Strange Parameterized Complexity Results of Natural Combinatorial Problems in Automata Theory and Algebra

Henning Fernau

June 22nd, 2021

University of Trier

Parameterized Prerequisites

Parameterized History

$\sim$ 1990, questioning the basics of classical complexity:
When is it ever the case that the only thing you know about your problem instance is its number of bits?

Parameterized History

$\sim$ 1990, questioning the basics of classical complexity:
When is it ever the case that the only thing you know about your problem instance is its number of bits?

$\rightsquigarrow$ What about finding a second (or third, etc.) dimension to measure an instance?

Parameterized History

$\sim$ 1990, questioning the basics of classical complexity:
When is it ever the case that the only thing you know about your problem instance is its number of bits?

$\rightsquigarrow$ What about finding a second (or third, etc.) dimension to measure an instance?

$\rightsquigarrow$ Parameterizations (a.k.a. *multivariate analysis* (A. Nerode))

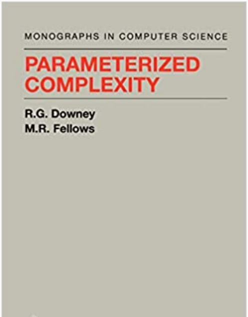
Parameterized History

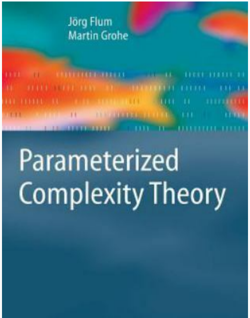
$\sim$ 1990, questioning the basics of classical complexity:

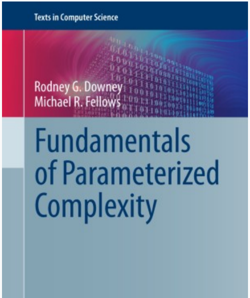
When is it ever the case that the only thing you know about your problem instance is its number of bits?

$\rightsquigarrow$ What about finding a second (or third, etc.) dimension to measure an instance?

$\rightsquigarrow$ Parameterizations (a.k.a. *multivariate analysis* (A. Nerode))







Parameterized Complexity Classes

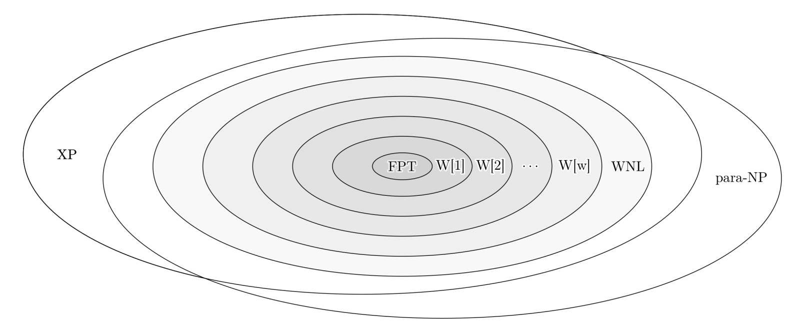


Figure 1: Parameterized Complexity Classes, parameter k:

FPT $\rightarrow \mathcal{O}(f(k) \cdot n^c)$,
XP $\rightarrow \mathcal{O}(n^{f(k)})$,
para-NP $\rightarrow$ non-det. $\mathcal{O}(f(k) \cdot n^c)$

Further Parameterized Complexity Classes

Definition (NONDETERMINISTIC TURING MACHINE COMPUTATION)

Input: A non-deterministic Turing machine M, $q \in \mathbb{N}$, $p \in \mathbb{N}$ in unary as well as $k \in \mathbb{N}$.

Problem: Does M accept the empty string in $\leq q$ steps, with $\leq p$ guessing steps, using $\leq k$ tape cells?

Further Parameterized Complexity Classes

Definition (Nondeterministic Turing Machine Computation)

Input: A non-deterministic Turing machine M, $q \in \mathbb{N}$, $p \in \mathbb{N}$ in unary as well as $k \in \mathbb{N}$.

Problem: Does M accept the empty string in $\leq q$ steps, with $\leq p$ guessing steps, using $\leq k$ tape cells?

Definition (Parameterized reduction)

For a parameterized Problem Π, let $[\Pi]^{\mathrm{FPT}}$ denote the problems reducible to Π in FPT time, obeying the parameterization.

Further Parameterized Complexity Classes

Definition (NONDETERMINISTIC TURING MACHINE COMPUTATION)

Input: A non-deterministic Turing machine M, $q \in \mathbb{N}$, $p \in \mathbb{N}$ in unary as well as $k \in \mathbb{N}$.

Problem: Does M accept the empty string in $\leq q$ steps, with $\leq p$ guessing steps, using $\leq k$ tape cells?

Definition (Parameterized reduction)

For a parameterized Problem Π, let $[\Pi]^{\mathsf{FPT}}$ denote the problems reducible to Π in FPT time, obeying the parameterization.

According to Cesati 2003, Guillemot 2011, we have:

$\mathrm{W}[1] = [\text{1-TAPE-NTMC}[q]]^{\mathsf{FPT}}$

$\mathrm{W}[2] = [\text{MULTI-TAPE-NTMC}[q]]^{\mathsf{FPT}}$

$\mathrm{WNL} = [\text{NTMC}[k]]^{\mathsf{FPT}}$

$\mathrm{W}[\mathrm{P}] = [\text{NTMC}[p]]^{\mathsf{FPT}}$

Further Parameterized Complexity Classes

Definition (Nondeterministic Turing Machine Computation)

Input: A non-deterministic Turing machine M, $q \in \mathbb{N}$, $p \in \mathbb{N}$ in unary as well as $k \in \mathbb{N}$.

Problem: Does M accept the empty string in $\leq q$ steps, with $\leq p$ guessing steps, using $\leq k$ tape cells?

Definition (Parameterized reduction)

For a parameterized Problem Π, let $[\Pi]^{\text{FPT}}$ denote the problems reducible to Π in FPT time, obeying the parameterization.

According to Cesati 2003, Guillemot 2011, we have:

W[1] $= [\text{1-Tape-NTMC}[q]]^{\text{FPT}}$

W[2] $= [\text{Multi-Tape-NTMC}[q]]^{\text{FPT}}$

WNL $= [\text{NTMC}[k]]^{\text{FPT}}$

W[P] $= [\text{NTMC}[p]]^{\text{FPT}}$

A[2]: ATM, start in existential state, one switch to universal states

Complexity, Visualized...

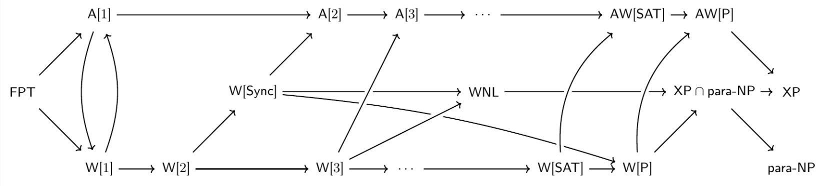


Figure 2: Overview of the complexity classes

Complexity, Visualized...

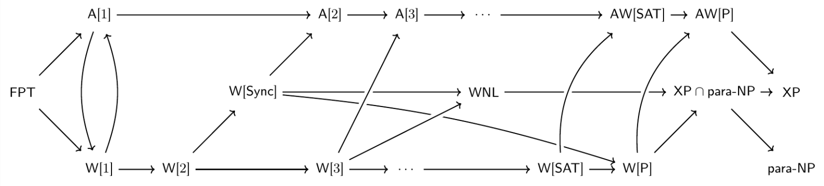


Figure 2: Overview of the complexity classes

A bit neglected in recent years ...
Focus clearly shifted towards algorithms.
But still of mathematical interest:

Exact classification of concrete problems.

A First Algebraic Flavor: Subsemigroup Isomorphism

Theorem
(Downey, Fellows 1999) The problem "Given two finite semigroups G, H, is H isomorphic to some subsemigroup of G?" is W[1]-complete. Our Parameter: $|H|$

A First Algebraic Flavor: Subsemigroup Isomorphism

Theorem
(Downey, Fellows 1999) The problem "Given two finite semigroups G, H, is H isomorphic to some subsemigroup of G?" is W[1]-complete. Our Parameter: $|H|$

Membership in W[1]: TM simulation, guessing $|H|$ elements of G

A First Algebraic Flavor: Subsemigroup Isomorphism

Theorem
(Downey, Fellows 1999) The problem "Given two finite semigroups G, H, is H isomorphic to some subsemigroup of G?" is W[1]-complete. Our Parameter: $|H|$

Membership in W[1]: TM simulation, guessing $|H|$ elements of G

W[1]-hardness by reduction from Clique (as in Booth 1978):
From graph $\Gamma = (V, E)$, define $\circ$ on $G = V \cup E \cup \{0\}$ by:

$$x \circ y = \begin{cases} x, & x = y \vee (x \in E \wedge y \in x) \\ y, & x = y \vee (y \in E \wedge x \in y) \\ \{x, y\}, & \{x, y\} \in E \\ 0, & \text{otherwise} \end{cases}$$

E.g., K_k yields a semigroup S_k of size $k(k+1)/2 + 1$.
S_k is a subsemigroup of G iff Γ contains a k-clique.

A Famous Combinatorial Question in Automata Theory

Synchronizing Words

Definition (Synchronizing Word)
A *synchronizing word (SW)* for a DFA $A = (Q, \Sigma, \delta(, q_0, F))$ is some $w_{\text{sync}} \in \Sigma^*$, so that there is one *synchronizing state* $q_{\text{sync}} \in Q$ with $\delta(q, w_{\text{sync}}) = q_{\text{sync}}$ for all $q \in Q$.

Definition (Problem DFA-SW)

Input:	DFA A, $k \in \mathbb{N}$
Problem:	Is there a synchronizing word w for A with $\lvert w \rvert \leq k$?

DFA-SW NP-complete and W[2]-hard wrt. standard param. k,
DFA Synchronizability (without length bound) poly-time.

Černý's conjecture: Every DFA with a SW has also one no longer than $(|Q| - 1)^2$.
Best upper bound $\mathcal{O}(|Q|^3)$, see STACS 2018, JALC 2019.

An Example: Černý's Automaton

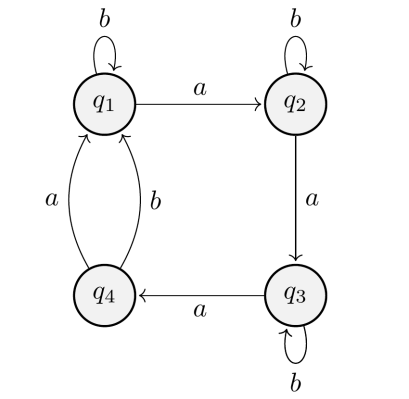


Figure 3: Černý's automaton

An Example: Černý's Automaton

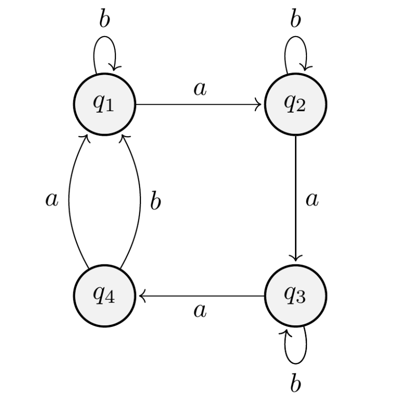


Figure 3: Černý's automaton

Minimum synchronizing word: ba^3ba^3b

Synchronizing Words

Definition (Synchronizing Word)
A *synchronizing word (SW)* for a DFA $A = (Q, \Sigma, \delta(, q_0, F))$ is some $w_{\text{sync}} \in \Sigma^*$, so that there is one *synchronizing state* $q_{\text{sync}} \in Q$ with $\delta(q, w_{\text{sync}}) = q_{\text{sync}}$ for all $q \in Q$.

Definition (DFA-SW)

Input: DFA A, $k \in \mathbb{N}$

Problem: Is there a synchronizing word w for A with $|w| \leq k$?

DFA-SW NP-complete and W[2]-hard wrt. standard param. k,
DFA Synchronizability (without length bound) poly-time.

Černý's conjecture (1964): Every DFA with a SW has also one no longer than $(|Q| - 1)^2$.
Best upper bound $\mathcal{O}(|Q|^3)$, see STACS 2018, JALC 2019.

DFA With a Sink State

- q sink state iff $\forall a \in \Sigma : \delta(q, a) = q$.

DFA With a Sink State

- q sink state iff $\forall a \in \Sigma : \delta(q, a) = q$.
- DFA with one sink state are synchronizable.
  The sink state is the synchronizing state.

DFA With a Sink State

- q sink state iff $\forall a \in \Sigma : \delta(q, a) = q$.
- DFA with one sink state are synchronizable.
  The sink state is the synchronizing state.

Definition (DFA-1SINK-SW)

Input: DFA A with one sink state, $k \in \mathbb{N}$

Problem: Is there a synchronizing word w for A with $|w| \leq k$?

DFA With a Sink State

- q sink state iff $\forall a \in \Sigma : \delta(q, a) = q$.
- DFA with one sink state are synchronizable.
  The sink state is the synchronizing state.

Definition (DFA-1SINK-SW)
Input: DFA A with one sink state, $k \in \mathbb{N}$
Problem: Is there a synchronizing word w for A with $|w| \leq k$?

Lemma
DFA-1SINK-SW $\equiv_{FPT}$ *DFA-SW.*

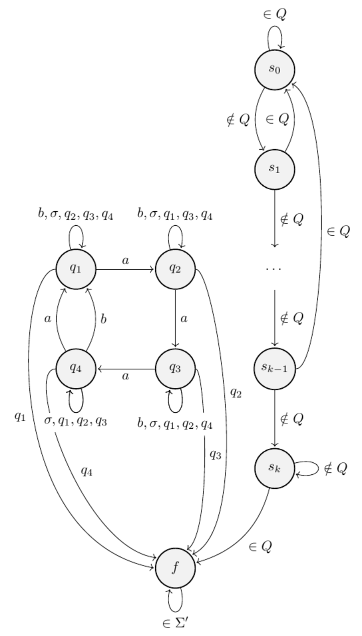


Figure 4: Construction applied to Černý's automaton, $w' = \sigma^{k-|w|}wq_1$

Searching a Home for DFA-SW

DFA-SW: WNL, XP and A[2], but W[SAT] ??

Theorem
DFA-SW $\in WNL \cap W[P] \cap A[2]$.

DFA-SW: WNL, XP and A[2], but W[SAT] ??

Theorem
DFA-SW $\in$ *WNL* $\cap$ *W[P]* $\cap$ *A[2]*.

Proof.

1. M (guessing…) writes a word $w \in \Sigma$, $|w| \leq k$, on its tape, followed by some letter q_{sync} over the alphabet Q.

DFA-SW: WNL, XP and A[2], but W[SAT] ??

Theorem
DFA-SW $\in$ *WNL* $\cap$ *W*[*P*] $\cap$ *A*[2].

Proof.

1. M (guessing…) writes a word $w \in \Sigma$, $|w| \leq k$, on its tape, followed by some letter q_{sync} over the alphabet Q.
2. For all $q \in Q$, M first moves its head to the left end of its tape and then starts reading it from left to right.
   For all read $a \in \Sigma$, M updates its state (transition function).
   When reading q_{sync}, M stops if q_{sync} is not the current state.

DFA-SW: WNL, XP and A[2], but W[SAT] ??

Theorem
DFA-SW $\in$ *WNL* $\cap$ *W*[*P*] $\cap$ *A*[2].

Proof.

1. M (guessing…) writes a word $w \in \Sigma$, $|w| \leq k$, on its tape, followed by some letter q_{sync} over the alphabet Q.
2. For all $q \in Q$, M first moves its head to the left end of its tape and then starts reading it from left to right.
   For all read $a \in \Sigma$, M updates its state (transition function).
   When reading q_{sync}, M stops if q_{sync} is not the current state.
3. M accepts only after correctly completing Step 2.
   $\Rightarrow w$ is synchronizing.
   $\Rightarrow M$ accepts λ iff there is a possibility to guess a SW of length $\leq k$, making at most $(|Q| + 1)(2k + 1)$ many steps, visiting at most $k + 1$ tape cells, making at most $k + 1$ guesses. $\square$

How to Factor Monoids

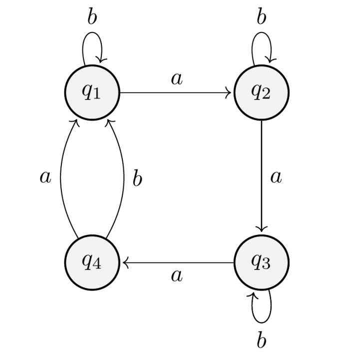
Definition (Monoid Factorization)

Input: A finite set Q, a collection $F = \{f_0, f_1, \ldots, f_m\}$ of mappings $f_i : Q \to Q$, $k \in \mathbb{N}$

Problem: Is there a selection of at most k mappings $f_{i_1}, \ldots, f_{i_{k'}}$, $k' \leq k$, with $i_j \in \{1, \ldots, m\}$ for $j = 1, \ldots, k'$, such that $f_0 = f_{i_1} \circ f_{i_2} \circ \cdots \circ f_{i_{k'}}$?

Standard parameter: $k \to$ W[2]-hard (Cai et al. 1997)

Back to Černý's Automaton



Recall:
transformation monoid

	f_a	f_b	f_0
q_1	q_2	q_1	q_1
q_2	q_3	q_2	q_1
q_3	q_a	q_3	q_1
q_4	q_1	q_1	q_1

Minimum synchronizing word: ba^3ba^3b

Putting Pieces Together

Theorem
Monoid Factorization is (parameterized and polynomial-time) equivalent to DFA-SW.

Proof.
We can reduce Monoid Factorization to DFA-SW.

Putting Pieces Together

Theorem
Monoid Factorization is (parameterized and polynomial-time) equivalent to DFA-SW.

Proof.
We can reduce Monoid Factorization to DFA-SW.
Conversely, start with DFA-1sink-SW (see Černý example).
Interpreting a given DFA $A = (Q, \Sigma, \delta(, q_0, F))$ with one sink state s_f as a collection F_A of $|\Sigma|$ many mappings $f_a : Q \to Q$, by setting $f_a(q) = \delta(q, a)$, we can solve the DFA-SW problem given by (A, k) by solving the instance (F, k) of Monoid Factorization, where $F = \{f_0 = s_f\} \cup F_A$ and the aim is to represent the constant target map $f_0 = s_f$. □

A New Complexity Class? W[Sync]

DFA-SW seems to be difficult to classify, but there are (more) problems with the same complexity.

This motivates the following definition.

W[Sync] $:=$ [DFA-Sync]$^{\mathsf{FPT}}$.

Corollary
Monoid Factorization is W[Sync]-complete.

Bounded DFA-Intersection

Definition (Bounded DFA Intersection (BDFAI))

Input: A finite set $\mathcal{A}$ of DFA over an alphabet Σ and a positive integer k.

Problem: Is there a string $x \in \Sigma^k$ that is accepted by each DFA in $\mathcal{A}$?

Known: BDFAI (param. k) is W[2]-hard (Wareham, 2001), with param. $|\mathcal{A}|$ WNL-complete (Guillemot, 2011).

Definition (Bounded DFA Intersection (BDFAI))

Input: A finite set $\mathcal{A}$ of DFA over an alphabet Σ and a positive integer k.

Problem: Is there a string $x \in \Sigma^k$ that is accepted by each DFA in $\mathcal{A}$?

Known: BDFAI (param. k) is W[2]-hard (Wareham, 2001), with param. $|\mathcal{A}|$ WNL-complete (Guillemot, 2011).

Theorem
Bounded DFA-Intersection, parameterized by the length of the commonly accepted string, is complete for W[Sync].

Definition (BOUNDED DFA INTERSECTION (BDFAI))

Input: A finite set $\mathcal{A}$ of DFA over an alphabet Σ and a positive integer k.

Problem: Is there a string $x \in \Sigma^k$ that is accepted by each DFA in $\mathcal{A}$?

Known: BDFAI (param. k) is W[2]-hard (Wareham, 2001), with param. $|\mathcal{A}|$ WNL-complete (Guillemot, 2011).

Theorem
BOUNDED DFA-INTERSECTION, parameterized by the length of the commonly accepted string, is complete for W[Sync].

A Multivariate Approach To BDFAI

Single Parameters — bivariate

$\lvert\mathcal{A}\rvert$	$q_{\max}$	$\lvert\Sigma\rvert$	k
WNL-c.	$\notin^*$ XP	$\notin^*$ XP	W[sync]-c.
Guillemot 2011	(Wareham 2001)	Wareham 2001	Bruchertseifer & F 2020

where * assumes $\mathrm{P} \neq \mathrm{NP}$ and $q_{\max} = \max\{\lvert Q_A\rvert \mid A \in \mathcal{A}\}$.

A Multivariate Approach To BDFAI

Single Parameters — bivariate

$\lvert\mathcal{A}\rvert$	$q_{\max}$	$\lvert\Sigma\rvert$	k
WNL-c.	$\notin^*$ XP	$\notin^*$ XP	W[sync]-c.
Guillemot 2011	(Wareham 2001)	Wareham 2001	Bruchertseifer & F 2020

where * assumes $\mathrm{P} \neq \mathrm{NP}$ and $q_{\max} = \max\{\lvert Q_A\rvert \mid A \in \mathcal{A}\}$.

Combined Parameters — multivariate

$\lvert\mathcal{A}\rvert + q_{\max}$	$\lvert\mathcal{A}\rvert + \lvert\Sigma\rvert$	$\lvert\mathcal{A}\rvert + k$	$q_{\max} + \lvert\Sigma\rvert$	$q_{\max} + k$	$\lvert\Sigma\rvert + k$
FPT	WNL-c.	W[1]-h.	FPT	W[2]-h.	FPT
Wareham	Guillemot	Wareham	Wareham	Wareham	Wareham

A Multivariate Approach To BDFAI

Single Parameters — bivariate

$\lvert\mathcal{A}\rvert$	$q_{\max}$	$\lvert\Sigma\rvert$	k
WNL-c.	$\notin^*$ XP	$\notin^*$ XP	W[sync]-c.
Guillemot 2011	(Wareham 2001)	Wareham 2001	Bruchertseifer & F 2020

where * assumes P $\neq$ NP and $q_{\max} = \max\{\lvert Q_A\rvert \mid A \in \mathcal{A}\}$.

Combined Parameters — multivariate

$\lvert\mathcal{A}\rvert + q_{\max}$	$\lvert\mathcal{A}\rvert + \lvert\Sigma\rvert$	$\lvert\mathcal{A}\rvert + k$	$q_{\max} + \lvert\Sigma\rvert$	$q_{\max} + k$	$\lvert\Sigma\rvert + k$
FPT	WNL-c.	W[1]-h.	FPT	W[2]-h.	FPT
Wareham	Guillemot	Wareham	Wareham	Wareham	Wareham

Notice: FPT-results are close to trivial (quite typical here . . .)

Proving Membership in W[1]

Theorem
BDFAI, param. by $|\mathcal{A}| + k$*, is W[1]-complete.*

Proving Membership in W[1]

Theorem
BDFAI, param. by $|\mathcal{A}| + k$, is W[1]-complete.

Revisit the **Guess & Check** paradigm:

1. Guess a word w from $(\Sigma Q^{|\mathcal{A}|})^k$.
2. Check if with these letters from Σ, the DFAs can move from state to state as guessed.

Proving Membership in W[1]

Theorem
BDFAI, param. by $|\mathcal{A}| + k$, is W[1]-complete.

Revisit the **Guess & Check** paradigm:

1. Guess a word w from $(\Sigma Q^{|\mathcal{A}|})^k$.
2. Check if with these letters from Σ, the DFAs can move from state to state as guessed.

Observe:

- Only $(|\mathcal{A}| + 1) \cdot k$ many letters are written on the tape.
- The checking can be done in $f(|\mathcal{A}| + k)$ time.

W[Sync] – What Is Inside?

Longest Common Subsequence

Definition (Longest Common Subsequence)

Input: The input consists of ℓ strings $x_1, \ldots, x_\ell$ over Σ.

Problem: Find a string $w \in \Sigma^k$ occurring in each of the x_i as a subsequence.

Known: Longest Common Subsequence with param. k is W[2]-hard, wrt. (k, ℓ) W[1]-complete (Bodlaender et al., 1995), with param. ℓ WNL-complete (Guillemot, 2011)

Definition (Longest Common Subsequence)

Input: The input consists of ℓ strings $x_1, \ldots, x_\ell$ over Σ.

Problem: Find a string $w \in \Sigma^k$ occurring in each of the x_i as a subsequence.

Known: Longest Common Subsequence with param. k is W[2]-hard, wrt. (k, ℓ) W[1]-complete (Bodlaender et al., 1995), with param. ℓ WNL-complete (Guillemot, 2011)

Build a DFA A_i for each x_i that accepts all subsequences of x_i.
$\rightsquigarrow$ One can solve a Longest Common Subsequence instance with a Bounded DFA-Intersection instance, preserving our parameter k. (Wareham, 2001)

Theorem

Longest Common Subsequence $\in W[Sync]$.

Definition (Longest Common Subsequence)

Input: The input consists of ℓ strings $x_1, \ldots, x_\ell$ over Σ.

Problem: Find a string $w \in \Sigma^k$ occurring in each of the x_i as a subsequence.

Known: Longest Common Subsequence with param. k is W[2]-hard, wrt. (k, ℓ) W[1]-complete (Bodlaender et al., 1995), with param. ℓ WNL-complete (Guillemot, 2011)

Build a DFA A_i for each x_i that accepts all subsequences of x_i.
$\leadsto$ One can solve a Longest Common Subsequence instance with a Bounded DFA-Intersection instance, preserving our parameter k. (Wareham, 2001)

Theorem
Longest Common Subsequence $\in W[Sync]$.

W[Sync]-hardness open

Constraint Satisfaction Sitting Between

Definition (CSP CNF Satisfiability)

Input: CSP CNF formula φ on k variables $x_1, \ldots, x_k$ over a finite universe U, atomic sentences $x_i = u$ for $1 \leq i \leq k$, $u \in U$, and a CNF built from these atomic sentences.

Problem: Is φ satisfiable?

Our parameter: k

Constraint Satisfaction Sitting Between

Definition (CSP CNF SATISFIABILITY)

Input:	CSP CNF formula φ on k variables $x_1, \ldots, x_k$ over a finite universe U, atomic sentences $x_i = u$ for $1 \leq i \leq k$, $u \in U$, and a CNF built from these atomic sentences.
Problem:	Is φ satisfiable?

Our parameter: k

Lemma
CSP CNF SATISFIABILITY $\leq_{FPT}$ *LONGEST COMMON SUBSEQUENCE*

Idea: Modify proof of Thm. 3 in Bodlaender et al. 1995

Definition (CSP CNF SATISFIABILITY)

Input: CSP CNF formula φ on k variables $x_1, \ldots, x_k$ over a finite universe U, atomic sentences $x_i = u$ for $1 \leq i \leq k$, $u \in U$, and a CNF built from these atomic sentences.

Problem: Is φ satisfiable?

Our parameter: k

Lemma
CSP CNF SATISFIABILITY $\leq_{FPT}$ LONGEST COMMON SUBSEQUENCE

Idea: Modify proof of Thm. 3 in Bodlaender et al. 1995

Lemma
CSP CNF SATISFIABILITY is $W[2]$-hard.

Idea: Reduce from HITTING SET

Back to Algebra: Permutation Group Factorization

Special case of Monoid Factorization if all f_i are bijections.

Back to Algebra: Permutation Group Factorization

Special case of Monoid Factorization if all f_i are bijections.

$\rightsquigarrow$ Permutation Group Factorization $\in$ W[sync].

Back to Algebra: Permutation Group Factorization

Special case of Monoid Factorization if all f_i are bijections.

$\rightsquigarrow$ Permutation Group Factorization $\in$ W[sync].

Theorem
(Cai et al. 1997) Permutation Group Factorization is W[1]-hard.

Reduction from Perfect Code.

Back to Algebra: Permutation Group Factorization

Special case of Monoid Factorization if all f_i are bijections.

$\rightsquigarrow$ Permutation Group Factorization $\in$ W[sync].

Theorem
(Cai et al. 1997) Permutation Group Factorization is W[1]-hard.

Reduction from Perfect Code.

Similar proof of W[1]-hardness for Sized Subset Sum:
Given positive integers $x_0, x_1, \ldots, x_m$, and integer k, select k integers from $x_1, \ldots, x_m$ that sum up to x_0.

W[Sync] – What Is Outside?

Non-Universality For NFAs

Definition (Bounded NFA Non-Universality)

Input: NFA A with input alphabet Σ, $k \in \mathbb{N}$.

Problem: Is there a word $w \in \Sigma^k$ not accepted by A?

Non-Universality For NFAs

Definition (Bounded NFA Non-Universality)

Input: NFA A with input alphabet Σ, $k \in \mathbb{N}$.

Problem: Is there a word $w \in \Sigma^k$ not accepted by A?

Lemma

Bounded NFA Non-Universality is $W[Sync]$-hard.

Non-Universality For NFAs

Definition (Bounded NFA Non-Universality)

Input: NFA A with input alphabet Σ, $k \in \mathbb{N}$.

Problem: Is there a word $w \in \Sigma^k$ not accepted by A?

Lemma

Bounded NFA Non-Universality is W[Sync]-hard.

Lemma

Bounded NFA Non-Universality $\in A[2]$.

Non-Universality For NFAs

Definition (Bounded NFA Non-Universality)

Input: NFA A with input alphabet Σ, $k \in \mathbb{N}$.

Problem: Is there a word $w \in \Sigma^k$ not accepted by A?

Lemma

Bounded NFA Non-Universality is W[Sync]-hard.

Lemma

Bounded NFA Non-Universality $\in A[2]$.

A[2]-Idea: Guess word $w \in \Sigma^k$, check if all paths through NFA reject.
For W[P]: Do powerset construction on the tape, updating additional $|Q|$ bits when digesting the guessed word.
Open problem with WNL

Bounded NFA-Intersection

Main parameters: k: length of the common string, or $|\mathcal{A}|$

Bounded NFA-Intersection

Main parameters: k: length of the common string, or $|\mathcal{A}|$

Lemma

Bounded NFA-Intersection (wrt. k) is $W[Sync]$-hard.

Lemma

Bounded NFA-Intersection (wrt. k) is in WNL.

Bounded NFA-Intersection

Main parameters: k: length of the common string, or $|\mathcal{A}|$

Lemma

Bounded NFA-Intersection (wrt. k) is $W[\text{Sync}]$-hard.

Lemma
Bounded NFA-Intersection (wrt. k) is in WNL.

Lemma

Bounded NFA-Intersection (wrt. k) is in $A[3]$.

Bounded NFA-Intersection

Main parameters: k: length of the common string, or $|\mathcal{A}|$

Lemma

Bounded NFA-Intersection (wrt. k) is W[Sync]-hard.

Lemma

Bounded NFA-Intersection (wrt. k) is in WNL.

Lemma

Bounded NFA-Intersection (wrt. k) is in A[3].

Open: What about W[P] or A[2]?

Bounded NFA-Intersection

Main parameters: k: length of the common string, or $|\mathcal{A}|$

Lemma

Bounded NFA-Intersection (wrt. k) is W[Sync]-hard.

Lemma
Bounded NFA-Intersection (wrt. k) is in WNL.

Lemma

Bounded NFA-Intersection (wrt. k) is in A[3].

Open: What about W[P] or A[2]?

Theorem
(Hoffmann) Bounded NFA-Intersection (wrt. $|\mathcal{A}|$) is co-W[2]-hard.

A Queer Problem: Small Synchronizable Sub-automata

Definition
DFA-MSS (Minimum synchronizable sub-automaton)]

Input:	DFA A with input alphabet Σ, $k \in \mathbb{N}$.
Problem:	Is there a sub-alphabet $\hat{\Sigma} \subseteq \Sigma$, $\|\hat{\Sigma}\| \leq k$, such that the restriction of A to $\hat{\Sigma}$ is synchonizable, i.e., is there a synchronizing word over $\hat{\Sigma}$?

A Queer Problem: Small Synchronizable Sub-automata

Definition
DFA-MSS (Minimum synchronizable sub-automaton)]

Input:	DFA A with input alphabet Σ, $k \in \mathbb{N}$.
Problem:	Is there a sub-alphabet $\hat{\Sigma} \subseteq \Sigma$, $\lvert\hat{\Sigma}\rvert \leq k$, such that the restriction of A to $\hat{\Sigma}$ is synchonizable, i.e., is there a synchronizing word over $\hat{\Sigma}$?

The NP-hardness proof by Türker and Yenegün (2015) gives:

Corollary
DFA-MSS is W[2]-hard.

A Queer Problem: Small Synchronizable Sub-automata

Definition
DFA-MSS (Minimum synchronizable sub-automaton)]

Input: DFA A with input alphabet Σ, $k \in \mathbb{N}$.

Problem: Is there a sub-alphabet $\hat{\Sigma} \subseteq \Sigma$, $|\hat{\Sigma}| \leq k$, such that the restriction of A to $\hat{\Sigma}$ is synchonizable, i.e., is there a synchronizing word over $\hat{\Sigma}$?

The NP-hardness proof by Türker and Yenegün (2015) gives:

Corollary
DFA-MSS is W[2]-hard.

Theorem
$DFA\text{-}MSS \in WNL \cap W[P]$.

Open: Relation to W[Sync]

Summary

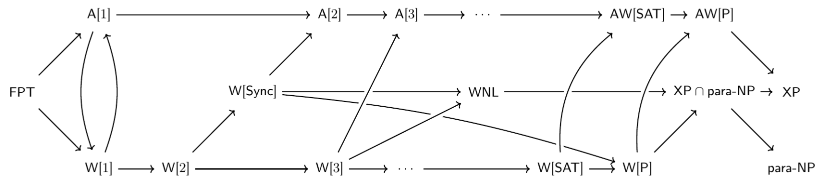


Figure 5: Overview of the complexity classes

Observe: Classes W[Sync] and WNL mostly inhabited by combinatorial formal language problems.

Summary

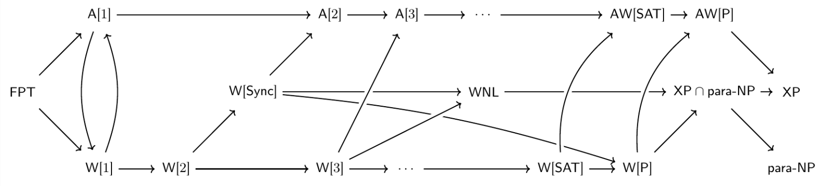


Figure 5: Overview of the complexity classes

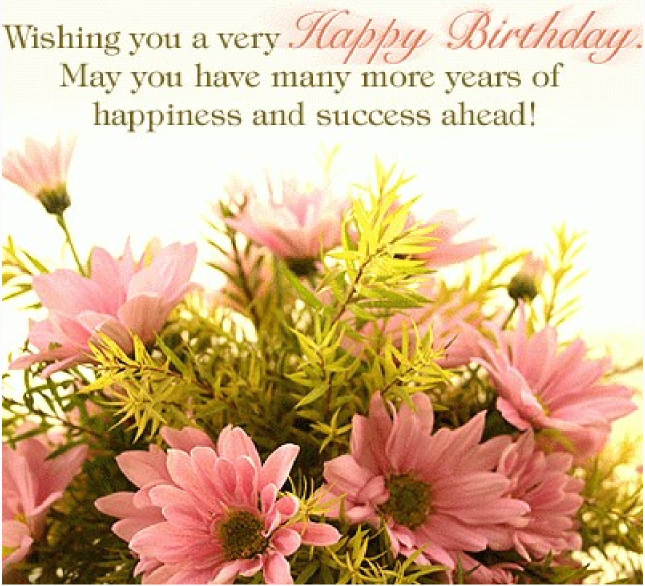
<u>Observe:</u> Classes W[Sync] and WNL mostly inhabited by combinatorial formal language problems.

As W[Sync] $\subseteq$ para-NP, it is "unlikely" that W[Sync] $=$ A[2].

Then, Σ_2^p-problems would as "easy" as SAT, see PhD of R. de Haan (Vienna, 2016).

Still, many questions are open, including positioning the co-W-classes.

Finally . . .


Wishing you a very Happy Birthday.
May you have many more years of
happiness and success ahead!