Edit distance

středa 8. října 2025 20:53

Σ ... alphabet $\quad x, y \in \Sigma^*$

edit distance of x & y: $ED(x,y)$ = # of insertions / deletions / substitutions needed to transform x into y.

variants: • $HAM(x,y)$ = # of substitutions needed to transform x to y

• $INSDEL(x,y)$ = # of insertions / deletions needed to transform x into y.

Ex: x = TEACHER, y = CHEATED — $ED(x,y) = 4$, $HAM(x,y) = 5$, $INSDEL(x,y) = 6$

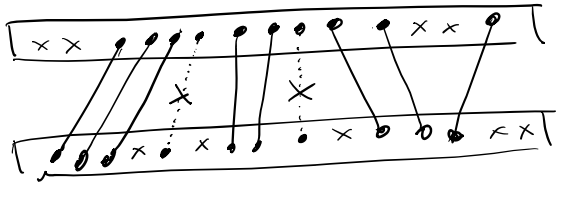
$x = 010101 \cdots 1$, $y = 101010 \cdots 0$ — $ED(x,y) = 2$, $HAM(x,y) = n$, $INSDEL(x,y) = 2$, $n = |x| = |y|$

Further variants: weighted ED, copy/move block

↳ different substitutions have different cost

insertions/deletions

Alignment / matching:



matching equal symbols in the optimal alignment

Prop.: $ED(x,y) = \min_i \; ED(x_1 \cdots x_{\lceil \frac{n}{2} \rceil}, y_1 \cdots i) + ED(x_{\lfloor \frac{n}{2} \rfloor} \cdots x_n, y_{i+1} \cdots n)$

$x = x_{1 \cdots n}$

$y = y_{1 \cdots n}$

$x_{i \cdots j} = \varepsilon$ for $i > j$

$x_{i\cdots j} = \varepsilon$ for $i > j$

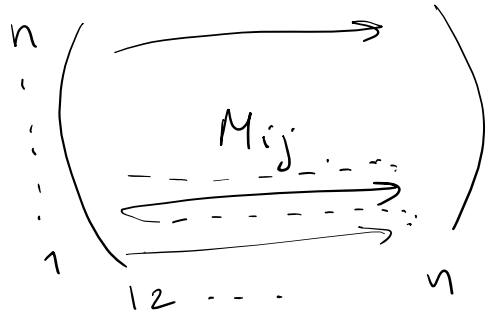
2) $ED(ux, uy) = ED(x, y) = ED(xu, yu)$.

→ time $O(n^5)$ ... $\forall i \le j, k \le \ell$ compute $ED(x_{i\cdots j}, y_{k\cdots \ell})$.

Prop: $ED(x_{1\cdots i}, y_{1\cdots j}) = \min \begin{cases} 1 + ED(x_{1\cdots i-1}, y_{1\cdots j}) & \text{del } x_i \\ 1 + ED(x_{1\cdots i}, y_{1\cdots j-1}) & \text{del } y_j \\ 1 + ED(x_{1\cdots i-1}, y_{1\cdots j-1}) & x_i \neq y_j \\ ED(x_{1\cdots i-1}, y_{1\cdots j-1}) & x_i = y_j \end{cases}$

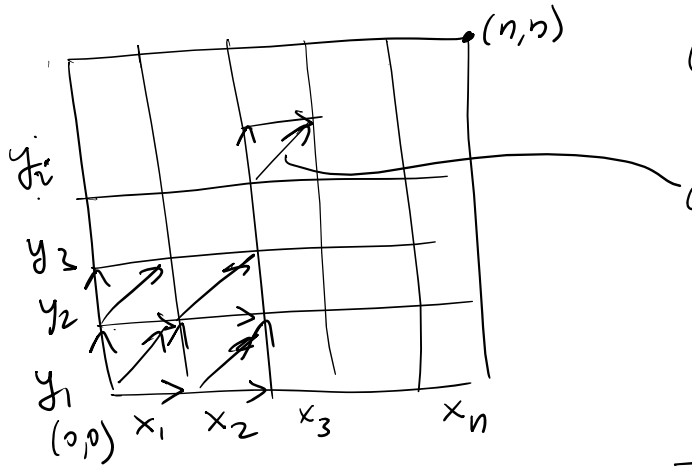
→ alg for $ED(x, y)$: for $j = 0$ to n, for $i = 0$ to n compute $ED(x_{1\cdots i}, y_{1\cdots j})$

Edit distance matrix M: $n \times n$ matrix



$M_{ij} = ED(x_{1\cdots i}, y_{1\cdots j})$

Edit distance graph $G_{x,y}$:



$(n+1) \times (n+1)$ grid

cost of edge: diag. ↗ $(i-1, j-1) \to (i, j)$ is $\begin{cases} 1 & x_i \neq y_j \\ 0 & x_i = y_j \end{cases}$

→ horizontal, ↑ vertical cost 1.

any path from $(0,0)$ to (n,n) corresponds to a sequence of operations

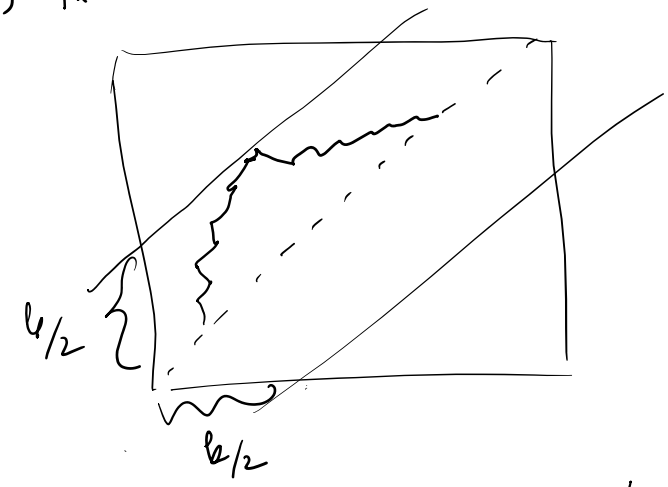
any path from $(0,0)$ to (n,n) corresponds to a sequence of operations:

horizontal edge ... delete x_i
vertical edge ... delete y_j
diagonal edge ... match/substitute x_i to y_j

Prop: $ED(x,y)$ = cost of shortest path from $(0,0)$ to (n,n).

$k \overset{\text{def}}{=} ED(x,y)$

- alg for $ED(x,y)$ in time $O(n \cdot k)$:



cost at most k to reach diagonal $-\frac{k}{2}$ or $\frac{k}{2}$ (and back)

for $k = 1, 2, 4, \ldots 2^i \ldots$ check shortest path between $(0,0)$ & (n,n) restricted to diagonals $-k, \ldots, k$.

$O(n \cdot k)$ size graph

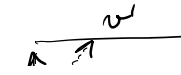
Prop.: For $v \in V(G_{x,y})$, let cost c_v = distance of v from $(0,0)$.

1) $\forall u, v$: $u \& v$ connected by an edge $\rightarrow |c_u - c_v| \leq 1$

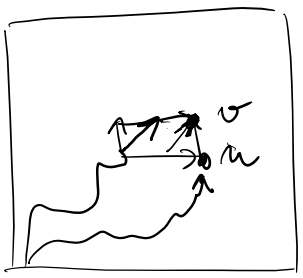
2) $\forall u, v$: if $u \to v$ connected by an diagonal edge $\rightarrow c_u \leq c_v$.

Pf: 1) 

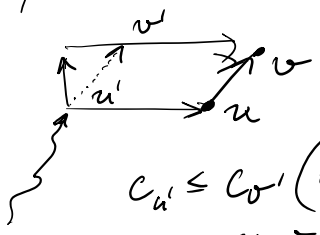
2) For contradiction. take u, v with smallest $c_u > c_v$



If: 1)



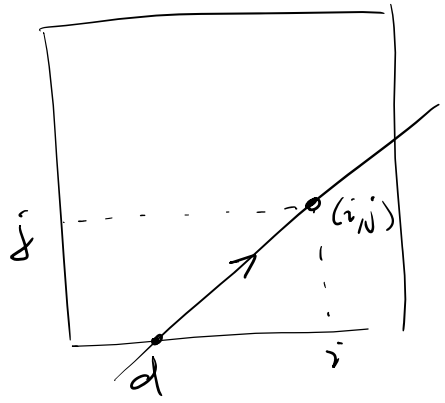
2) take u, v with smallest $c_u - c_v$



$c_{u'} \le c_{v'}$ (otherwise u, v are not minimal)

$\Rightarrow c_u \le c_v$. X

→ cost of diffing is monotone non-decreasing along each diagonal

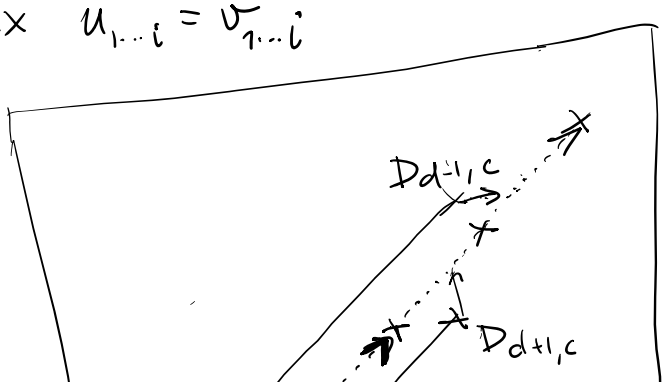


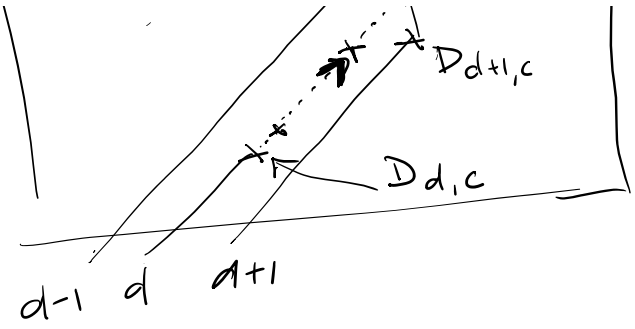
diagonal $d = i - j$

$d \in [-k, k]$, define $D_{d,c}$ = furthest j s.t. $c_{(j+d,j)} \le c$.

(*)
$$D_{d,c+1} = \min \begin{cases} 1 + LCP(x_{d+1+D_{d,c}} \cdots n, y_{1+D_{d,c}} \cdots n) \\ LCP(x_{d+D_{(d-1),c}} \cdots n, y_{D_{d-1,c}} \cdots n) \\ 1 + LCP(x_{d+1+D_{(d+1),c}} \cdots n, y_{D_{d+1,c}} \cdots n) \end{cases}$$

$$LCP(u,v) = \max_i \; u_{1\cdots i} = v_{1\cdots i}$$





→ alg for $ED(x,y)$: compute $D_{d,c}$ for $d \in [-k,k]$, $c \in \{0..k\}$
using (*)
check if we reach (n,n)
(i.e. $D_{0,k} \geq n$.)

→ $O(k^2)$ LCP queries for sufixes of x & y.

- Claim: in time $O(n)$ we can preprocess x & y so that LCP queries on sufixes of x & y can be answered in time $\tilde{O}(1)$.

- $\tilde{O}(1) = \text{poly-log}(n)$. (probabilistic preprocessing, all queries will be answered correctly w. prob $\geq 1 - \frac{1}{n}$.)

Karp-Rabin fingerprint:

prime $p \simeq n^{10}$, GF_p ... Finite Field of size p
(=computing mod p)

For $x, y \in GF_p$ pick $g \in GF_p \setminus \{0\}$ at random.

Prepare three arrays: $G[]$ | g^1 | g^2 | ... | g^n |

$F_x[]$ |

$F_y[]$ |

$$F_x[\ell] = \sum_{i=1}^{\ell} x_i \cdot g^i \qquad F_y[\ell] = \sum_{i=1}^{\ell} y_i \cdot g^i$$

to check $x_{i+1} \dots i+\ell = y_{j+1} \dots j+\ell$

test whether $g^{j-i} \cdot (F_y[j+\ell] - F_y[j]) =$ (**)

$= F_x[i+\ell] - F_x[i]$

→ use binary search to find maximum ℓ s.t.

$x_i \dots x_{i+\ell} = y_j \dots j+\ell$

to answer $LCP(x_{i..n}, y_{j..n})$.

Prop: if $x_{i+1} \dots i+\ell = y_{j+1} \dots j+\ell$ then (**) is always true.
if $x_{i+1} \dots i+\ell \neq y_{j+1} \dots j+\ell$ then (**) w.p. $\leq \frac{n}{p} = \frac{1}{n^9}$.

(two distinct polynomials of degree $\leq n$ agree on at most n points)

By union bound over choice of i, j, ℓ

the probability that (**) gives correct answer for all substrings of x & $y \geq 1 - \frac{n^3}{n^9} \geq 1 - \frac{1}{n}$.

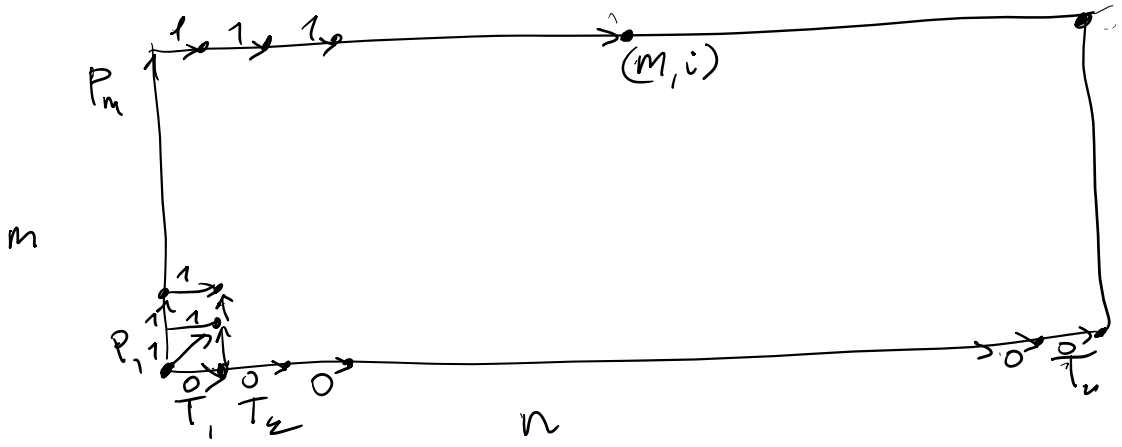
(in this case the binary search always finds correct LCP value.)

(in this case the binary [illegible] finds correct LCP value.)

Approximate Pattern Matching upto k-edits

pattern P, text T $\quad |P|=m$ & $|T|=n$

Goal: Find all $i \in [n]$ s.t. $\min_{j<i} ED(P, T_{j..i}) \leq k$



Take the E.D. graph of P & T and give 0 cost to the bottom horizontal edges

Claim: cost of $(0,0) \to (m,i) = \min_j ED(P, T_{j..i})$

→ want to find all points (m,i) where the cost $\leq k$.

Solution: For each diagonal in the graph determine the farthest point reachable for cost $\leq k$.
- use the same algorithm as for the $O(n+k^2)$ ED algorithm

→ for each of the n diagonals, needs to make $3k$ LCP queries

→ $\tilde{O}(n \cdot k)$ time to execute the algorithm.

→ $\tilde{O}(n \cdot k)$ time to execute the algorithm.