Finding Diverse Strings and Longest Common Subsequences in a Graph


Yuto Shida[1]

Giulia Punzi[2,3]

Yasuaki Kobayashi[1]

Takeaki Uno[2]

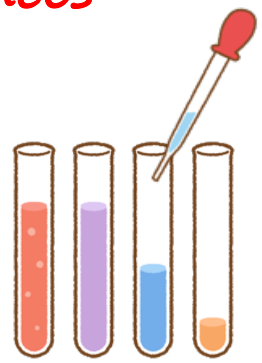
Hiroki Arimura[1]

1) Hokkaido University, Sapporo, Japan
2) National Inst. Informatics, Tokyo, Japan
3) University of Pisa, Pisa, Italy

This work is partly supported by MEXT Grant-in-Aid for Transformative Research Areas (A) JP20H05963 AFSA, "Innovative algorithmic creation"; JP 20A402 Grant-in-Aid for Basic Research A.

Backgrounds

- **A Classic problem: Longest Common Subsequence (m-LCS)**
  - The problem of finding one of the longest (non-contiguous) subsequences common to all M input strings (LCS).
  - One of the most fundamental problems in computer science and bioinformatics.
  - It has been studied for over 50 years in theory and applications.



Longest Common Subsequence (LCS)

A longest common subsequence (LCS) of a set S of input strings is a (non-contiguous) subsequence common to all of m inputs strings.

The set S of m=2 input strings

X = ABABCDDEE

Y = ABCBAEEDD

Common subsequences

ε, A,B,C,D,E

AA, AB, AC, AD, AE, BA, ..., CD, CE, DD, EE,

ABA, ABB, ABC, ABD, ..., CEE,

ABAD, ABAE, ABBD, . . . , BCEE,

ABADD, ABAEE, ABBDD,

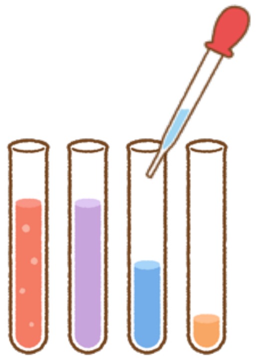
ABBEE, ABCDD, ABCEE

Six LCSs of S with length r=5

Shida, Punzi, Kobayashi, Uno, and Arimura, 2024

Backgrounds

- A Classic problem: Longest Common Subsequence (m-LCS)
  - The problem of finding one of the longest (non-contiguous) subsequences common to all M input strings (LCS).
  - One of the most fundamental problems in computer science and bioinformatics.
  - It has been studied for over 50 years in theory and applications.

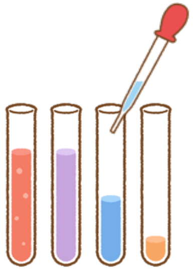


- Computational complexity of m-LCS:
  - Polynomial-time solvable if M is a constant (Irving & Fraser, CPM'92), while it is NP-hard if M is an input.
  - W[t]-hard if M is a parameter, and W[2]-hard if L is a parameter (Bodlaender, Downey, Fellows, & Wareham, TCS, 1995).
  - FPT by other parameterization (Bulteau, Jones, Niedermeier+, CPM'22)
- Our goal: We introduce the diversity maximization problem for LCSs, and study its computational complexity (approximability & parameterized complexity)

Motivations: Finding Multiple Diverse Solutions

- In combinatorial optimization, much effort has been done for finding a single best solution.
  - **Examples:** Drag discovery, route planning in delivery networks, factory automation, etc.

- However, there has been growing interest in finding multiple diverse solutions in optimization problems



- Reasons:
  - The specification may not be perfect
  - There can be too many optimal solutions (algorithm-dependent)
  - Human experts may want to intervene ("Human-in-the-Loop")
- Diversity maximization problem attracts much attension

Motivations: Finding Multiple Diverse Solutions

To find diverse solutions

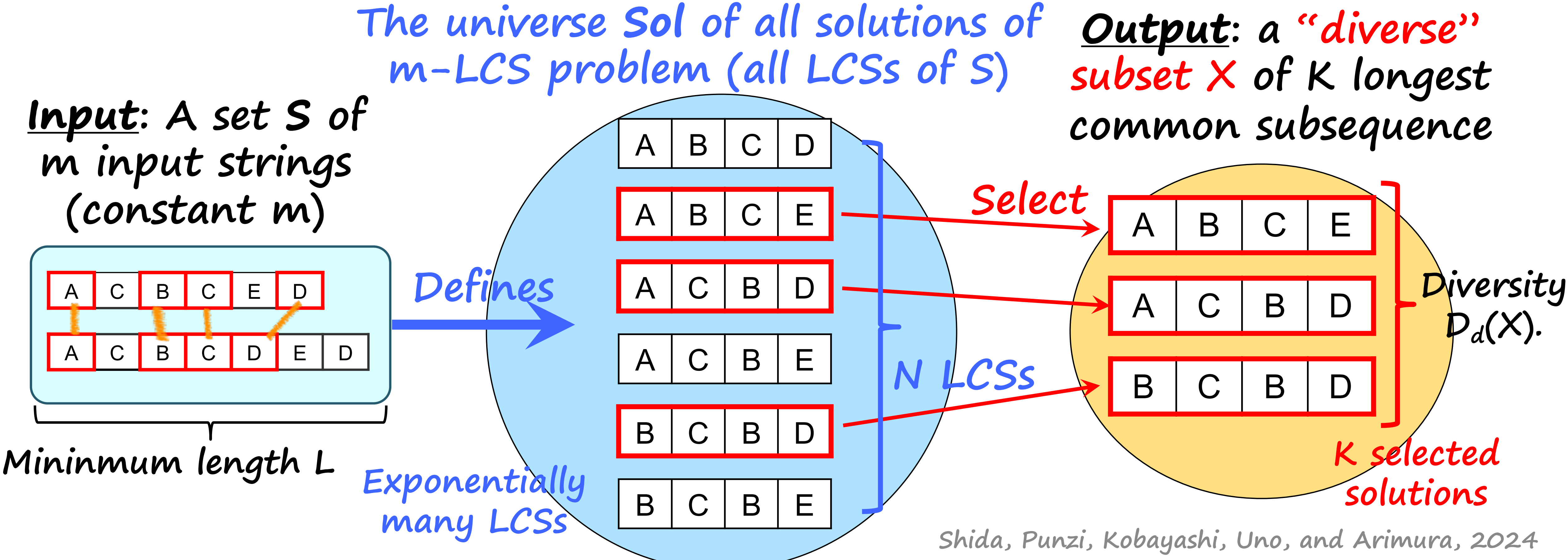
- There has been a variety of methods studied in the past:

• Random generation	– Generate solutions randomly.
• Enumeration	– Generate solutions exhaustively.
• Top-K search	– Generate in decreasing order of objectives

- However, any of these methods are not satisfactory from the view point of (i) the size of a solution set and (ii) the explicit guarantee of the diversity
- Our goal: We study the computational complexity of diversity maximization problem for LCSs

Our problem: Diversity Maximization

Given a set S of m input strings, find a subset X of K longest common subsequence among all of N solutions of the m-LCS problem that maximizes a specified diversity measure $D_d(X)$.



Def. Distance between strings

Def. The *Hamming distance* $d_{HD}(X, Y)$ between two strings X and Y of the same length n is *the total number of positions at which the strings disagree* (differ).

$$d_{HD}(X, Y) = \sum_{i=1}^{n} \mathbb{1}\{X[i] \neq Y[i]\}$$

	1	2	3	4	5
X	A	B	A	D	D
Y	A	B	C	D	E

$$d_{HD}(X, Y) = 2$$

Def. Diversity Maximization

Def. Diversity measures for K-solution sets $X = \{x_1, \dots, x_K\} \subseteq Sol$.

- **Sum-Diversity** $D_d^{\mathrm{sum}}(X)$: the sum of the pairwise distance over all pairs in X.

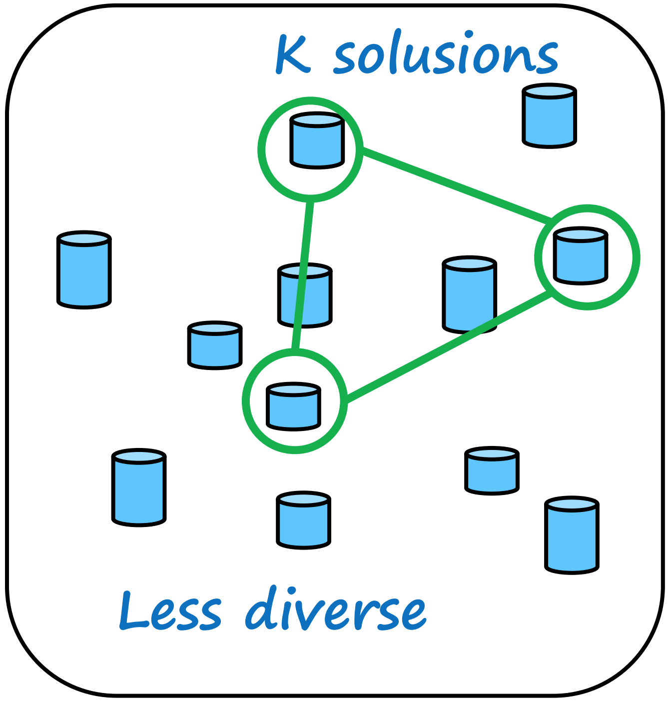
$$D_d^{\mathrm{sum}}(X) \coloneqq \sum_{i<j} d(x_i, x_j),$$

- **Min-Diversity** $D_d^{\min}(X)$: the minimum of the pairwise distance over all pairs in X

$$D_d^{\min}(X) \coloneqq \sum_{i<j} d(x_i, x_j)$$

A set Sol of optimal solutions

K = 3



Def. Diversity Maximization

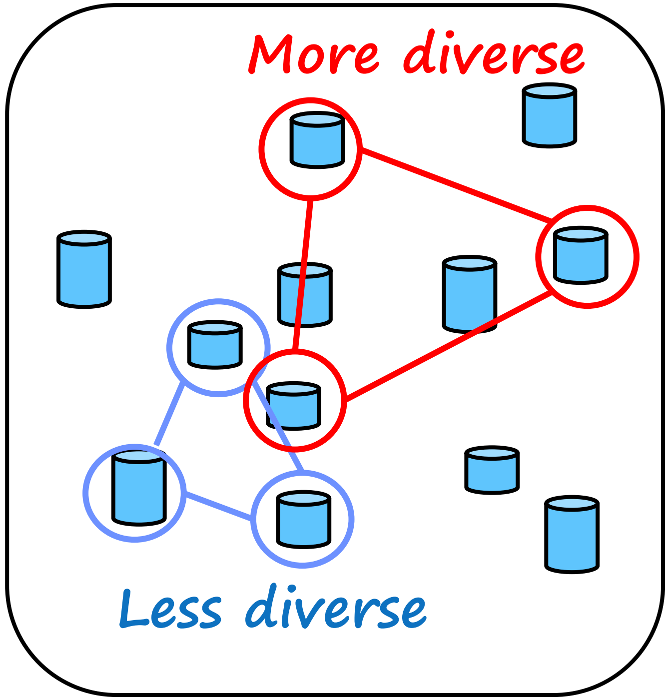
Sum-Diversity Maximization Problem (Max-Sum Divese(LCS))

Given: A set of solutions Sol, distance function $d: Sol^2 \to \mathbb{R}_+$ integers $K \geq 1, \Delta \geq 0$

Task: Find a subset $X = \{x_1, \ldots, x_K\} \subseteq Sol$ of K solutions such that
(1) $|X| \leq K$, and (2) $D_d^{\mathrm{sum}}(X) \geq \Delta$ ($D_d^{\mathrm{min}}(X) \geq \Delta$).

A set Sol of optimal solutions

K = 3



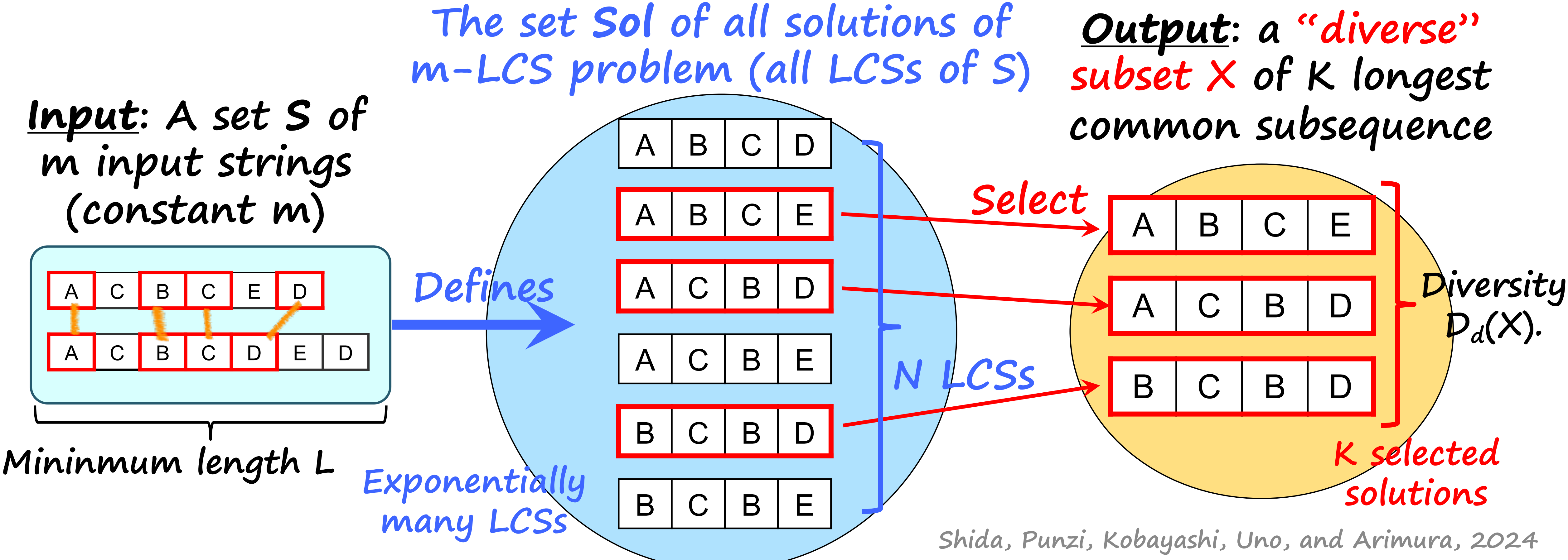
Similarly, we can define the Max-Min variant (Max-Min(LCS))

$D_d^{\mathrm{sum}}(X) := \sum_{i<j} d(x_i, x_j)$ **Sum-Diversity**

$D_d^{\mathrm{min}}(X) := \sum_{i<j} d(x_i, x_j)$ **Min-Diversity**

Our problem: Diversity Maximization

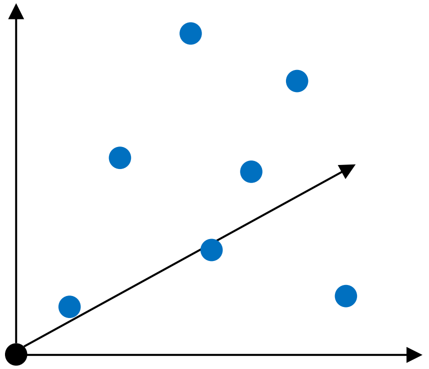
Given a set S of m input strings, find a subset X of K longest common subsequence among all of N solutions of the m-LCS problem that maximizes the specified diversity measure $D_d(X)$.



Related Work

In 1970s, early days: K-dispersion problem

- Selecting a diverse set of K points among N points in a metric space
- $O(N^K t_{\text{dist}})$ time : try all K combinations among N



In 2020s: "Diverse-X program"

- proposed by Michael R. Fellows (Dagstuhl seminar, 2019)

> Let's study the complexity of finding diverse solutions in various discrete optimization problems X!

Michael R. Fellows
U. Bergen, Norway

- Examples of X = MSTs, Matchings, Shortest paths, etc.
  - Typically, there are exponentially many solutions.
- NP-hard in most case. Sometimes, FPT or Approximable

Our goal: We study the computational complexity of diversity maximization problem in strings, especially LCSs

Baste, Fellows+, Dagstuhl seminar 18421 "Algorithmic enumeration", 2019. Also in Artificial Intelligence, 303:103644, 2022.

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

Polynomial time computable

Proof: Dynamic programming

When K is input

NP-hard, but

admits PTAS (Min-Sum only)

Approximable within any constant error

Proof: Local search (Cevallos+ 2019) + DP + Negative type metric

Parameterized by K and r

FPT

Proof: Color coding

Parameterized by K

W[1]-hard

Proof: FPT-reduction from p-CLIQUE

K : #solutions to select, r : max. length of LCS

Shida, Punzi, Kobayashi, Uno, and Arimura, 2024

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

Polynomial time computable

Proof: Dynamic programming

When K is input

K : #solutions to select, r : max. length of LCS

Result 1: When the #solutions K is bounded

Main result of Sec.3 (upperbound)

Assumption: the number of input strings m≥2 is a constant, throughout

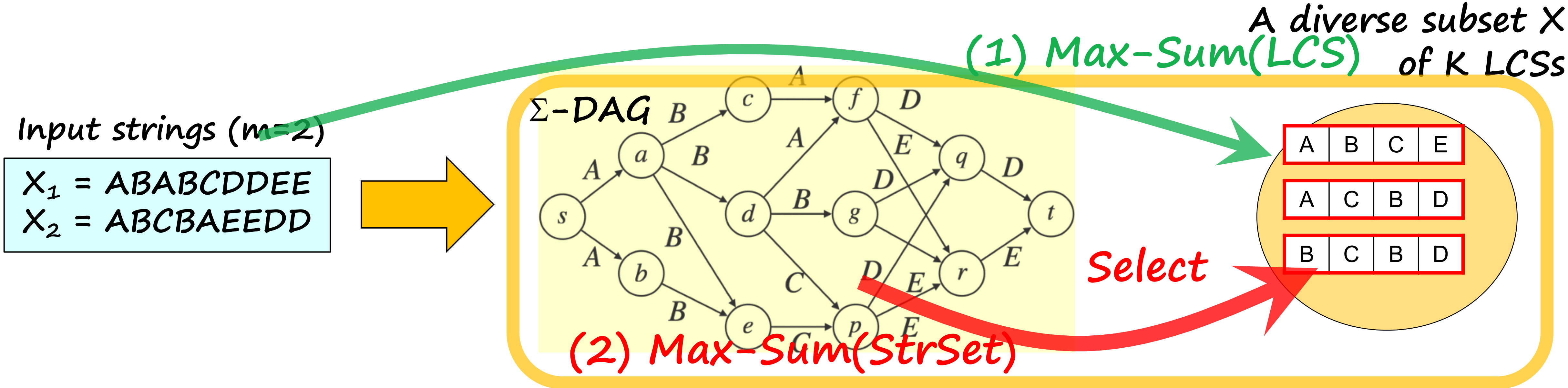
- **Corollary:** When K is a constant, Max-Sum(LCS) is solvable in polynomial time.
- The same result holds for Max-Min.

Our approach: First show a more general result (Thm 6), solve it using dynamic programming on a DAG, and then apply it to the Corollary.

- **Theorem 6** (Max-Sum(StrSet)):
  When K is a constant, for any set L of strings of equal length, and given a Σ-labeled acyclic directed graph (Σ-DAG) representing it, Max-Sum(StrSet) can be solved in polynomial time.

Result 1: When the #solutions K is bounded

Original problem to consider:
(1) Max-Sum(LCS): the Max-Sum diverse LCSs problem



- **Observation (folklore)**: The set of all (exponentially many) LCSs can be stored in a polynomial-sized DAG G (called a Σ-DAG).

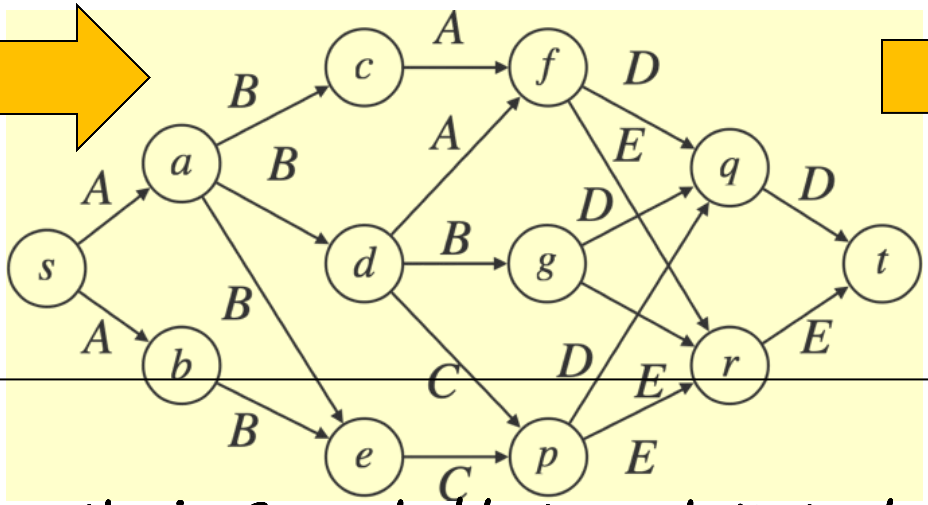
Hence, we consider a more general problem instead: (2) Max-Sum(StrSet): the Max-Sum diverse String Set problem for Σ-DAGs of equi-length strings

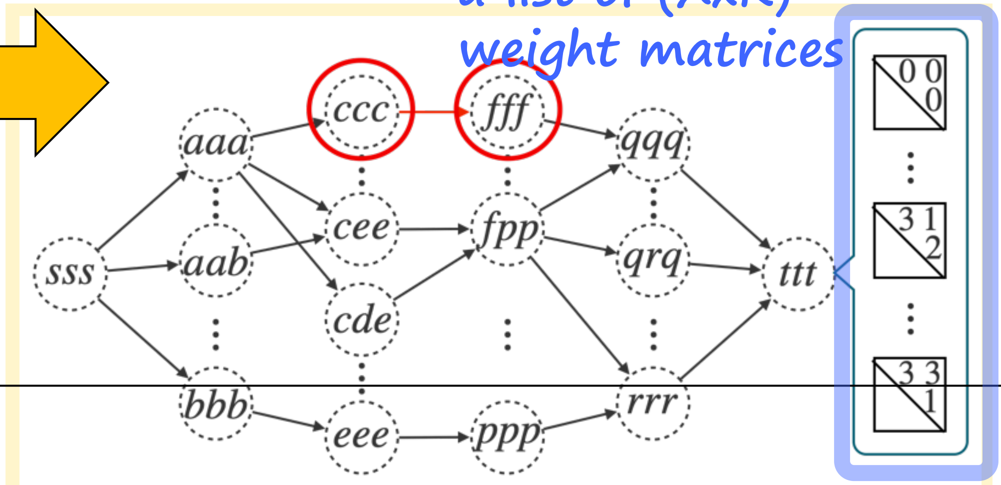
Result 1: When the #solutions K is bounded

Polynomial-time Algorithm: Performs dynamic programming over all K-tuples of vertices of G to collect the set of (KxK)-weight matrices for all combinations of K paths that maximize the diversity measure.

Input strings (m=2)

X_1 = ABABCDDEE
X_2 = ABCBAEEDD





Details:

- A combination of K paths in G reachable to each K-tuple of vertices uniquely determines a set of K prefixes of solution strings, and thus, the associated (KxK)-weight matrice $M = (W_{ij})$,
  - where the weight W_{ij} is defined by the pairwise Hamming distances $d(P_i, P_j)$ between string labels of two paths P_i and P_j).
- To each K-tuple of vertices, maintain the list of all possible weight matrices

Observation: There are at most $(\Delta+1)^{K\times K}$ (polynomially many) distinct weight matrices for constant K and Δ.

Shida, Punzi, Kobayashi, Uno, and Arimura, 2024

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

When K is input

NP-hard, but

admits PTAS (Min-Sum only)

Approximable within any constant error

Proof: Local search (Cevallos+ 2019) + DP + Negative type metric

K : #solutions to select, r : max. length of LCS

Result 2: When #solutions K is unbounded

- **Theorem 3 (Hardness):** When K is an input, the Max-Sum maximum diversity LCSs problem is
  - NP-hard. (Even if # of input strings is a constant m=2).
  - W[1]-hard if K is a parameter.
- The same result holds for the Max-Min variant.

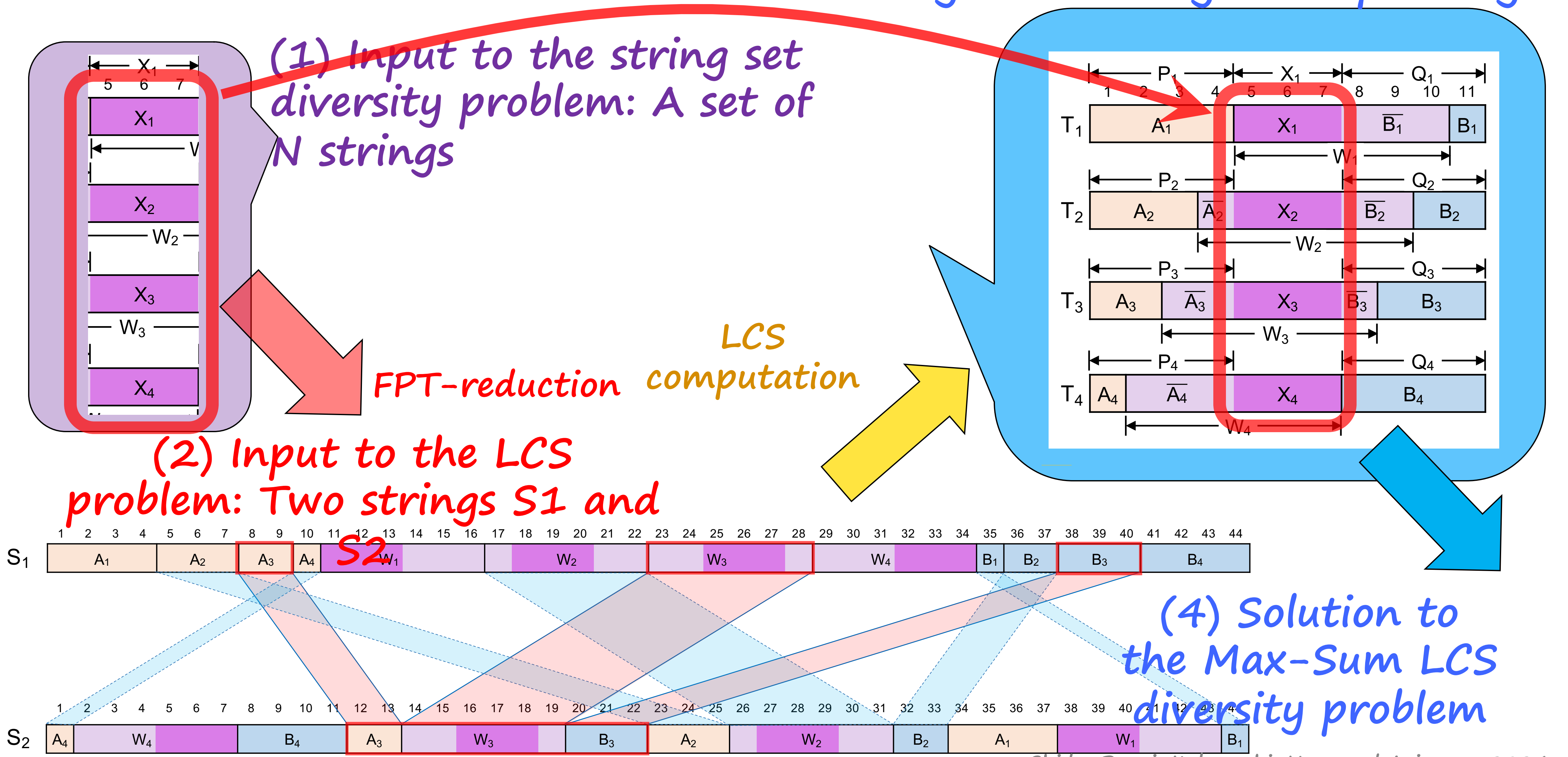
Approach: We show the NP-hardness of a more basic "string set diversity problem" and reduce it to the original problem.

- Lemma 1: When K is an input, Max-Sum(StringSet) is NP-hard, and W[1]-hard if K is a parameter. (Proof: reduction from p-clique)

The next lemma is a key.

- Lemma 2: Max-Sum(StringSet) is FPT-reducible to Max-Sum(LCS)

Proof of Lemma 2



Shida, Punzi, Kobayashi, Uno, and Arimura, 2024

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

When K is input

NP-hard, but

admits PTAS (Min-Sum only)

Approximable within any constant error

Proof: Local search (Cevallos+ 2019) + DP + Negative type metric

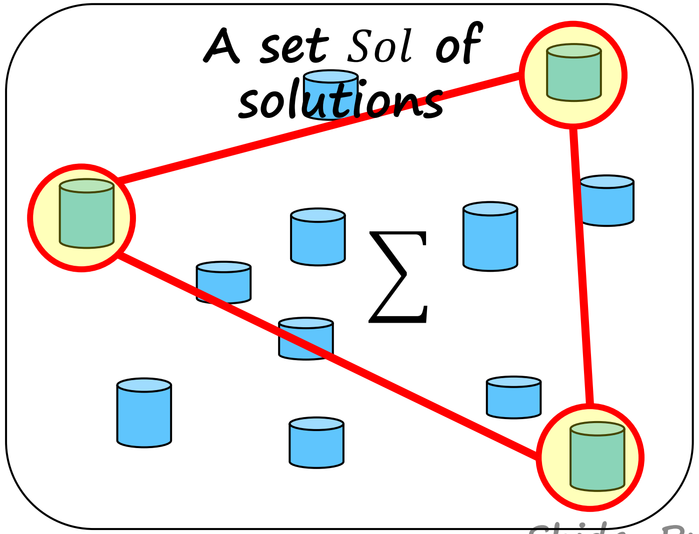
K : #solutions to select, r : max. length of LCS

Result 3: When #solutions K is unbounded: Approximability

- **Theorem 13:** When K is an input, the Max-Sum diversity problem for LCSs has a PTAS.
(i.e., for any constant ε>0, it is (1- ε)-approximable in polynomial time).
- Note: The same result does not hold for the Max-Min variant.

Sum-Diversity:

$$D_d^{\mathrm{sum}}(X) := \sum_{i<j} d(x_i, x_j),$$



Select K = 3 solutions

Proof of Theorem 13 (PTAS for unbounded K)

Theorem 13 follows from Theorem 10, Lemma 11, and Lemma 12 below.

- Theorem 10 (Cevallos, Eisenbrand, Zenklusen et al., 2019):
  When K is an input, the Max-Sum version of the maximum diversity problem for a point set has a PTAS if the condition (1) and (2) hold:
  - The distance function satisfies the "negative type inequality" (NEG).
  - The "farthest point problem" is polynomial time solvable.

Our case:

- A point = an LCS
- A distance function = Hamming distance

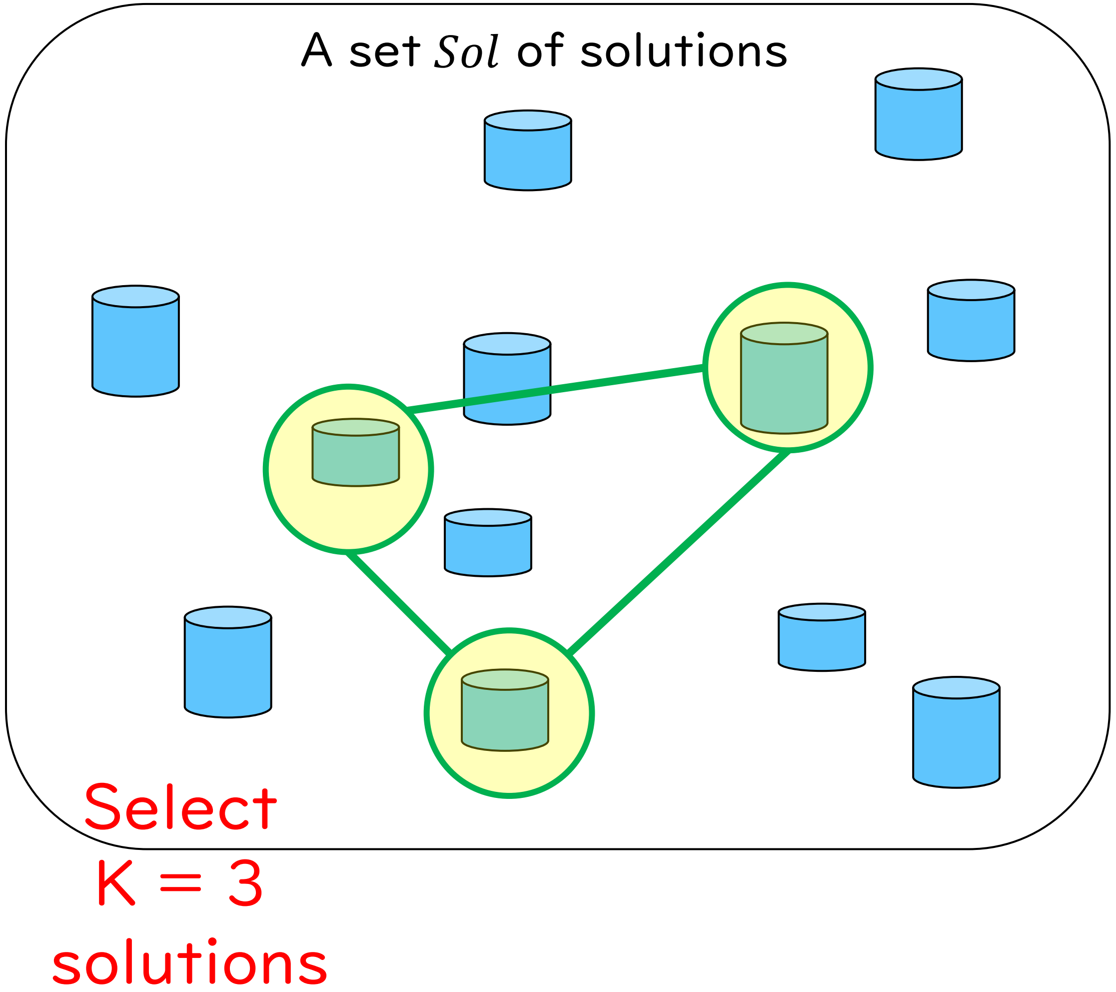
- **Lemma 11**: The Hamming distance between strings is a quasi-metric that satisfies the "negative type inequality".

  (Proof: Hamming distance on strings over Σ with length L can be embedded into L1-metric over bitvectors of length $|\Sigma| \cdot L$. Since (i) L1-metric satisfies NEG, and (ii) NEG is closed under positive linear combinations (i.e., a cone), Lemma follows.)

- **Lemma 12**: The farthest point problem for LCS can be solved in polynomial time.
  (Proof: In the DP-algorithm of Thm.6, fix K-1 strings, and search for the rest one.)

Alfonso Cevallos, Friedrich Eisenbrand, and Rico Zenklusen. "An improved analysis of local search for max-sum diversification," *Mathematics of Operations Research*, 44(4):1494–1509, 2019.

Result 3: When #solutions K is unbounded (PTAS)

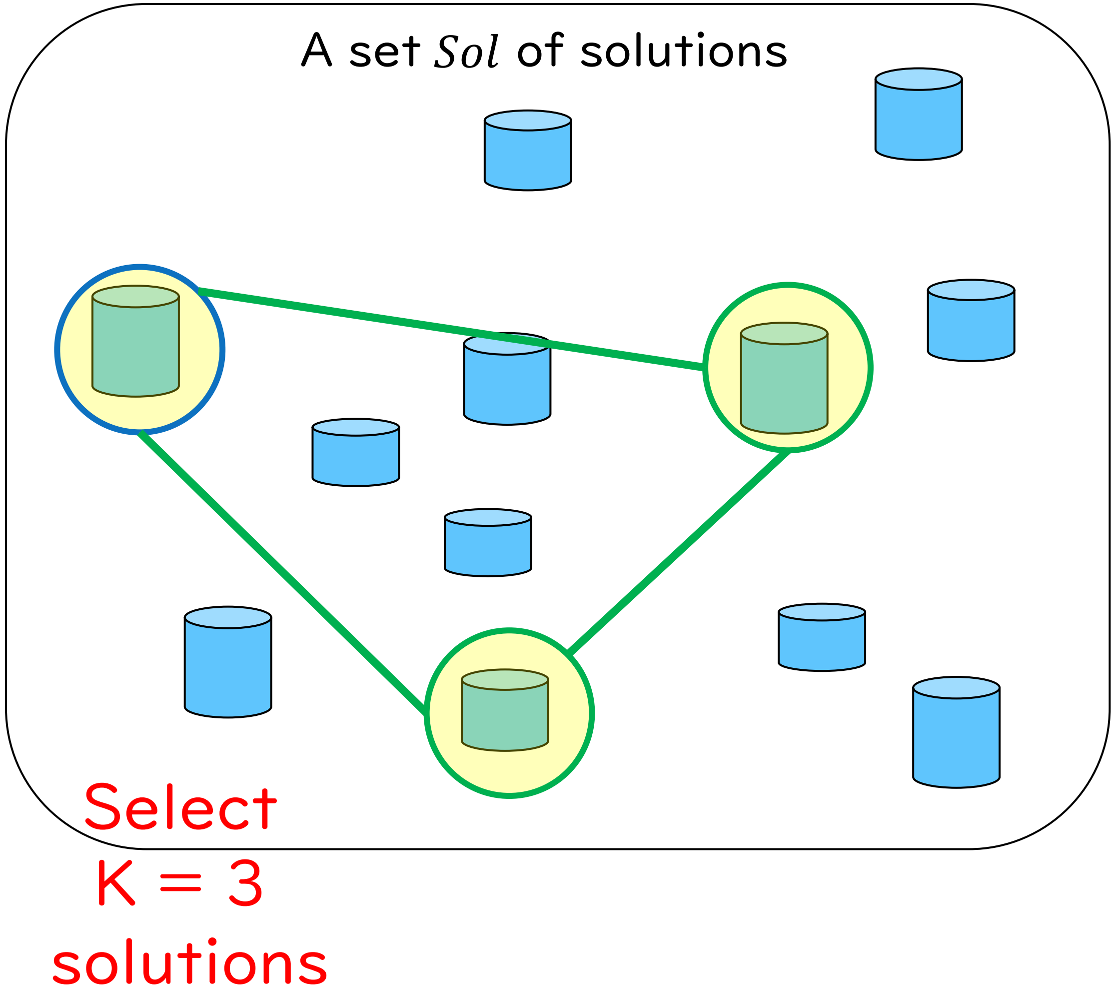


- Use the local search method below

Local search:

- Initially, select any solution set S = {X1, ..., XK}.
- Repeat the following process $O(K^2 \log K)$ times as long as the updated diversity $D((S-\{X\}) \cup \{Y\})$ increases:
  - Select a solution X from S
  - Replace X with a new solution Y
- Return the solution set S.

Result 3: When #solutions K is unbounded (PTAS)

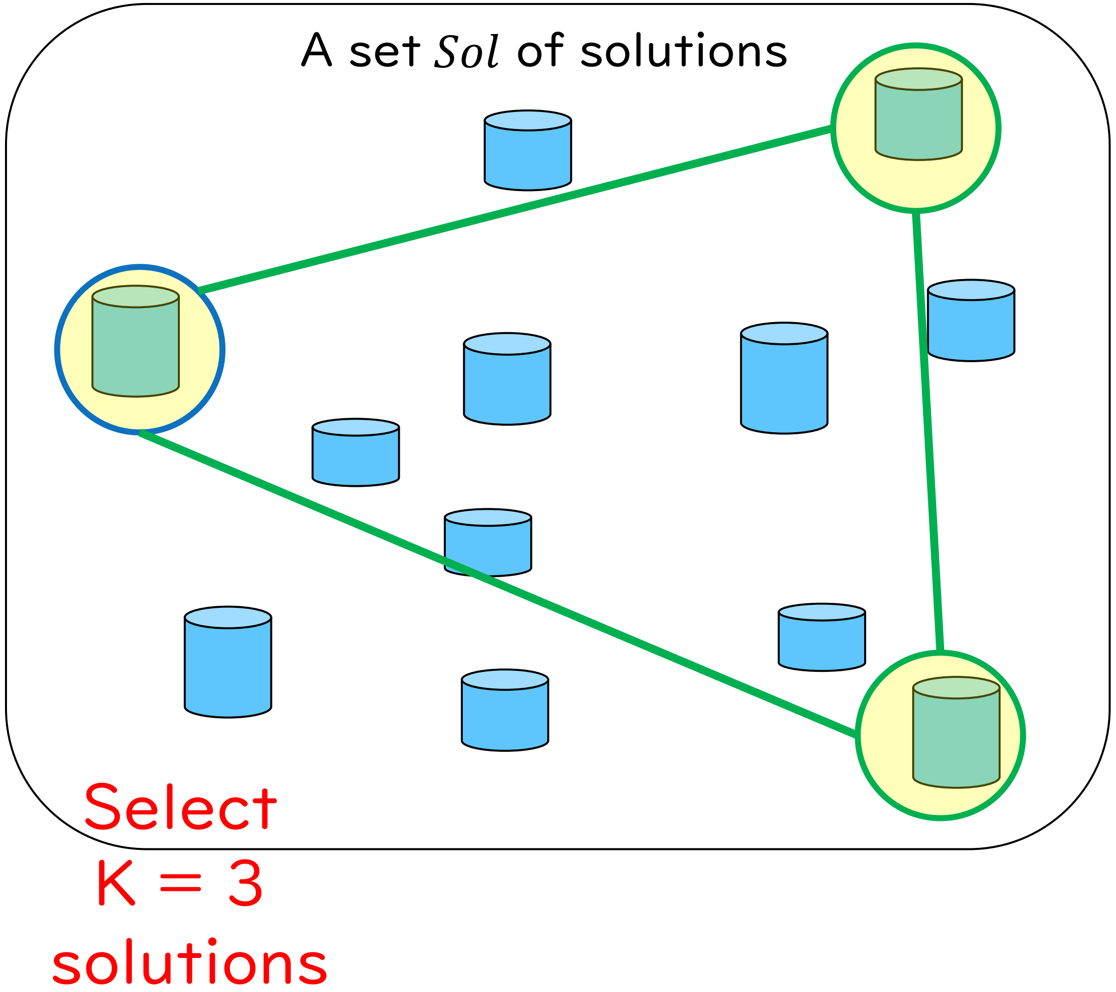


- Use the local search method below

Local search:

- Initially, select any solution set S = {X1, ..., XK}.
- Repeat the following process $O(K^2 \log K)$ times as long as the updated diversity D(S−{X})∪{Y}) increases:
  - Select a solution X from S
  - Replace X with a new solution Y
- Return the solution set S.

Result 3: When #solutions K is unbounded (PTAS)

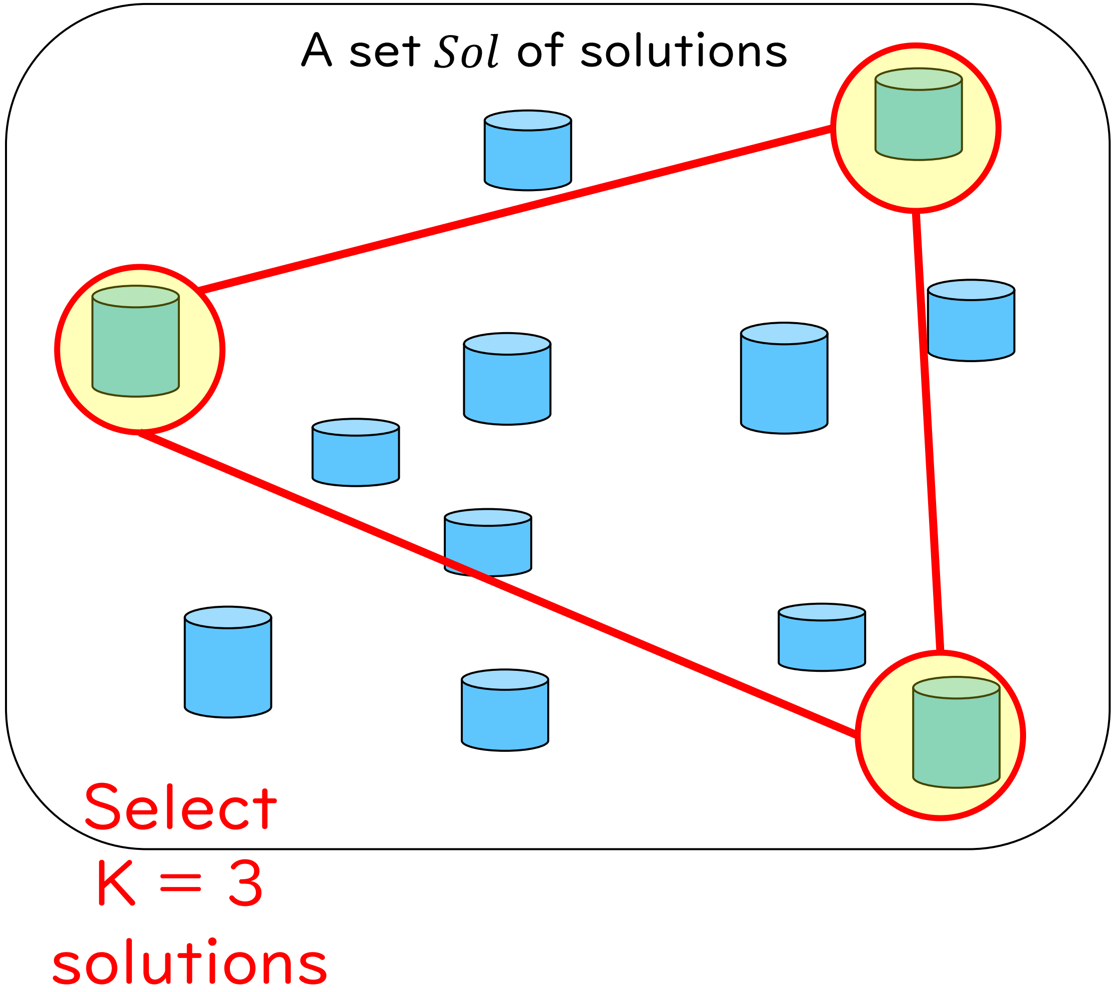


- Use the local search method below

Local search:

- Initially, select any solution set S = {X1, …, XK}.
- Repeat the following process $O(K^2 \log K)$ times as long as the updated diversity D(S−{X})∪{Y}) increases:
  - Select a solution X from S
  - Replace X with a new solution Y
- Return the solution set S.

Result 3: When #solutions K is unbounded (PTAS)



- Use the local search method below

Local search:

- Initially, select any solution set S = {X1, ..., XK}.
- Repeat the following process $O(K^2 \log K)$ times as long as the updated diversity D(S−{X})∪{Y}) increases:
    - Select a solution X from S
    - Replace X with a new solution Y
- Return the solution set S.

Result 3: When #solutions K is unbounded: Approximability

- **Theorem 13:** When K is an input, the Max-Sum diversity problem for LCSs has a PTAS.
  (i.e., for any constant ε>0, it is (1- ε)-approximable in polynomial time).
- Note: The same result does not hold for the Max-Min variant.

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

When K is input

Parameterized by K and r

FPT

Proof: Color coding

(This part will be skipped due to time constraint)

K : #solutions to select, r : max. length of LCS

Summary: Complexity of Max. Diverse LCSs

Notes: All results below except PTAS hold for the Min-Sum(LCS) & MinSum(StringSet)

Assumption: the number m of input strings is always constant, throughout.

When K is bounded

Polynomial time computable

Proof: Dynamic programming

When K is input

NP-hard, but

admits PTAS (Min-Sum only)

Approximable within any constant error

Proof: Local search (Cevallos+ 2019) + DP + Negative type metric

Parameterized by K and r

FPT

Proof: Color coding

Parameterized by K

W[1]-hard

Proof: FPT-reduction from p-CLIQUE

K : #solutions to select, r : max. length of LCS

Summary of Results

bounded K

unbounded K

Problem	Type	K: const	K: param	K: input
Max-Sum Diverse String & LCS	Exact	Poly-Time (Theorem 3.2)	W[1]-hard on Σ-DAG (Theorem 6.2)) W[1]-hard on LCS (Corollary 6.2))	NP-hard on Σ-DAG if $r \geqslant 3$:const (Theorem 6.1) NP-hard on LCS (Corollary 6.1)
	Approx. or FPT	—	FPT if r: param (Theorem 5.2)	PTAS (Theorem 4.2)
Max-Min Diverse String & LCS	Exact	Poly-Time (Theorem 3.1)	W[1]-hard on Σ-DAG (Theorem 6.2) W[1]-hard on LCS (Corollary 6.2)	NP-hard on Σ-DAG if $r \geqslant 3$:const (Theorem 6.1) NP-hard on LCS (Corollary 6.1)
	Approx. or FPT	—	FPT if r: param (Theorem 5.1)	Open

Tractable

Intractable

Approximable

Conclusion

- Diversity Maximization problem for LCS and a Σ-DAG
  - w/ Sum- and Min-diversities (for the first time)
- Investigated the complexity in various settings cases:
  - Bounded K => Tractable (PTIME)
  - Unbounded K => Intractable (NP-hard); Approximable (PTAS) for Max-Sum
  - Parameterized by K only => Param. Intractable (W[1]-hard)
- Not surprising results. However, we required to establish:
  - Negative-typeness of String Hamming distance in O(Lσ) dimension (seems new)
  - Color-coding technique for automata (NFAs or Σ-DAGs)

K: #solutions, r: max. length of input strings

Future work

- ☐ Approximability of Max-Min(LCS) in the case of unbouned K
- ☐ Max-Sum(MCS)/Max-Min(MCS) for MCS (maximal common subsequences) under Edit Distance. (Recently, it was independently shown: m-MCSs have a DAG representation of poly-size with m=2 [Punzi, Grossi, Uno, ISAAC'23], [Hirota & Sakai, arXiv'23], while it is open for any m>= 3.)

After the start up meeting, Sapporo, Nov. 2024


Yuto Shida[1]


Giulia Punzi[2,3]


Yasuaki Kobayashi[1]


Takeaki Uno[2]


Hiroki Arimura[1]

Thank you! And Question?

1) Hokkaido University, Sapporo, Japan

2) National Inst. Informatics, Tokyo, Japan

3) University of Pisa, Pisa, Italy

Slide pdf will be found at “Code section” of this paper’s arXIv site: https://arxiv.org/abs/2405.00131