Streaming algorithms for embedding and computing edit distance in the low distance regime

Diptarka Chakraborty    Elazar Goldenberg    Michal Koucký

Department of Computer Science & Engineering,
Indian Institute of Technology Kanpur

21st June, 2016
Outline

1. Introduction
2. Result on Embedding
3. Embedding Algorithm
   - Description
   - Result
   - Proof Sketch
4. Computing Edit Distance
   - Comparison with Previous Work
   - Proof Sketch
5. Conclusion
1 Introduction

2 Result on Embedding

3 Embedding Algorithm
   • Description
   • Result
   • Proof Sketch

4 Computing Edit Distance
   • Comparison with Previous Work
   • Proof Sketch

5 Conclusion
Problem Definition

Given two strings \( x, y \in \Sigma^* \)

- **Edit distance** \( \Delta_e(x, y) \) - minimum number of
  - insertion,
  - deletion, and
  - character substitution

  operations needed for converting from \( x \) to \( y \)
Problem Definition

Given two strings $x, y \in \Sigma^*$

- **Edit distance** ($\Delta_e(x, y)$) - minimum number of
  - insertion,
  - deletion, and
  - character substitution

operations needed for converting from $x$ to $y$

- **Hamming distance** ($\Delta_H(x, y)$) - minimum number of
  - character substitution

operations needed
Problem Definition

$\Delta_H(x, y)$ can be as large as $n$ (length of the inputs) while $\Delta_e(x, y)$ being only 1.
Problem Definition

\( \Delta_H(x, y) \) can be as large as \( n \) (length of the inputs) while \( \Delta_e(x, y) \) being only 1

Example:
\[
x = babababababab \cdots
\]
\[
y = ababababababa \cdots
\]
Problem Definition

$\Delta_H(x, y)$ can be as large as $n$ (length of the inputs) while $\Delta_e(x, y)$ being only 1

Example:

$x = babababababab \cdots$

$y = ababababababa \cdots$
Problem Definition

\[ \Delta_H(x, y) \] can be as large as \( n \) (length of the inputs) while \( \Delta_e(x, y) \) being only 1.

**Example:**

\[ x = babababababab \cdots \]
\[ y = babababababab \cdots \]
Problem Definition

- The problem is to find a *randomized embedding* from edit metric to Hamming metric with small *distortion factor*
Problem Definition

- The problem is to find a *randomized embedding* from edit metric to Hamming metric with small *distortion factor*.
- Define a map \( f : \Sigma^n \rightarrow \Sigma^{l(n)} \) s.t. \( \forall x, y \in \Sigma^n \),

\[
\beta \cdot \Delta_e(x, y) \leq \Delta_H(f(x), f(y, r)) \leq \alpha \cdot \Delta_e(x, y)
\]

where distortion factor is \( \phi_d = \alpha / \beta \).
Problem Definition

- The problem is to find a \textit{randomized embedding} from edit metric to Hamming metric with small \textit{distortion factor}.

Now instead consider randomized map, i.e.,
\[ f : \Sigma^n \times \{0, 1\}^{r(n)} \to \Sigma^{l(n)} \text{ s.t. } \forall x, y \in \Sigma^n, \text{ w.h.p. (over } r \in \{0, 1\}^{r(n)}) \]

\[
\beta \cdot \Delta_e(x, y) \leq \Delta_H(f(x, r), f(y, r)) \leq \alpha \cdot \Delta_e(x, y)
\]

where distortion factor is \( \phi_d = \alpha / \beta \).
Problem Definition

- The problem is to find a randomized embedding from edit metric to Hamming metric with small distortion factor.
Problem Definition

- The problem is to find a randomized embedding from edit metric to Hamming metric with small distortion factor.
- **Remark:** Previous best known bound by Jowhari ’12: 
  \[ \phi_d \leq O(\log n \log^* n) \]
Problem Definition

- The problem is to find a \textit{randomized embedding} from edit metric to Hamming metric with small \textit{distortion factor}.
- \textbf{Remark:} Previous best known bound by Jowhari ’12: \( \phi_d \leq O(\log n \log^* n) \)

We achieve distortion factor of \( O(\Delta_e(x, y)) \)
Motivation

- From computational perspective, problems on Hamming distance are somehow easier than that on edit distance
- Embedding provides us power to use results from the world of Hamming metric
Motivation

- From computational perspective, problems on Hamming distance are somehow easier than that on edit distance
- Embedding provides us power to use results from the world of Hamming metric
- Applications include
  - Computing edit distance (nontrivial)
  - Document exchange problem under edit metric
  - Designing sketching protocol for gap-edit distance
  - Approximately nearest neighbor search
1 Introduction

2 Result on Embedding

3 Embedding Algorithm
   • Description
   • Result
   • Proof Sketch

4 Computing Edit Distance
   • Comparison with Previous Work
   • Proof Sketch

5 Conclusion
Result on Embedding

There exists a mapping $f : \Sigma^n \times \{0, 1\}^{r(n)} \rightarrow \Sigma^{3n}$ which satisfies the following conditions:

1. For every $x, y$, $\Delta e(x, y) / 2 \leq \Delta H(f(x, r), f(y, r)) \leq O(\Delta e(x, y))$ with probability at least $2/3$.

2. Given $f(x, r)$ and $r$, it is possible to decode back $x$ with probability $1 - \exp(-\Omega(n))$.

Moreover, both the mapping $f$ and its decoding (given $f(x, r)$ and $r$) take linear time and can be performed in a streaming fashion.
There exists a mapping $f : \Sigma^n \times \{0, 1\}^{r(n)} \rightarrow \Sigma^{3n}$ which satisfies the following conditions:

1. For every $x, y$,
   $\Delta_e(x, y)/2 \leq \Delta_H(f(x, r), f(y, r)) \leq O(\Delta_e(x, y))^2$ with probability at least $2/3$. 
There exists a mapping $f : \Sigma^n \times \{0, 1\}^{r(n)} \rightarrow \Sigma^{3n}$ which satisfies the following conditions:

1. For every $x, y$,
   $$\frac{\Delta_e(x, y)}{2} \leq \Delta_H(f(x, r), f(y, r)) \leq O(\Delta_e(x, y))^2$$
   with probability at least $2/3$.

2. Given $f(x, r)$ and $r$, it is possible to decode back $x$ with probability $1 - \exp(-\Omega(n))$. 
Result on Embedding

There exists a mapping $f : \Sigma^n \times \{0, 1\}^{r(n)} \rightarrow \Sigma^{3n}$ which satisfies the following conditions:

1. For every $x, y$,
   $$\Delta_e(x, y)/2 \leq \Delta_H(f(x, r), f(y, r)) \leq O(\Delta_e(x, y))^2$$
   with probability at least $2/3$.

2. Given $f(x, r)$ and $r$, it is possible to decode back $x$ with probability $1 - \exp(-\Omega(n))$.

Moreover, both the mapping $f$ and its decoding (given $f(x, r)$ and $r$) take linear time and can be performed in a streaming fashion.
1 Introduction

2 Result on Embedding

3 Embedding Algorithm
   • Description
   • Result
   • Proof Sketch

4 Computing Edit Distance
   • Comparison with Previous Work
   • Proof Sketch

5 Conclusion
Description of Embedding Algorithm

- Pick a sequence of random functions \( h_1, \ldots, h_{3n} : \Sigma \to \{0, 1\} \)
- Maintain a pointer \( i \) for current position on input and initially set to 1
- In time \( t \leq 3n \), append output by \( x_i \) and increment \( i \) by \( h_t(x_i) \)
- If \( i \) exceeds \( n \), append zeros
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; \ldots 
Consider input $= babba \ldots$
Example

Suppose \( h_1(a) = 0, \ h_1(b) = 1; \ h_2(a) = 0, \ h_2(b) = 0; \ h_3(a) = 1, \ h_3(b) = 0; \ldots \)
Consider input = b abba \ldots
Example

Suppose $h_1(a) = 0, h_1(b) = 1; h_2(a) = 0, h_2(b) = 0; h_3(a) = 1,$ $h_3(b) = 0; \ldots$

Consider input = $b\ abba\ \ldots$

1. $i = 1$, output = $b$. 
Example

Suppose $h_1(a) = 0, h_1(b) = 1; h_2(a) = 0, h_2(b) = 0; h_3(a) = 1, h_3(b) = 0; \cdots$

Consider $input = b\ \text{abba} \cdots$

1. $i = 1, output = b, i = i + h_1(b) = 2$
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; ···
Consider input = b a bba ···

1. $i = 1$, output = b, $i = i + h_1(b) = 2$
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; \ldots

Consider input = b\textcolor{red}{a} bba \cdots

1. $i = 1$, output = b, $i = i + h_1(b) = 2$
2. $i = 2$, output = ba,
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; ⋮
Consider input = $b\underline{a}bba$ ⋮

1. $i = 1$, output = $b$, $i = i + h_1(b) = 2$
2. $i = 2$, output = $ba$, $i = i + h_2(a) = 2$
Example

Suppose \( h_1(a) = 0, \ h_1(b) = 1; \ h_2(a) = 0, \ h_2(b) = 0; \ h_3(a) = 1, \ h_3(b) = 0; \ \cdots \)

Consider \( \text{input} = b \ a \ bba \cdots \)

1. \( i = 1, \ \text{output} = b, \ i = i + h_1(b) = 2 \)
2. \( i = 2, \ \text{output} = ba, \ i = i + h_2(a) = 2 \)
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; \ldots 
Consider input $= ba bba \cdots$

1. $i = 1$, output $= b$, $i = i + h_1(b) = 2$
2. $i = 2$, output $= ba$, $i = i + h_2(a) = 2$
3. $i = 2$, output $= ba a$, 

Diptarka Chakraborty, diptarka@cse.iitk.ac.in
Example

Suppose $h_1(a) = 0$, $h_1(b) = 1$; $h_2(a) = 0$, $h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; \ldots
Consider input = b\textcolor{red}{a} bba \ldots$

1. $i = 1$, output = b, $i = i + h_1(b) = 2$
2. $i = 2$, output = ba, $i = i + h_2(a) = 2$
3. $i = 2$, output = baa, $i = i + h_3(a) = 3$
Example

Suppose $h_1(a) = 0, h_1(b) = 1$; $h_2(a) = 0, h_2(b) = 0$; $h_3(a) = 1$, $h_3(b) = 0$; \ldots
Consider $input = ba b ba \cdots$

1. $i = 1$, $output = b$, $i = i + h_1(b) = 2$
2. $i = 2$, $output = ba$, $i = i + h_2(a) = 2$
3. $i = 2$, $output = baa$, $i = i + h_3(a) = 3$
Example

Suppose \( h_1(a) = 0, \ h_1(b) = 1; \ h_2(a) = 0, \ h_2(b) = 0; \ h_3(a) = 1, \ h_3(b) = 0; \ldots \)
Consider input = \( ba \ b \ ba \ \cdots \)

1. \( i = 1, \ output = b, \ i = i + h_1(b) = 2 \)
2. \( i = 2, \ output = ba, \ i = i + h_2(a) = 2 \)
3. \( i = 2, \ output = baa, \ i = i + h_3(a) = 3 \)
4. \( i = 3, \ output = baab, \ \cdots \)
Recall the Result

There exists a mapping $f : \Sigma^n \times \{0, 1\}^{r(n)} \rightarrow \Sigma^{3n}$ which satisfies the following conditions:

1. For every $x, y$,
   $$\Delta_e(x, y)/2 \leq \Delta_H(f(x, r), f(y, r)) \leq O(\Delta_e(x, y))^2$$
   with probability at least $2/3$.

2. Given $f(x, r)$ and $r$, it is possible to decode back $x$ with probability $1 - \exp(-\Omega(n))$.

Moreover, both the mapping $f$ and its decoding (given $f(x, r)$ and $r$) take linear time and can be performed in a streaming fashion.
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
Idea behind the Upper Bound

Suppose edit distance is 1 (one insertion)

\[ x = \text{bababacdeabacdbab} \cdots \]
\[ y = \text{bababaxcdeabacda} \cdots \]
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index \( i \) where \( x_i \neq y_i \)
  - \( x = bababcdeabacdab \cdots \)
  - \( y = bababaxcdeabacda \cdots \)
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$

$x = \textcolor{red}{\text{c}}\text{deabac}dab \cdots \ f(x, r) = \text{baabbabaa}$

$y = \text{bababa} \ \textcolor{red}{\text{x}} \text{cdeabac}da \cdots \ f(y, r) = \text{baabbabaa}$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = \text{bababa} \textcolor{red}{c} \text{deabacdab} \cdots$ $f(x, r) = \text{baabbabaa}$

$y = \text{bababa} \textcolor{red}{x} \text{cdeabacda} \cdots$ $f(y, r) = \text{baabbabaa}$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = \text{bababa} \cancel{c} \text{deabacdac}\cdots \ f(x, r) = \text{baabbabaac}$

$y = \text{bababa} \cancel{x} \text{cdeabacda}\cdots \ f(y, r) = \text{baabbabaax}$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability 1/4 they got synced, with probability 1/2 defer by 1 and with remaining probability defer by 2

$x = bababa\textcolor{red}{c}deabacdab \cdots \quad f(x, r) = baabbabaacc \cdots$

$y = bababax\textcolor{red}{c}deabacda \cdots \quad f(y, r) = baabbabaaxc \cdots$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = \text{bababa} \text{c deabacdab} \cdots \quad f(x, r) = baabbabaac$

$y = \text{bababa} \text{x cdeabacda} \cdots \quad f(y, r) = baabbabaax$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = bababac\text{deabacdacb}\cdots \ f(x, r) = baabbabaacd$

$y = bababaxc\text{deabacda}\cdots \ f(y, r) = baabbbaaxc$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = bababa\textcolor{red}{c}deabacdab\cdots$ $f(x, r) = baabbabaac$
$y = bababa\textcolor{red}{x}cdeabacda\cdots$ $f(y, r) = baabbabaax$
Idea behind the Upper Bound

- Suppose edit distance is 1 (one insertion)
- Consider the first index $i$ where $x_i \neq y_i$
- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with remaining probability defer by 2

$x = \text{bababac} \text{d} \text{eabac}d \text{ab} \cdots \ f(x, r) = \text{baabbabaacd}$

$y = \text{bababa} \text{x} \text{cdeabac}d \text{a} \cdots \ f(y, r) = \text{baabbabaa}xx$
Idea behind the Upper Bound

- With probability 1/4 they got synced, with probability 1/2 defer by 1 and with probability 1/4 defer by 2
Idea behind the Upper Bound

- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with probability $1/4$ defer by 2
- Same as random walk on integer line starting from position 1 and moving one step left with probability $1/4$, staying in the same position with probability $1/2$ and moving one step right with probability $1/4$
Idea behind the Upper Bound

- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with probability $1/4$ defer by 2
- Same as random walk on integer line starting from position 1 and moving one step left with probability $1/4$, staying in the same position with probability $1/2$ and moving one step right with probability $1/4$
- Probability of sync within $l$ steps is same as probability of visiting origin for the first time within $l$ steps
Idea behind the Upper Bound

- With probability $1/4$ they got synced, with probability $1/2$ defer by 1 and with probability $1/4$ defer by 2
- Same as random walk on integer line starting from position 1 and moving one step left with probability 1/4, staying in the same position with probability 1/2 and moving one step right with probability 1/4
- Probability of sync within $l$ steps is same as probability of visiting origin for the first time within $l$ steps
- For constant probability, we need $l$ to be constant
Idea behind the Upper Bound

- With probability 1/4 they got synced, with probability 1/2 defer by 1 and with probability 1/4 defer by 2
- Same as random walk on integer line starting from position 1 and moving one step left with probability 1/4, staying in the same position with probability 1/2 and moving one step right with probability 1/4
- Probability of sync within $l$ steps is same as probability of visiting origin for the first time within $l$ steps
- For constant probability, we need $l$ to be constant
- $l$ is an upper bound on hamming distance
Some Remarks:

- Similar idea was used for computing edit distance approximately in Saha14.
Some Remarks:

- Similar idea was used for computing edit distance approximately in Saha14
- Idea was to randomly delete mismatched character
Some Remarks:

- Similar idea was used for computing edit distance approximately in Saha14
- Idea was to randomly delete mismatched character
- **Main technical challenge:** do not have access to both the strings at the same time
Some Remarks:

- Similar idea was used for computing edit distance approximately in Saha14
- Idea was to randomly delete mismatched character
- **Main technical challenge:** do not have access to both the strings at the same time
- And also random deletion destroys information content
1 Introduction

2 Result on Embedding

3 Embedding Algorithm
   - Description
   - Result
   - Proof Sketch

4 Computing Edit Distance
   - Comparison with Previous Work
   - Proof Sketch

5 Conclusion
Comparison with Previous Work

<table>
<thead>
<tr>
<th>Authors</th>
<th>Time</th>
<th>Space</th>
<th>Approx. Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>WF74</td>
<td>$O(n^2)$</td>
<td>$O(n)$</td>
<td>Exact</td>
</tr>
<tr>
<td>LMS98</td>
<td>$O(n + k^2)$</td>
<td>$O(n)$</td>
<td>Exact</td>
</tr>
<tr>
<td>Saha14</td>
<td>$O(n)$ (randomized)</td>
<td>$O(\log n)$</td>
<td>$O(k)$</td>
</tr>
<tr>
<td><strong>This paper</strong></td>
<td>$O(n + k^6)$ (randomized with promise)</td>
<td>$O(k^6)$</td>
<td>Exact</td>
</tr>
</tbody>
</table>
Proof Sketch

- Compute an alignment of cost $ck^2$ using technique similar to embedding (randomized linear time)
Proof Sketch

- Compute an alignment of cost $ck^2$ using technique similar to embedding (randomized linear time)
- Using the above alignment and the property on periodicity of input strings, shrink both the inputs to strings (namely, kernels) of size $O(k^6)$ each s.t. edit distance remains unchanged (linear time)
Proof Sketch

- Compute an alignment of cost $ck^2$ using technique similar to embedding (randomized linear time)
- Using the above alignment and the property on periodicity of input strings, shrink both the inputs to strings (namely, kernels) of size $O(k^6)$ each s.t. edit distance remains unchanged (linear time)
- Run algorithm of LMS98 on those kernels ($O(k^6 + k^2)$ time)
Proof Sketch

- Compute an alignment of cost $ck^2$ using technique similar to embedding (randomized linear time)
- Using the above alignment and the property on periodicity of input strings, shrink both the inputs to strings (namely, kernels) of size $O(k^6)$ each s.t. edit distance remains unchanged (linear time)
- Run algorithm of LMS98 on those kernels ($O(k^6 + k^2)$ time)
- Probability of success is at least $2/3$
1. Introduction

2. Result on Embedding

3. Embedding Algorithm
   - Description
   - Result
   - Proof Sketch

4. Computing Edit Distance
   - Comparison with Previous Work
   - Proof Sketch

5. Conclusion

Diptarka Chakraborty, diptarka@cse.iitk.ac.in

Streaming algorithms for embedding and computing edit distance
Conclusion

Further Improvements:

- Building on the same technique one can device a deterministic streaming algorithm that takes $O(n + k^4)$ time and $O(k^4)$ space
Further Improvements:

- Building on the same technique one can device a deterministic streaming algorithm that takes $O(n + k^4)$ time and $O(k^4)$ space.
- Recently using a completely different approach we have achieved $O(n + k^2)$ time and $O(k)$ space bound.
Further Improvements:

- Building on the same technique one can device a deterministic streaming algorithm that takes $O(n + k^4)$ time and $O(k^4)$ space
- Recently using a completely different approach we have achieved $O(n + k^2)$ time and $O(k)$ space bound

Open Problem:

- Improving distortion factor of randomized embedding
- More specifically achieving $o(\log n)$ distortion factor
Acknowledgements

- Received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement n. 616787
- DC was partially supported by Research-I Foundation, Microsoft Research India Travel Grant and Google India Student Travel Grant
- MK was partly supported by the project 14-10003S of GA ČR.
THANK YOU!!!