Why is it hard to beat $\mathcal{O}(n^2)$ for Longest Common Weakly Increasing Subsequence?

Adam Polak

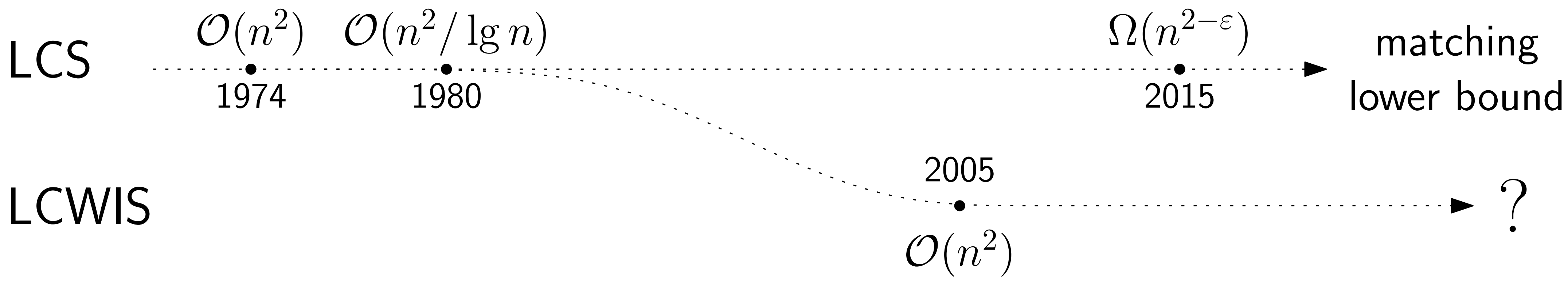
THEORETICAL COMPUTER SCIENCE
Jagiellonian University

Longest Common Weakly Increasing Subsequence (LCWIS) – younger brother of classic Longest Common Subsequence (LCS)

Input: Two integer sequences A, B
Output: Sequence C such that

- it is a subsequence of both A and B,
- it is weakly increasing,
- its length is maximum possible.

Example: $A = \mathbf{1}2\mathbf{4337}$
$B = \mathbf{143}36\mathbf{7}$
$\mathrm{LCWIS}(A, B) = 4$



Why is it hard to beat $\mathcal{O}(n^2)$ for LCWIS?

LCWIS in $\mathcal{O}(n^{2-\varepsilon})$ time $\implies$ SETH is false
Proof Via reduction from Orthogonal Vectors Problem

Strong Exponential Time Hypothesis (SETH)

CNF-SAT on N variables cannot be solved in $\mathcal{O}((2-\varepsilon)^N)$

Orthogonal Vectors Problem (OVP)

Input: Two sets U, V of d-dimensional $(0,1)$-vectors, $|U| = |V| = n$
Output: Is there $u \in U$, $v \in V$ such that $u \cdot v = 0$?

OVP in $\mathcal{O}(n^{2-\varepsilon}\mathrm{poly}(d))$ time $\implies$ SETH is false (Williams, 2005)

Alignment gadget framework

If a problem *admits an alignment gadget* it cannot be solved in $\mathcal{O}(n^{2-\varepsilon})$ unless SETH fails (Bringmann, Künnemann, 2015)

- Gives lower bounds for LCS, Edit Distance, and many similar problems
- Does not seem to work for LCWIS

Weighted LCWIS

- Auxiliary problem to simplify reduction
- Weight function $w : \Sigma \to \mathbb{N}_+$
- Instead of length, maximize total weight of elements of subsequence
- Equivalent to computing unweighted LCWIS for sequences with each symbol σ appearing $w(\sigma)$ times

Coordinate gadgets and vector gadgets

$$U \ni u = u[1]u[2]\dots u[d] \qquad V \ni v = v[1]v[2]\dots v[d]$$

$$\mathrm{CG}_1(0,i) = \langle 3i, 3i+1\rangle \qquad \mathrm{CG}_2(0,i) = \langle 3i, 3i+2\rangle$$
$$\mathrm{CG}_1(1,i) = \langle 3i+2\rangle \qquad \mathrm{CG}_2(1,i) = \langle 3i+1\rangle$$

$$\mathrm{LCWIS}(\mathrm{CG}_1(u[i],i), \mathrm{CG}_2(v[i],i)) = \begin{cases} 0, & \text{if } u[i]=1 \text{ and } v[i]=1, \\ 1, & \text{otherwise} \end{cases}$$

$$\mathrm{VG}_1(u) = \mathrm{CG}_1(u[1],1)\ \mathrm{CG}_1(u[2],2)\ \dots\ \mathrm{CG}_1(u[d],d)$$
$$\mathrm{VG}_2(v) = \mathrm{CG}_2(v[1],1)\ \mathrm{CG}_2(v[2],2)\ \dots\ \mathrm{CG}_2(v[d],d)$$

$$\mathrm{LCWIS}(\mathrm{VG}_1(u), \mathrm{VG}_2(v)) = d - (u \cdot v)$$

Gluing vector gadgets together

Four new symbols: $\mathrm{A} < \mathrm{B} <$ any symbol in $\mathrm{VG} < \mathrm{Y} < \mathrm{Z}$

$$w(\mathrm{A}) = w(\mathrm{Z}) = 2d \quad w(\mathrm{B}) = w(\mathrm{Y}) = 4d$$

$$U = \{u_1, u_2, \dots, u_n\} \quad V = \{v_1, v_2, \dots, v_n\}$$

$$P_1 = \mathrm{A}^{2n}\ \mathrm{VG}_1(u_1)\ \mathrm{YB}\ \mathrm{VG}_1(u_2)\ \mathrm{YB}\ \dots\ \mathrm{YB}\ \mathrm{VG}_1(u_n)\ \mathrm{Z}^{2n}$$
$$P_2 = (\mathrm{ZYBA})^n\ \mathrm{VG}_2(v_1)\ \mathrm{ZYBA}\ \mathrm{VG}_2(v_2)\ \mathrm{ZYBA}\ \dots\ \mathrm{ZYBA}\ \mathrm{VG}_2(v_n)\ (\mathrm{ZYBA})^n$$

$$\mathrm{WLCWIS}(P_1, P_2) = \max_{1 \leqslant i,j \leqslant n} d - (u_i \cdot v_j) + const$$

Solving OVP for n vectors of dimension d reduced to finding LCWIS of two sequences of length $\mathcal{O}(nd)$

Alphabet size

- LCS: quadratic time hard even for binary alphabet
- LCWIS: linear time algorithm for 3-letter alphabet
  hardness reduction using $\log n$ size alphabet
- Open problem: close the gap

Longest Common Increasing Subsequence (LCIS)

- Yet another variant – substitute *weakly* with *strictly* in problem definition
- Virtually identical $\mathcal{O}(n^2)$ algorithm works
- Above reduction does not work
- New, more involved construction required – coming soon!

`https://arxiv.org/abs/1703.01143` `polak@tcs.uj.edu.pl`