IEICE TRANS. INF. & SYST., VOL.E91–D, NO.6 JUNE 2008

LETTER

A Simple Algorithm for Transposition-Invariant Amplified (δ, γ)-Matching

Inbok LEE[†a)], ***Member***

SUMMARY Approximate pattern matching plays an important role in various applications. In this paper we focus on (δ, γ)-matching, where a character can differ at most δ and the sum of these errors is smaller than γ. We show how to find these matches when the pattern is transformed by $y = \alpha x + \beta$, without knowing α and β in advance.

key words: *combinatorics, pattern matching, fast Fourier transform*

1. Introduction

Approximate pattern matching plays an important role in various applications, such as bioinformatics, computer-aided music analysis and computer vision where the pattern does not appear exactly but within small differences.

Let T and P be strings over a positive integer alphabet Σ. $T[i]$ denotes the i-th character of T. $T[i, j]$ denotes the substring $T[i]T[i+1]\cdots T[j]$. We focus on (δ, γ)-matching which is defined as follows.

Definition 1: Given a text $T = T[1, n]$, a pattern $P = P[1, m]$, and two integer parameters δ and γ, (δ, γ)-matching refers to the problem of finding all the substrings $T[i, i+m-1]$ satisfying two conditions.

- $\forall 1 \le j \le m, |T[i+j-1] - P[j]| \le \delta$ (δ-matching).
- $\sum_{j=1}^{m} |T[i+j-1] - P[j]| \le \gamma$ (γ-matching).

Usually the size of alphabet $|\Sigma|$ (the number of elements in Σ) is large and the value of δ is small.

In addition, we consider *transposition-invariant amplified matching* where each character of P is multiplied by an arbitrary integer α (amplified) and added by another integer β (transposition-invariant).

Definition 2: Transposition-invariant amplified (δ, γ) matching refers to the problem of finding all the substrings $T[i, i+m-1]$ satisfying two conditions with two integers α and β which are not known in advance.

- $\forall 1 \le j \le m, |T[i+j-1] - (\alpha P[j] + \beta)| \le \delta$ (δ-matching).
- $\sum_{j=1}^{m} |T[i+j-1] - (\alpha P[j] + \beta)| \le \gamma$ (γ-matching).

Figure 1 shows an example with $\delta = 1$ and $\gamma = 2$. The original pattern P is $(1, 3, 2, 1, 2)$ in (a). In (b), $T = (2, 5, 4, 3, 4)$. We can find an occurrence of $(1, 2)$-matching if P is transformed into $(2, 6, 4, 2, 4)$ by $y = 2x$. Similarly, in (c), $T = (3, 4, 3, 2, 4)$ and by $y = x + 1$, we get an occurrence of $(1, 2)$-matching of $(2, 4, 3, 2, 3)$. Finally, $T = (3, 7, 6, 3, 5)$ and we can find $(1, 2)$-matching of $(3, 7, 5, 3, 5)$ by $y = 2x + 1$ in (d). Note that we know just P, T, δ, and γ. We need to determine α and β if such (δ, γ)-matching by some transform $y = \alpha x + \beta$ exists.

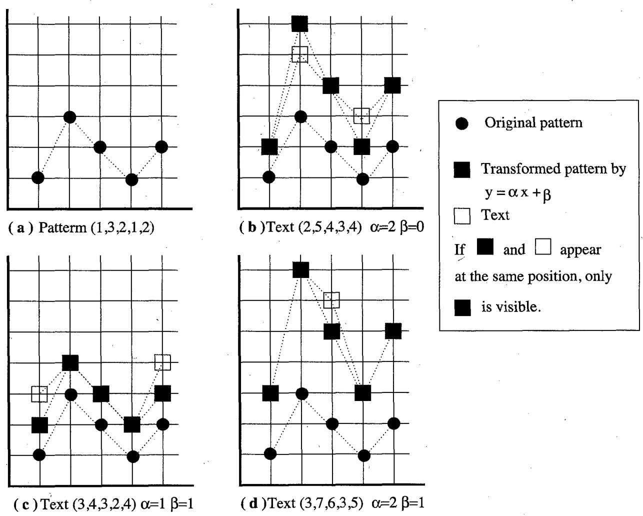

Fig. 1 (a) The original pattern, (b) amplified, (c) transposition-invariant, and (d) transposition-invariant amplified occurrence of the pattern.

This problem arises from computer-aided music analysis. A simple motif (short melody) in music can evolve into different variations by changing the frequency or duration of each note in the motif. In these variations, each character x (either frequency or duration) is transformed by a linear equation $y = \alpha x + \beta$.

(δ, γ)-matching can be solved using the Fast Fourier Transform (FFT, [1]) in $O(\delta n \log m + occ \cdot m)$ time where occ is the number of δ-matches of P in T. First, we find δ-matching of P. The key to our solution is that all the inner-products between $P[1, m]$ and $T[i, i+m-1]$ $(1 \le i \le n-m+1)$

$$P[1, m] \cdot T[i, i+m-1] = \sum_{j=1}^{m} P[j]T[i+j-1]$$

can be calculated in $O(n \log m)$ time [2, Chap 32]. If there is an exact match between $P[1, m]$ and $T[i, i+m-1]$,

$$\sum_{j=1}^{m} (P[j] - T[i+j-1])^2$$

Manuscript received October 22, 2007.
Manuscript revised January 28, 2008.

†The author is with School of Electronic, Telecommunication, and Computer Engineering, Korea Aerospace University, Republic of Korea.
a) E-mail: inboklee@kau.ac.kr

DOI: 10.1093/ietisy/e91–d.6.1824

$$= \sum_{j=1}^{m} P[j]^2 - 2P[1,m] \cdot T[i,i+m-1] + \sum_{j=1}^{m} T[i+j-1]^2$$

should be zero. The total time complexity is $O(n \log m)$ because the first and last terms can be computed in $O(n+m)$ time.

To solve the δ-matching problem, we need a function $g(x,y)$ where $\sum_{j=1}^{m} g(P[j], T[i+j-1]) = 0$ if and only if there is a δ-matching between $P[1,m]$ and $T[i,i+m-1]$. We briefly explain how to define such $g(x,y)$ when $\delta = 1$. Let $g(x,y) = (x-y)^2 + 0.5 \times (-1)^{x+y} - 0.5$. It is easy to show that $g(x,y) = 0$ if $|x-y| \leq 1$ and $g(x,y) > 0$ otherwise. To compute the second term, we define two strings ϵ_T and ϵ_P: $\epsilon_T[i] = 1$ if $T[i]$ is even and $\epsilon_T[i] = -1$ if $T[i]$ is odd. ϵ_P can be defined similarly. Then, using $P[1,m] \cdot T[i,i+m-1]$ and $\epsilon_P[1,m] \cdot \epsilon_T[i,i+m-1]$, we can calculate $\sum_{j=1}^{m} g(P[j], T[i+j-1])$ in $O(n \log m)$ time for all $1 \leq i \leq n-m+1$. The idea can be extended to the general case where $\delta > 1$. In that case the time complexity is $O(\delta n \log m)$. We skip the details of the general case here. Interested readers are directed to [1, pages 70–74].

For the transposition-invariant matching, several algorithms have been proposed recently in [3]–[5] based on sparse dynamic programming, which is not easy to understand and implement.

2. Algorithms

Our aim is to develop a simple and efficient algorithm for finding transposition-invariant amplified (δ,γ)-matches. To do so, we will show how to modify the original FFT matching algorithm in [1].

We first find occurrences of transposition-invariant amplified δ-matches. If $T[i,i+m-1]$ is a δ-match of P by some linear transformation $y = \alpha x + \beta$, we can check whether they are also γ-matches using the technique in [5]. What we need to know is just α. If there exists such a transform $y = \alpha x + \beta$, β should minimize the value of $\sum_{j=1}^{m} |T[i+j-1] - (\alpha P[j] + \beta)|$. For $1 \leq j \leq m$, we compute $T[i+j-1] - \alpha P[j]$. From these m values, we choose the median $T[i+j'-1] - \alpha P[j'] (1 \leq j \leq m)$ in $O(m)$ time. Then we get $\beta = -T[i+j'-1] + \alpha P[j']$ which minimizes the sum of differences. Using this β, we check whether it is also a γ-match of P in $O(m)$ time.

Now our problem is to find δ-matches of P after the transformation $y = \alpha x + \beta$. We first explain how to solve amplified δ-mating (by $y = \alpha x$) and move to transposition-invariant amplified matching (by $y = \alpha x + \beta$).

2.1 Amplified δ-Matching

Now we want to find occurrences of δ-matches of amplified P where every character of P is multiplied by an integer α. If $T[i,i+m-1]$ is such an occurrence, it should satisfy

- $\forall 1 \leq j \leq m$, $|T[i+j-1] - \alpha \cdot P[j]| \leq \delta$, and
- $\sum_{1 \leq j \leq m} |T[i+j-1] - \alpha \cdot P[j]| \leq \gamma$.

The problem is that we do not know α in advance. Once we know the exact value of α, the following equation can be computed in $O(n \log m)$ time and its value should be zero when there is an exact match.

$$\sum_{j=1}^{m} (\alpha \cdot P[j] - T[i+j-1])^2 = \alpha^2 \sum_{j=1}^{m} P[j]^2 - 2\alpha P[1,m] \cdot T[i,i+m-1] + \sum_{j=1}^{m} T[i+j-1]^2.$$

To find δ-matches, we do the same as we did in the original δ-matching problem in $O(\delta n \log m)$ time. The difference is that now we use $\alpha P[j]$ instead of $P[j]$.

We do not know the exact value of α in advance. Therefore, for each substring $T[i,i+m-1]$ $(1 \leq i \leq n-m+1)$, we create an integer array $\alpha[1,n-m+1]$ and store the candidate of α between $P[1,m]$ and $T[i,i+m-1]$ at $\alpha[i]$. To compute $\alpha[i]$, we first select a base element $P[k]$. For simplicity, assume that $P[k]$ is the greatest in $P[1,m]$. Then

$$\alpha[i] = \left\lceil \frac{T[i+k-1]}{P[k]} \right\rfloor (1 \leq i \leq n-m+1).$$

The base element $P[k]$ should meet one condition. From the above equation, $\alpha[i] - 0.5 \leq T[i+k-1]/P[k] < \alpha[i] + 0.5$. Also, if there is a δ-match between $P[1,m]$ and $T[i,i+m-1]$, $|T[i+k-1] - \alpha[i] \cdot P[k]| \leq \delta$. We obtain

$$\alpha[i] \cdot P[k] - \delta \leq T[i+k-1] \leq \alpha[i] \cdot P[k] + \delta$$

and by eliminating $T[i+k-1]$, we obtain

$$\alpha[i] - 0.5 \leq \frac{\alpha[i] \cdot P[k] - \delta}{P[k]} < \alpha[i] + 0.5 \text{ and}$$
$$\alpha[i] - 0.5 \leq \frac{\alpha[i] \cdot P[k] + \delta}{P[k]} < \alpha[i] + 0.5.$$

After some calculation using $\delta \geq 0$, we get $P[k] > 2\delta$: at least one character in P should be greater than 2δ. In real applications this condition can be met easily.

Theorem 1: The amplified (δ,γ)-matching can be solved in $O(\delta n \log m + occ \cdot m)$ time, where occ is the number of candidates.

Proof. Computing the array $\alpha[1,n-m+1]$ takes $O(n)$ time. The FFT runs in $O(\delta n \log m)$ time. After finding occ δ-matches (candidates), each of them requires $O(m)$ time verification. □

2.2 Transposition-Invariant Amplified δ-Matching

We show a simple algorithm without using sparse-dynamic programming. We create two new strings $T' = T'[1,n-1]$

and $P' = P'[1, m-1]$ such that $T'[i] = T[i+1] - T[i]$ and $P'[i] = P[i+1] - P[i]$. Then the following simple lemma holds.

Lemma 1: If there is a (δ, γ)-matching between $P[1, m]$ and $T[i, i+m-1]$, then there is a $(2\delta, 2\gamma)$-matching between $P'[1, m-1]$ and $T'[i, i+m-2]$.

Proof. We first prove 2δ-matching part. If there is an occurrence of δ-matching at position i of T, for $1 \le j \le m-1$, it is evident that

$$-\delta \le T[i+j-1] - P[j] \le \delta \text{ and}$$
$$-\delta \le T[i+j] - P[j+1] \le \delta.$$

It follows that

$$\begin{aligned}-2\delta &\le (T[i+j] - T[i+j-1]) - (P[j+1] - P[j]) \\ &= T'[i+j-1] - P'[j] \le 2\delta.\end{aligned}$$

Now we prove 2γ-matching part. If there is γ-matching between $P[1, m]$ and $T[i, i+m-1]$, $\sum_{j=1}^{m} |T[i+j-1] - P[j]| \le \gamma$. By using the simple fact $|A| + |B| \ge |A + B|$, we obtain

$$\begin{aligned}&\sum_{j=1}^{m-1} |T'[i+j-1] - P'[j]| \\ &= \sum_{j=1}^{m-1} |(T[i+j] - T[i+j-1]) - (P[i+1] - P[i])| \\ &= \sum_{j=1}^{m-1} |(T[i+j] - P[i+1]) + (P[i] - T[i+j-1])| \\ &\le \sum_{j=1}^{m-1} (|(T[i+j] - P[i+1])| + |P[i] - T[i+j-1])|) \\ &\le 2 \cdot \sum_{i=1}^{m} |T[i+j-1] - P[i]| \le 2\gamma.\end{aligned}$$

□

For simplicity, we used the basic (δ, γ)-matching. Once we compute α array from T' and P', amplified (δ, γ)-matching can be solved straightforward. Using this fact, we find occurrences of $(2\delta, 2\gamma)$-matching of P' from T'. The results are candidates for (δ, γ)-matching of P from T. Then we check whether they are real (δ, γ)-matches or not.

Theorem 2: The transposition-invariant (δ, γ)-matching can be solved in $O(\delta n \log m + occ \cdot m)$ time, where occ is the number of candidates.

Proof. Computing T' and P' takes $O(n)$ time ($n > m$). The FFT runs in $O(\delta n \log m)$ time. Again, verifying each of occ candidates takes $O(m)$ time. □

Now we consider the size of occ. If T and P are drawn randomly from Σ, it is easy to show that the probability that $T'[i]$ and $P'[j]$ can have a 2δ-matching is $(4\delta+1)/|\Sigma|$. Hence, the probability is $((4\delta+1)/|\Sigma|)^{m-1}$. The expected number of candidates is $n((4\delta+1)/|\Sigma|)^{m-1}$, which is small when δ is small and $|\Sigma|$ is large.

3. Conclusion

We showed a simple $O(\delta n \log m + occ \cdot m)$ time algorithm for transposition-invariant amplified (δ, γ)-matching. Its space complexity is $O(n)$ [2]. It is an improvement over [5] which requires $O(mn)$ time and space. Furthermore, we also consider the amplified (δ, γ)-matching. The results in [3], [4] cannot be compared directly because their problem is transposition-invariant approximate pattern matching under edit distance. The algorithm in [3] requires $O(n + d^3)$ time and space where d is the maximal edit distance between T and P. The one in [4] runs in $O((mn + \log|\Sigma|)|\Sigma|)$ time and space.

The merit of our algorithm lies in its simplicity. Using the FFT libraries available, it is also easy to implement. Further research includes finding (δ, γ)-matches of P after more complex transformations.

Acknowledgment

This work was supported by 2007 Korea Aerospace University Faculty Research Grant.

References

[1] P. Clifford, R. Clifford, and C.S. Iliopoulos, "Faster algorithms for (δ, γ)-matching and related problems," Proc. 16th Combinatorial Pattern Matching (CPM '05), pp.68–78, 2005.
[2] T.H. Cormen, C.E. Leiserson, and R.L. Rivest, Introduction to Algorithms, MIT Press, 1990.
[3] H. Hyrrö, "Restricted transposition invariant approximate string matching," Proc. 12th String Processing and Information Retrieval (SPIRE '05), pp.257–267, 2005.
[4] K. Lemström, G. Navarro, and Y. Pinzón, "Practical algorithms for transposition-invariant string-matching," J. Discrete Algorithms, vol.3, no.2-4, pp.267–292, 2005.
[5] V. Mäkinen, G. Navarro, and E. Ukkonen, "Transposition invariant string matching," J. Algorithms, vol.56, no.2, pp.124–153, 2005.