APPLICATION OF SEMI-LOCAL LCS TO STRING APPROXIMATE MATCHING*

DIANNE DOE†, PAUL T. FRANK‡, AND JANE E. SMITH‡

Abstract. This is an example SIAM LaTeX article. This can be used as a template for new articles. Abstracts must be able to stand alone and so cannot contain citations to the paper's references, equations, etc. An abstract must consist of a single paragraph and be concise. Because of online formatting, abstracts must appear as plain as possible. Any equations should be inline.

Key words. example, LaTeX

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction.

Approximate string matching is an important task in many fields such as computational biology, signal processing, text retrieval and etc. It also refers to a duplicate detection subtask.

In general form it formulates as follows: Given some pattern p and text t need to find all occurrences of pattern p in text t with some degree of similarity.

There are many algorithms that solve the above problem. Nonetheless, the number of algorithms sharply decreases when the algorithm needs to meet some specific requirements imposed by running time, space complexity or specific criterion for the algorithm itself. For example, recently there was developed an approach for interactive duplicate detection for software documentation[]. The core of this approach is an algorithm that detects approximate clones of a given user pattern with a specified degree of similarity. The main advantage of the algorithm is that it meets a specific requirement of completeness. Nonetheless, it has an unpleasant time complexity.

The algorithm for approximate detection utilizes mainly algorithm for solving the longest commons subsequence (LCS) problem. The longest common subsequence is a well-known fundamental problem in computer science that also has many applications of its own. The major drawback of it that it shows only the global similarity for given input strings. For many tasks, it's simply not enough. The approximate matching is an example of it.

There exist generalazation for LCS called *semi-local LCS* [] which overcome this constraint. The effective theoretical solutions for this generalized problem found applications to various algorithmic problems such as bla bla add cited. For example, there has been developed algorithm for approximate matching in the grammar-compresed strings[].

Although the algorithms for *semi-local LCS* have good theoretical properties, there is unclear how they would behave in practice for a specific task and domain.

To show the applicability of semi-local lcs on practice we developed several algorithms based mainly on it and the underlying algebraic structure. As well as developing new algorithms we improve and significantly outperform the existing one for interactive duplicate detection for software documentation []. It should be noted that improvement preserves all properties of this algorithm. Do we need to state that ant algo is slow for current strucute of algorithm

*Submitted to the editors DATE.

Funding: This work was funded by the Fog Research Institute under contract no. FRI-454.

†Imagination Corp., Chicago, IL (ddoe@imag.com, http://www.imag.com/~ddoe/).

‡Department of Applied Mathematics, Fictional University, Boise, ID (ptfrank@fictional.edu, jesmith@fictional.edu).

The paper is organized as follows. Blablabla
section 2, our new algorithm is in section 3, experimental results are in section 4, and the conclusions follow in section 6.

2. Main results. We interleave text filler with some example theorems and theorem-like items.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Here we state our main result as Theorem 2.1; the proof is deferred to **??**.

THEOREM 2.1 (LDL^T Factorization [1]). *If $A \in \mathbb{R}^{n\times n}$ is symmetric and the principal submatrix $A(1:k, 1:k)$ is nonsingular for $k = 1:n-1$, then there exists a unit lower triangular matrix L and a diagonal matrix*

$$D = \operatorname{diag}(d_1, \ldots, d_n)$$

such that $A = LDL^T$. The factorization is unique.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

THEOREM 2.2 (Mean Value Theorem). *Suppose f is a function that is continuous on the closed interval $[a, b]$. and differentiable on the open interval (a, b). Then there exists a number c such that $a < c < b$ and*

$$f'(c) = \frac{f(b) - f(a)}{b - a}.$$

In other words,

$$f(b) - f(a) = f'(c)(b - a).$$

Observe that Theorems 2.1 and 2.2 and Corollary 2.3 correctly mix references to multiple labels.

COROLLARY 2.3. *Let $f(x)$ be continuous and differentiable everywhere. If $f(x)$ has at least two roots, then $f'(x)$ must have at least one root.*

Proof. Let a and b be two distinct roots of f. By Theorem 2.2, there exists a number c such that

$$f'(c) = \frac{f(b) - f(a)}{b - a} = \frac{0 - 0}{b - a} = 0. \qquad \square$$

Note that it may require two LaTeX compilations for the proof marks to show.

Display matrices can be rendered using environments from `amsmath`:

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad C = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.1}$$

Equation (2.1) shows some example matrices.

We calculate the Fréchet derivative of F as follows:

$$F'(U,V)(H,K) = \langle R(U,V), H\Sigma V^T + U\Sigma K^T - P(H\Sigma V^T + U\Sigma K^T)\rangle \tag{2.2a}$$
$$= \langle R(U,V), H\Sigma V^T + U\Sigma K^T\rangle$$
$$= \langle R(U,V)V\Sigma^T, H\rangle + \langle \Sigma^T U^T R(U,V), K^T\rangle. \tag{2.2b}$$

Equation (2.2a) is the first line, and (2.2b) is the last line.

3. Algorithm. Sed gravida lectus ut purus. Morbi laoreet magna. Pellentesque eu wisi. Proin turpis. Integer sollicitudin augue nec dui. Fusce lectus. Vivamus faucibus nulla nec lacus. Integer diam. Pellentesque sodales, enim feugiat cursus volutpat, sem mauris dignissim mauris, quis consequat sem est fermentum ligula. Nullam justo lectus, condimentum sit amet, posuere a, fringilla mollis, felis. Morbi nulla nibh, pellentesque at, nonummy eu, sollicitudin nec, ipsum. Cras neque. Nunc augue. Nullam vitae quam id quam pulvinar blandit. Nunc sit amet orci. Aliquam erat elit, pharetra nec, aliquet a, gravida in, mi. Quisque urna enim, viverra quis, suscipit quis, tincidunt ut, sapien. Cras placerat consequat sem. Curabitur ac diam. Curabitur diam tortor, mollis et, viverra ac, tempus vel, metus.

Our analysis leads to the algorithm in Algorithm 3.1.

Algorithm 3.1 Build tree

Define $P := T := \{\{1\}, \ldots, \{d\}\}$
while $\#P > 1$ **do**
  Choose $C' \in \mathcal{C}_p(P)$ with $C' := \operatorname{argmin}_{C \in \mathcal{C}_p(P)} \varrho(C)$
  Find an optimal partition tree $T_{C'}$
  Update $P := (P \backslash C') \cup \{\bigcup_{t \in C'} t\}$
  Update $T := T \cup \{\bigcup_{t \in \tau} t : \tau \in T_{C'} \backslash \mathcal{L}(T_{C'})\}$
end while
return T

Curabitur ac lorem. Vivamus non justo in dui mattis posuere. Etiam accumsan ligula id pede. Maecenas tincidunt diam nec velit. Praesent convallis sapien ac est. Aliquam ullamcorper euismod nulla. Integer mollis enim vel tortor. Nulla sodales placerat nunc. Sed tempus rutrum wisi. Duis accumsan gravida purus. Nunc nunc. Etiam facilisis dui eu sem. Vestibulum semper. Praesent eu eros. Vestibulum tellus nisl, dapibus id, vestibulum sit amet, placerat ac, mauris. Maecenas et elit ut erat placerat dictum. Nam feugiat, turpis et sodales volutpat, wisi quam rhoncus neque, vitae aliquam ipsum sapien vel enim. Maecenas suscipit cursus mi.

4. Experimental results. Quisque facilisis auctor sapien. Pellentesque gravida hendrerit lectus. Mauris rutrum sodales sapien. Fusce hendrerit sem vel lorem. Integer pellentesque massa vel augue. Integer elit tortor, feugiat quis, sagittis et, ornare non, lacus. Vestibulum posuere pellentesque eros. Quisque venenatis ipsum dictum nulla. Aliquam quis quam non metus eleifend interdum. Nam eget sapien ac mauris malesuada adipiscing. Etiam eleifend neque sed quam. Nulla facilisi. Proin a ligula. Sed id dui eu nibh egestas tincidunt. Suspendisse arcu.

Figure 1 shows some example results. Additional results are available in the supplement in **??**.

Maecenas dui. Aliquam volutpat auctor lorem. Cras placerat est vitae lectus. Curabitur massa lectus, rutrum euismod, dignissim ut, dapibus a, odio. Ut eros erat,

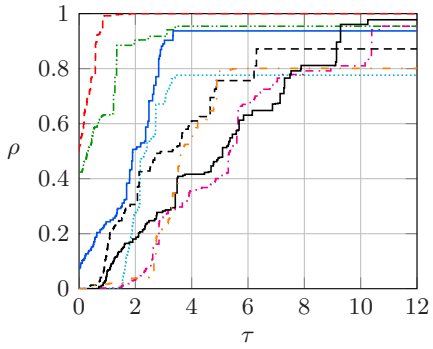

FIG. 1. *Example figure using external image files.*

vulputate ut, interdum non, porta eu, erat. Cras fermentum, felis in porta congue, velit leo facilisis odio, vitae consectetuer lorem quam vitae orci. Sed ultrices, pede eu placerat auctor, ante ligula rutrum tellus, vel posuere nibh lacus nec nibh. Maecenas laoreet dolor at enim. Donec molestie dolor nec metus. Vestibulum libero. Sed quis erat. Sed tristique. Duis pede leo, fermentum quis, consectetuer eget, vulputate sit amet, erat.

5. Discussion of $Z = X \cup Y$. Curabitur nunc magna, posuere eget, venenatis eu, vehicula ac, velit. Aenean ornare, massa a accumsan pulvinar, quam lorem laoreet purus, eu sodales magna risus molestie lorem. Nunc erat velit, hendrerit quis, malesuada ut, aliquam vitae, wisi. Sed posuere. Suspendisse ipsum arcu, scelerisque nec, aliquam eu, molestie tincidunt, justo. Phasellus iaculis. Sed posuere lorem non ipsum. Pellentesque dapibus. Suspendisse quam libero, laoreet a, tincidunt eget, consequat at, est. Nullam ut lectus non enim consequat facilisis. Mauris leo. Quisque pede ligula, auctor vel, pellentesque vel, posuere id, turpis. Cras ipsum sem, cursus et, facilisis ut, tempus euismod, quam. Suspendisse tristique dolor eu orci. Mauris mattis. Aenean semper. Vivamus tortor magna, facilisis id, varius mattis, hendrerit in, justo. Integer purus.

6. Conclusions. Some conclusions here.

Appendix A. An example appendix. Aenean tincidunt laoreet dui. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Integer ipsum lectus, fermentum ac, malesuada in, eleifend ut, lorem. Vivamus ipsum turpis, elementum vel, hendrerit ut, semper at, metus. Vivamus sapien tortor, eleifend id, dapibus in, egestas et, pede. Pellentesque faucibus. Praesent lorem neque, dignissim in, facilisis nec, hendrerit vel, odio. Nam at diam ac neque aliquet viverra. Morbi dapibus ligula sagittis magna. In lobortis. Donec aliquet ultricies libero. Nunc dictum vulputate purus. Morbi varius. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In tempor. Phasellus commodo porttitor magna. Curabitur vehicula odio vel dolor.

LEMMA A.1. *Test Lemma.*

Acknowledgments. We would like to acknowledge the assistance of volunteers in putting together this example manuscript and supplement.

REFERENCES

[1] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 4th ed., 2013.