

Fischer and Paterson’s algorithm [1] runs in time $O(n \log m \log \Sigma)$\(^1\). Since their deterministic algorithm in 1974, the only improvements were by Muthukrishnan and Palem [5], who reduced the constant factor, and Indyk [3], who gave a randomized algorithm that also involved convolutions, running in time $O(n \log n)$. In addition to the small time improvement\(^2\) ($O(n \log m)$ over $O(n \log n)$), we hope the conceptually simple algorithm is also of interest. Determining the complexity of this problem was on a list of Galil’s [2] as an open problem.

2 Problem and Solution
Let text $X = x_1 x_2 \cdots x_n$, with $x_i \in \Sigma = \{1, 2, \ldots, s\}$, and a pattern $Y = y_1 y_2 \cdots y_m$, with $y_i \in \Sigma \cup \{\ast\}$. We think of $\ast$ as a “don’t care.” Then, we say that $X(j) = x_jx_{j+1} \cdots x_{j+m}$ matches $Y$ if

$$x_{j-1+i} = y_i \text{ or } y_i = \ast, \text{ for } 1 \leq i \leq m.$$  

The following algorithm finds all positions $j$ where $X(j)$ matches $Y$.

<table>
<thead>
<tr>
<th>Input: Text $X = x_1 \ldots x_n$, pattern $Y = y_1 \ldots y_m$, and parameter $N$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. For $1 \leq i \leq m$, set $r_i = \begin{cases} 0 &amp; \text{if } y_i = \ast \ \text{random from } {1, 2, \ldots, N} &amp; \text{otherwise.} \end{cases}$</td>
</tr>
<tr>
<td>2. Compute $t = \sum_{i=1}^{m} y_i r_i$.</td>
</tr>
<tr>
<td>3. For $1 \leq j \leq n - m$, compute $s(j) = \sum_{i=1}^{m} x_{j-i+1} r_i$ using FFT.</td>
</tr>
<tr>
<td>4. Output MATCH for those $j$’s where $s(j) = t$.</td>
</tr>
</tbody>
</table>

If $X(j)$ matches $Y$, then the above algorithm will certainly output MATCH. However, if $X(j)$ does not match $Y$, then for some $i$, $y_i \neq x_{j+i}$ and $y_i \neq \ast$. Fixing the remaining variables, the equation $s(j) = t$ then has a unique solution for $r_i$, which will choose with probability at most $1/N$. For any $N = n^2$, we would have at most a probability $1/n$ of having any false matches.

Runtime is $O(n \log m)$ assuming we can do calculations of the size $\log(N \Sigma n)$ in constant time, since Fast Fourier Transforms can be done in time $O(n \log m)$. All calculations could also be done modulo a prime $p \geq N$.

3 Don’t Cares in Pattern and Text
The above algorithm can be extended to handle the case of don’t cares in the text $X$ as well. In this case, a $\ast$ in the text matches any symbol in the pattern. We use a second convolution to compute what $s(j)$ should be for a match, which depends on $j$.

$$t(j) = \sum_{1 \leq i \leq m, |x_{j+i} = \ast|} y_i r_i.$$  

This is an FFT of the vectors $(y_i r_i)_{i=1}^{m}$ with $(\delta_{x_j \neq \ast})_{j=1}^{m}$. $(\delta_{x_j \neq \ast}$ is 1 if $x_j \neq \ast$ and 0 if $x_j = \ast$.) If we replace $\ast$’s with 0’s for step 3, then simple calculation shows we will certainly have $s(j) = t(j)$ if $X(j)$ matches $Y$. And for the same reason as above, we will err with probability at most $1/N$.

Acknowledgements. I thank Piotr Indyk and the anonymous SODA referees for helpful comments.
References


