A Heuristic Approach for Solving the Longest Common Square Subsequence Problem

EUROCAST 19, Las Palmas de Gran Canaria, Spain

Marko Djukanovic[1], Günther Raidl[1], and Christian Blum[2]

[1]Institute of Logic and Computation, TU Wien, Vienna, Austria,
[2] Artificial Intelligence Research Institute (IIIA-CSIC),
Campus UAB, Bellaterra, Spain

February 20, 2019

ALGORITHMS AND COMPLEXITY GROUP

IIIA Institut d'Investigació en Intel·ligència Artificial

Introduction

- A *string* is a finite sequence of characters over (finite) alphabet Σ.
- Strings are used as:
  - data types: words, complete texts
  - models for DNA molecules, proteins, RNA molecules.

String problems in bioinformatics:

- comparing molecules
- the detected similarities serve to better understand biological processes (diseases, developmental defects etc.)
- similarities between molecules: present combinatorial (optimization) problems

Longest Common Subsequence Problem (LCS)

- String $\tilde{s}$ is a *subsequence* of a string s iff it is obtained from s by deleting zero or more characters.
- LCS:
  - Input: A set of strings $S = \{s_1, \ldots, s_m\}, m \in \mathbb{N}$, and an alphabet Σ.
  - Task: Find a *subsequence* of *maximum* length that is *common* for all the input strings.
  - Example: $|S| = 2$, $S = \{abcabcda, accbccaba\}$, $\Sigma = \{a, b, c, d\}$

Longest Common Subsequence Problem (LCS)

- String $\tilde{s}$ is a *subsequence* of a string s iff it is obtained from s by deleting zero or more characters from s.
- LCS:
  - Input: A set of strings $S = \{s_1, \ldots, s_m\}, m \in \mathbb{N}$, and an alphabet Σ.
  - Task: Find a *subsequence* of *maximum* length that is *common* for all strings from S.
  - Example: $|S| = 2$, $S = \{abcabcda, accbccaba\}$, $\Sigma = \{a, b, c, d\}$
    LCS: *abcaa*.

Longest Common Subsequence Problem (LCS)

- String $\tilde{s}$ is a *subsequence* of a string s iff it is obtained from s by deleting zero or more characters from s.
- LCS:
  - Input: A set of strings $S = \{s_1, \ldots, s_m\}, m \in \mathbb{N}$, and an alphabet Σ.
  - Task: Find a *subsequence* of *maximum* length that is *common* for all strings from S.
  - Example: $|S| = 2$, $S = \{abcabcda, accbccaba\}$, $\Sigma = \{a, b, c, d\}$
    LCS: $abcaa$.
  - solvable in polynomial time if m fixed (Dynamic Programming (DP) in $O(n^m)$, $n = \max\{|s_i| \mid i = 1, \ldots, m\}$)
  - $\mathcal{NP}$-hard if S arbitrary

The Longest Common Square Subsequence Problem (LCSqS)

- A string s is a square iff $(\exists s' \in \Sigma^*)\ s = s' \cdot s' = s'^2$
- LCSqS:
  - LCS + sequence is a *square*
  - Example: $s_1 = dabcbacbabc, s_2 = abbcbadc$; LCSqS: $bcbc$.

The Longest Common Square Subsequence Problem (LCSqS)

- A string s is a square iff $(\exists s' \in \Sigma^*)\ s = s' \cdot s' = s'^2$
- LCSqS:
  - LCS + sequence is a *square*
  - Example: $s_1 = dabcbacbabc, s_2 = abbcbadc$; LCSqS: $bcbc$.
- Applications:
  - LCS: a general measure of comparison (*diff* command, *Git*)
  - LCSqS: includes "internal" similarity between molecules
    - similarity between the parts of the compared molecules measured by LCSqS

Solving LCS and LCSqS

- LCS:
  - Beam Search (BS) (Blum et al., 2009)
  - BS + spec. branching (Mousavi and Tabataba, 2012)
  - Chemical Reaction Optimization (Islam et al., 2018)
- LCSqS:
  - introduced by Inoue et al. (2018)
  - the case for $m = 2$ solved by
    - ★ DP in $O(n^6)$
    - ★ sparse DP approach: by 3D–range search tree (in $O(|M|^3 \log^2 n \log\log n + n)$-time, $|M|$=#matchings between strings)

Solving LCS and LCSqS

- LCS:
  - Beam Search (BS) (Blum et al., 2009)
  - BS + spec. branching (Mousavi and Tabataba, 2012)
  - Chemical Reaction Optimization (Islam et al., 2018)
- LCSqS:
  - introduced by Inoue et al. (2018)
  - the case for $m = 2$ solved by
    - DP in $O(n^6)$
    - sparse DP approach: by 3D–range search tree (in $O(|M|^3 \log^2 n \log\log n + n)$-time, $|M|$=#matchings between strings)
  - no algorithm proposed for $m > 2$

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

s_1:	a	c	c	b	c	b	d	c	d
s_2:	b	a	c	c	b	b	c	d	b

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

s_1: a c c b c b d c d

↑ p_1^{L}

s_2: b a c c b b c d b

↑ p_2^{L}

$$p_1^{\mathrm{L}} = p_2^{\mathrm{L}} = 1, s^p = \varepsilon$$

p^{L} : left position vectors

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

s_1:	a	c	c	b	c	b	d	c	d
	$\uparrow p_1^{\mathrm{L}}$								
s_2:	b	a	c	c	b	b	c	d	b
	$\uparrow p_2^{\mathrm{L}}$								

$p_1^{\mathrm{L}} = p_2^{\mathrm{L}} = 1, s^p = \varepsilon$

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

s_1: a c c b c b d c d

$\uparrow p_1^{\mathrm{L}}$

s_2: b a c c b b c d b

$\uparrow p_2^{\mathrm{L}}$

$g(x)$: min. # of letters we skip from search when extending s^p by letter x

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

s_1:	a	c	c	b	c	b	d	c	d
s_2:	b	a	c	c	b	b	c	d	b

LCS: solution approaches

- Best-Next heuristic (BNH) for LCS:
  - Huang et al. (2004)
  - at each iteration feasibly extend current partial sol. s^p by single letter
  - decision which letter to choose: greedy function g
- $S = \{s_1, s_2\}$

$\downarrow p_1^{\mathrm{L}}$

s_1: a c c b c b d c d

s_2: b a c c b b c d b

$\uparrow p_2^{\mathrm{L}}$

Update: $p_1^{\mathrm{L}} = 2$, $p_2^{\mathrm{L}} = 3$, $s^p = a$.

$S[p^{\mathrm{L}}]$: remaining strings w.r.t. position p^{L} (relevant for extension)

BS for LCS

- BS: one of the most simple and effective approaches for the LCS
  - a heuristic search–tree algorithm: principle of a limited BFS
  - expand the most promising nodes of the same level: extensions
  - the best β nodes among extensions further pursued for a beam of the next level
  - BS for the LCS: Blum et al., 2009

BS for the LCS

- Each node v stores:
  - $S[p^{\mathrm{L},v}]$: the remaining strings to extend partial solution
  - l^v: length of the corresp. partial solution
- Expansions of node v:
  - calculate non-dominated feasible letters for $S[p^{\mathrm{L},v}]$
  - extend the partial solution in all possible ways by updating left pos. vectors and l^v (by adding 1) accordingly
- Evaluation of node v:
  - Upper bound: $\mathrm{UB}(v) = \sum_{a \in \Sigma} c_a,\ c_a := \min_{i=1,\ldots,m} |s_i[p_i^{\mathrm{L},v}, |s_i|]|_a$
  - Expected length for a LCS: EX(v)
    (the palindromic LCS (Djukanovic et al., 2018: submitted))

Derivation of EX for the LCS

- $\mathcal{P}(s,t)$: the probability that a string s is a subsequence of a uniform random string t (Mousavi and Tabataba, 2012):
  - $\mathcal{P}(i,j) = \mathcal{P}(|s|,|t|)$ has a matrix presentation
- Some research about the expected length of a LCS:
  - Dixon (2013), Ning and Choi (2013), Znamenskij (2016)
- Derivation based on the assumptions:
  - strings in S are mutually independent
  - an event that a sequence s of length k (over Σ) appears as a subsequence in S is independent of any other such events
- $\Rightarrow \mathrm{EX}(v) = \sum_{k=1}^{l_{\max}} \left(1 - \left(1 - \prod_{k=1}^{m} \mathcal{P}(k, |s_i| - p_i^{\mathrm{L},v} + 1)\right)^{|\Sigma|^k}\right)$,

  where $l_{\max} = \min_{i=1,\ldots,m} |s_i| - p_i^{\mathrm{L},v} + 1$.

The LCSqS problem: transformation

- $\mathbb{P} := \{q \in \mathbb{N}^m \mid 1 \leq q_i \leq |s_i|\}$: the space of partitionings
- Each vector $q \in \mathbb{P}$ divides input set S into the non-empty sets:
  - $S^{\mathrm{L},q} = \{s_i[1, q_i] \mid i = 1, \ldots, m\}$ and
  - $S^{\mathrm{R},q} = \{s_i[q_i + 1, |s_i|] \mid i = 1, \ldots, m\}$.

The LCSqS problem: transformation

- $\mathbb{P} := \{q \in \mathbb{N}^m \mid 1 \leq q_i \leq |s_i|\}$: the space of partitionings
- Each vector $q \in \mathbb{P}$ divides input set S into the non-empty sets:
  - $S^{\mathrm{L},q} = \{s_i[1, q_i] \mid i = 1, \ldots, m\}$ and
  - $S^{\mathrm{R},q} = \{s_i[q_i + 1, |s_i|] \mid i = 1, \ldots, m\}$.
- Example: $s_1 = abcbacbabc$, $s_2 = abbbabaccbcc$, $s_3 = accbcbacbba$ and $q = (3, 3, 4)$:

  $s_1 = \underbrace{abc}_{\in S^{\mathrm{L},q}} \,\|\, \underbrace{bacbabc}_{\in S^{\mathrm{R},q}},$

  $s_2 = \underbrace{abb}_{\in S^{\mathrm{L},q}} \,\|\, \underbrace{babaccbcc}_{\in S^{\mathrm{R},q}},$

  $s_3 = \underbrace{accb}_{\in S^{\mathrm{L},q}} \,\|\, \underbrace{cbacbba}_{\in S^{\mathrm{R},q}}.$

The LCSqS problem: transformation

- $\mathbb{P} := \{q \in \mathbb{N}^m \mid 1 \leq q_i \leq |s_i|\}$: the space of partitionings
- Each vector $q \in \mathbb{P}$ divides input set S into the non-empty sets:
  - $S^{\mathrm{L},q} = \{s_i[1, q_i] \mid i = 1, \ldots, m\}$ and
  - $S^{\mathrm{R},q} = \{s_i[q_i + 1, |s_i|] \mid i = 1, \ldots, m\}$.
- Example: $s_1 = abcbacbabc$, $s_2 = abbbabaccbcc$, $s_3 = accbcbacbba$ and $q = (3, 3, 4)$:

$$s_1 = \underbrace{abc}_{\in S^{\mathrm{L},q}} \parallel \underbrace{bacbabc}_{\in S^{\mathrm{R},q}},$$

$$s_2 = \underbrace{abb}_{\in S^{\mathrm{L},q}} \parallel \underbrace{babaccbcc}_{\in S^{\mathrm{R},q}},$$

$$s_3 = \underbrace{accb}_{\in S^{\mathrm{L},q}} \parallel \underbrace{cbacbba}_{\in S^{\mathrm{R},q}}.$$

- If $s_{\mathrm{lcs}} = \mathrm{LCS}(S^{\mathrm{L},q} \cup S^{\mathrm{R},q} := S^q) \Rightarrow s_{lcsqs} := s_{\mathrm{lcs}} \cdot s_{\mathrm{lcs}}$ is a feasible LCSqS solution on S.

Basic idea of the approach

- LCSqS as a map: $q \in \mathbb{P} \mapsto \mathrm{LCS}(S^q)$
- Solving LCSqS $\Rightarrow$ solving a series of standard LCS instances
- $|\mathrm{LCSqS}| = 2 \cdot \max_{q \in \mathbb{P}} |\mathrm{LCS}(S^q)|$:
  - the overall number of partitionings exponential in problem size
  - LCS is $\mathcal{NP}$-hard

Iterated Greedy (IG) approach

- Destruct: randomly sample $S' \subseteq S$ s.t. $|S'| \approx \lfloor \mathrm{destr} \cdot |S| \rfloor$ where $\mathrm{destr} \in (0,1)$ is the parameter of destruction
- q'=Construct(q): generate q' by mutating q as follows:
  - $q'_i \in \mathcal{D}_i(q_i, \sigma) := q_i + \lceil \mathcal{N}(0, \sigma^2) \rceil$, $s_i \in S'$, $\sigma > 0$ parameter
  - if $q'_i \notin \{1, \ldots, |s_i|\}$, sample again
- Second phase (construction of a LCSqS sol.): $q \xmapsto{g_{\mathrm{appx}}} \mathrm{BNH}(S^q)$
- Acceptance criterion: always better partitioning acc. to $|g_{\mathrm{appx}}(q)|$-value

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

$q_1 = 4$

s_1: a c b b | a a d d c

$q_2 = 4$

s_2: a c c b | c b d c d

$q_3 = 4$

s_3: b a c c | b b c d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

				$q_1 = 4$					
s_1:	a	c	b	b |	a	a	d	d	c
				$q_2 = 4$					
s_2:	a	c	c	b |	c	b	d	c	d
				$q_3 = 4$					
s_3:	b	a	c	c |	b	b	c	d	b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

$q_1 = 4$

s_1: a c b b | a a d d c

$q'_2 = 5$

s_2: a c c b c | b d c d

$q_3 = 4$

s_3: b a c c | b b c d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

s_1: a c b b | a a d d c

s_2: a c c b c | b d c d

s_3: b a c c | b b c d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

s_1: a c b b | a a d d c

s_2: a c c b c | b d c d

$q'_3 = 2$

s_3: b a | c c b b c d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

s_1: a c b b | a a d d c

s_2: a c c b c | b d c d

$q'_3 \leq \frac{|s_{\mathrm{lcsqs}}|}{2} \Rightarrow$ invalid

s_3: b a | c c b b c d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

s_1: a c b b | a a d d c

s_2: a c c b c | b d c d

$q'_3 \geq |s_3| - \frac{|s_{\mathrm{lcsqs}}|}{2} \Rightarrow \text{invalid}$

s_3: b a c c b b c | d b

$\mathcal{D}_i$ distribution, reduction of search space $\mathbb{P}$

- Properties of $\mathcal{D}_i(q_i, \sigma)$:
  - similarity between q and q' (generated by mutating q) are controlled by $\sigma > 0$
- Search space reduction:
  - assume that s_{lcsqs} is the best so far found LCSqS solution

$\Rightarrow \mathcal{D}_i$ defined on $I_i = \left\{ \frac{|s_{\mathrm{lcsqs}}|}{2} + 1, \ldots, |s_i| - \frac{|s_{\mathrm{lcsqs}}|}{2} - 1 \right\}, i = 1, \ldots, m$

IG algorithm

Data: an instance (S, Σ),
$t_{\max} > 0$, *destr* $\in (0,1)$, $\sigma > 0$: std deviation

Result: a feasible LCSqS solution

$q \leftarrow \left(\left\lfloor \frac{|s_i|}{2} \right\rfloor \right)_{i=1}^{m}$ //initialize;

$s_{\mathrm{lcsqs}} \leftarrow \varepsilon$;

while $t_{\max}$ *not exceeded* **do**

  $q' \leftarrow$ Destruct-Construct$(q, \mathcal{D}, destr)$;

  $s_{q'} \leftarrow$ BNH$(S^{q'})$;

  if $2 \cdot |s_{q'}| > |s_{\mathrm{lcsqs}}|$ **then**

    $s_{\mathrm{lcsqs}} \leftarrow s_{q'} \cdot s_{q'}$;

    $q \leftarrow q'$;

  end

end

return s_{lcsqs};

IG algorithm for the LCSqS.

VNS & BS approach

- VNS (Mladenovic, 1997):
  - systematically change neighborhoods in search space $\mathbb{P}$:
    - ★ for a fixed vector $q = (q_1, \ldots, q_m)$, the k-th neighborhood defined as

$$N_k(q) := \{q' \in \mathbb{P} : q \text{ and } q' \text{ differ at } k \text{ positions}\}, 1 \leq k \leq m,$$

    - ★ $q' \in N_k(q)$ gen. by mutating q w. r. t. $\mathcal{D}(q,\sigma)=(\mathcal{D}_1(q_1,\sigma), \ldots, \mathcal{D}_m(q_m,\sigma))$

VNS & BS approach

- VNS (Mladenovic, 1997):
  - systematically change neighborhoods in search space $\mathbb{P}$:
    - for a fixed vector $q = (q_1, \ldots, q_m)$, the k-th neighborhood defined as

$$N_k(q) := \{q' \in \mathbb{P} : q \text{ and } q' \text{ differ at } k \text{ positions}\}, 1 \leq k \leq m,$$

    - $q' \in N_k(q)$ gen. by mutating q w. r. t. $\mathcal{D}(q,\sigma) = (\mathcal{D}_1(q_1,\sigma), \ldots, \mathcal{D}_m(q_m,\sigma))$
- Evaluating partitionings:
  - LCS study:
    - BS gives solutions of better quality than BNH
    - BS too expensive to perform in each partitioning of $\mathbb{P}$
  - $\Rightarrow$ trade off found (next slide)...

VNS & BS: details

- Evaluating partitionings: realization of $q \xrightarrow{\text{Eval}} f_q$ by:

$ub_{lcsqs} \leftarrow 2 \cdot \mathrm{UB}(S^q)$;
if $ub_{lcsqs} > |s_{\mathrm{lcsqs}}|$ **then**
  $f_q \leftarrow 2 \cdot |\mathrm{BNH}(S^q)|$;
  if $f_q > \alpha \cdot |s_{\mathrm{lcsqs}}|$ **then**
    $f_{\mathrm{bs}} \leftarrow 2 \cdot |\mathrm{BS}(S^q)|$;
    if $f_{bs} > f_q$ **then**
      $f_q \leftarrow f_{\mathrm{bs}}$;
    end
  end
  //update s_{lcsqs} possibly;
else
  $f_q \leftarrow 0$ //invalid;
end

Eval(q).

VNS & BS: details

- Evaluating partitionings: realization of $q \xmapsto{\text{Eval}} f_q$ by:

```
ub_lcsqs ← 2 · UB(S^q);
if ub_lcsqs > |s_lcsqs| then
    f_q ← 2 · |BNH(S^q)|;
    if f_q > α · |s_lcsqs| then
        f_bs ← 2 · |BS(S^q)|;
        if f_bs > f_q then
            f_q ← f_bs;
        end
    end
    //update s_lcsqs
     possibly;
else
    f_q ← 0  //invalid;
end
```

Eval(q).

- All partitionings evaluated by BS stored in a hash map (together with its f_q-val.)

Experiments

- Machine settings:
  - Intel Xeon E5-2640 v4 CPU, 2.40GHz
  - memory limit: 8GB
- Instances: LCS instances (Blum, 2016):
  - for each combination of $|\Sigma| \in \{4, 12, 20\}$, $m \in \{10, 50, 100, 150, 200\}$ and $n \in \{100, 500, 1000\}$, 10 instances are generated $\Rightarrow$ 450 instances, 10 independent runs per single instance
  - the results are grouped by each combination presenting:
    - ★ the avg. over solution lengths
    - ★ the avg. median times when best sol. has been found
- Fixed: $t_{max} = 600s$ for all algorithms

Parameters' settings

- IG parameters:
  - $destr = 0.3$
  - $n = 100$: $\sigma = 5$
  - $n = 500$: $\sigma = 10$
  - $n = 1000$: $\sigma = 20$
- VNS & BS parameters:
  - $\beta = 200$
  - heuristic guidance of BS: EX
  - $\alpha = 0.9$ for $n \in \{100, 500\}$ and $\alpha = 0.95$ for $n = 1000$.
  - σ-settings: the same like in the IG for corresponding n

Numerical results: $n = 100$

m	$\|\Sigma\|$	VNS & BS		IG & BS		VNS & Dive		IG	
		$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$
10	4	**27.04**	40.62	26.54	44.94	26.96	51.20	26.58	40.10
	12	**8.40**	16.97	8.04	13.59	8.28	19.27	7.98	22.56
	20	3.96	0.34	**4.00**	1.66	3.96	0.05	**4.00**	3.15
50	4	18.48	26.43	18.16	24.12	**18.54**	45.81	18.32	36.93
	12	**3.88**	3.45	3.82	11.01	**3.88**	5.00	3.80	8.51
	20	0.22	1.37	**0.46**	4.77	0.20	0.00	**0.46**	10.49
100	4	**16.20**	29.51	16.02	17.95	16.14	8.44	16.02	14.76
	12	1.58	0.00	**2.00**	0.10	1.64	6.19	**2.00**	0.07
	20	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
150	4	**15.10**	61.37	14.40	38.22	15.06	85.49	14.34	34.57
	12	0.40	0.00	**2.00**	10.47	0.40	0.00	**2.00**	8.72
	20	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00
200	4	**14.04**	3.71	14.00	8.68	14.00	1.36	13.92	21.14
	12	0.00	0.00	**1.58**	33.89	0.00	0.00	**1.58**	42.54
	20	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00	**0.00**	0.00

Table: Results for $n = 100$.

Numerical results: $n = 500$

m	$\lvert\Sigma\rvert$	VNS & BS		IG & BS		VNS & Dive		IG	
		$\overline{\lvert s\rvert}$	$\overline{t_{\text{best}}}[s]$	$\overline{\lvert s\rvert}$	$\overline{t_{\text{best}}}[s]$	$\overline{\lvert s\rvert}$	$\overline{t_{\text{best}}}[s]$	$\overline{\lvert s\rvert}$	$\overline{t_{\text{best}}}[s]$
10	4	**156.76**	132.86	156.46	151.96	150.42	122.27	150.14	111.66
	12	**58.88**	94.19	58.40	115.06	56.58	101.47	56.48	60.93
	20	**36.02**	89.36	35.46	44.85	34.76	56.93	34.66	54.49
50	4	**124.34**	68.61	124.24	64.19	120.86	87.26	120.36	127.39
	12	**38.82**	79.30	38.56	58.59	38.18	32.68	38.12	19.71
	20	21.18	79.60	20.84	70.09	**21.20**	81.55	20.94	68.38
100	4	**115.84**	53.76	115.72	66.50	113.18	116.39	111.96	98.33
	12	**34.00**	48.61	33.92	53.06	33.16	86.84	33.20	49.58
	20	**18.00**	29.05	**18.00**	52.47	**18.00**	58.39	17.76	65.87
150	4	**112.06**	47.59	111.86	115.71	109.56	129.93	107.68	94.80
	12	**31.94**	111.87	31.84	94.16	31.06	123.15	30.86	59.42
	20	**16.00**	5.71	**16.00**	4.65	**16.00**	5.90	**16.00**	6.53
200	4	**109.80**	141.07	109.08	121.19	106.84	109.65	105.00	103.23
	12	**30.00**	29.76	**30.00**	17.43	28.44	79.77	29.84	72.14
	20	**14.76**	73.08	14.00	0.00	14.22	40.04	14.08	9.36

Table: Results for $n = 500$.

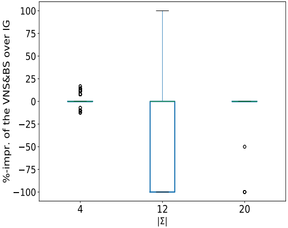
Numerical results: $n = 1000$

m	$\|\Sigma\|$	VNS & BS		IG & BS		VNS & Dive		IG	
		$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$	$\overline{\|s\|}$	$\overline{t_{\text{best}}}[s]$
	4	320.44	143.22	**321.04**	185.80	304.48	186.65	304.34	161.08
10	12	124.12	143.59	**124.16**	158.39	117.86	151.51	117.88	134.88
	20	**77.02**	123.92	76.68	122.87	73.80	118.86	73.72	76.98
	4	**261.48**	107.59	260.84	129.52	252.94	131.88	249.84	153.18
50	12	**86.06**	108.55	85.98	134.32	83.34	132.11	83.98	100.37
	20	**49.86**	133.71	49.64	112.94	48.12	54.48	48.70	74.04
	4	**247.24**	165.80	246.28	141.55	240.24	109.11	234.36	145.40
100	12	77.60	199.63	**77.70**	197.42	75.44	137.65	75.28	118.57
	20	**43.66**	176.18	43.56	169.68	42.02	17.65	42.28	30.80
	4	**240.06**	181.95	239.30	155.08	234.02	127.31	227.02	127.81
150	12	**73.78**	179.46	73.40	179.63	71.76	121.40	71.36	120.95
	20	**40.04**	112.26	40.02	109.40	39.88	121.36	39.96	59.40
	4	**235.54**	185.20	234.94	163.46	230.10	135.37	222.66	145.99
200	12	**70.80**	195.07	70.32	190.15	69.10	144.78	68.30	44.32
	20	**38.06**	122.44	38.02	115.88	38.00	59.74	38.02	24.07

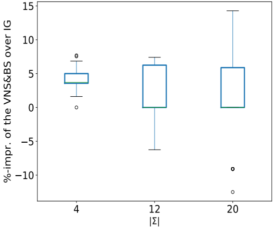
Table: Results for $n = 1000$.

Plots: VNS & BS vs. IG: sol. quality comparison:

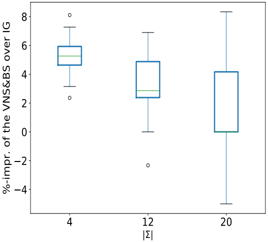
$n = 100$.



$n = 500$.

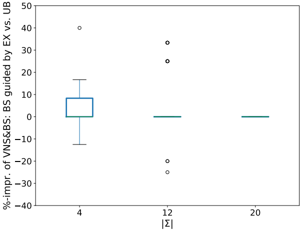


$n = 1000$.

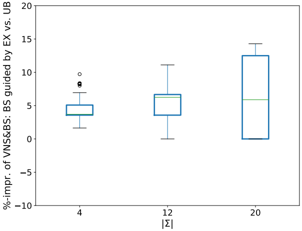


Heuristic guidance of BS component: EX vs. UB

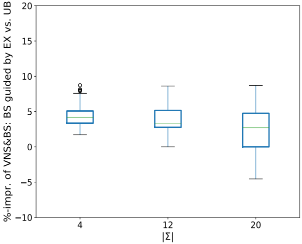
$n = 100.$



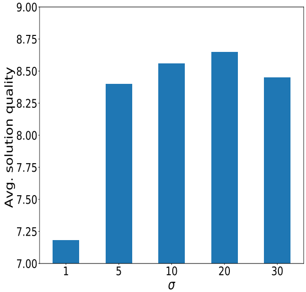
$n = 500.$



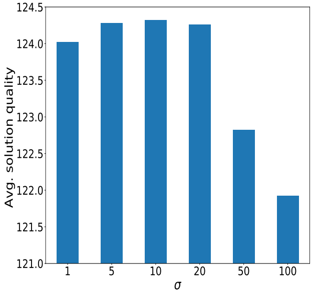
$n = 1000.$



Impact of different values for σ on VNS & BS



$n = 100, m = 10, |\Sigma| = 12.$



$n = 500, m = 50, |\Sigma| = 4.$

Conclusion & Future Work

- Conclusion:
  - We introduced a reduction and approaches to solve the LCSqS for arbitrary sets of input strings
  - We derived a heuristic guidance based on the approximated expected length for a LCS

Conclusion & Future Work

- Conclusion:
  - We introduced a reduction and approaches to solve the LCSqS for arbitrary sets of input strings
  - We derived a heuristic guidance based on the approximated expected length for a LCS
- Future work:
  - The LCSqS approaches $\Rightarrow$ we get a BS for the LCS guided by EX (a new state-of-the-art for LCS possibly?)
  - Exact ways of solving the LCSqS:
    - ★ BS creates independently a BS–tree for each partitioning
    - ★ Is there any connection between already created nodes?
    - ★ Node's structure $v = (p^{\mathrm{L}}, l^v, \underbrace{X(q_v)}_{?}), q_v \in \mathbb{P}$: upper bounds...
    - ★ Applying A* (or some other exact algorithm)...

Thank you for your attention!