The Simplest Non-Regular Deterministic Context-Free Language

Jiří Šíma
sim@cs.cas.cz
Institute of Computer Science
Czech Academy of Sciences, Prague, Czechia

joint work with
Petr Jančar
pj.jancar@gmail.com
Department of Computer Science
Palacký University Olomouc, Czechia
**C-Hard Problems**

* C is a complexity class of decision problems (i.e. formal languages)

* A \( \leq \) B is a reduction transforming a problem A to a problem B (a preorder), which is assumed not to have a higher computational complexity than C

* H is a C-hard problem (under the reduction \( \leq \)) if for every \( A \in C \), \( A \leq H \)

- If a C-hard problem has a (computationally) “easy” solution, then each problem in C has an “easy” solution (via the reduction).

- If a C-hard problem H is in C (a so-called C-complete problem), then H belongs to the hardest problems in the class C.
The Most Prominent Example: NP-Hard Problems

$\mathcal{C} = \text{NP}$ is the class of decision problems solvable in polynomial time by a nondeterministic Turing machine.

$A \leq_{P}^{m} B$ is a polynomial-time many-one reduction (Karp reduction) from $A$ to $B$.

The satisfiability problem SAT is NP-hard: for every $A \in \text{NP}$, $A \leq_{P}^{m} \text{SAT}$.

• If an NP-hard problem is polynomial-time solvable, then each NP problem would be solved in polynomial time.

• The NP-hard problem SAT is in NP (i.e. SAT is NP-complete), that is, SAT belongs to the hardest problems (NPC) in the class NP.
**C-Simple Problems**

a conceptual counterpart to C-hard problems:

\( S \) is a \( C \)-simple problem (under the reduction \( \leq \)) if for every \( A \in C \), \( S \leq A \)

- If a \( C \)-simple problem \( S \) proves to be not “easy”, e.g. \( S \) is not solvable by a machine \( M \) (\( M \) can compute the reduction \( \leq \)), then all problems in \( C \) are not “easy”, i.e. \( C \) cannot be solved by \( M \).

→ **a new proof technique:** a lower bound known for one \( C \)-simple problem \( S \) extends to the whole class of problems \( C \)

- If a \( C \)-simple problem \( S \) is in \( C \), then \( S \) is the simplest problem in the class \( C \).

**A Trivial Example:** SAT is simple for the class of NP-hard problems under \( \leq_{P_m} \)
A Nontrivial Example of a $C$-Simple Problem

$C = \text{DCFL'} = \text{DCFL} \setminus \text{REG}$ is the class of non-regular deterministic context-free languages.

$L_1 \leq_{tt}^A L_2$ is a truth-table reduction (a stronger Turing reduction) from $L_1$ to $L_2$ implemented by a Mealy machine with the oracle $L_2$.

The Main Result:

The language $L_# = \{0^n1^n \mid n \geq 1\}$ over the binary alphabet $\{0,1\}$ is $\text{DCFL'}$-simple under the reduction $\leq_{tt}^A$: for every $L \in \text{DCFL'}$, $L_# \leq_{tt}^A L$.

$ightarrow L_# \in \text{DCFL'}$ is the simplest non-regular deterministic context-free languages.

cf. the hardest context-free language $L_0$ due to S. Greibach (1973) is $\text{CFL}$-hard.
Mealy Machines

\( \mathcal{A} \) is a Mealy Machine with an input/output alphabet \( \Sigma / \Delta \)
i.e. a deterministic finite automaton with an output tape:

- **Input tape** with input word \( w \in \Sigma^* \)
- **Initial state** \( q_0 \)
- **Output tape**
Mealy Machines

$A$ is a Mealy Machine with an input/output alphabet $\Sigma/\Delta$

i.e. a deterministic finite automaton with an output tape:

- Current input symbol $a \in \Sigma$
- State transition from $q_1$ to $q_2$
- Output string $u \in \Delta^*$
Mealy Machines

\( \mathcal{A} \) is a Mealy Machine with an input/output alphabet \( \Sigma / \Delta \)
i.e. a deterministic finite automaton with an output tape:

\[ \text{input } w \in \Sigma^* \]

“final” state \( q \)

\[ \text{output } \mathcal{A}(w) \in \Delta^* \]

\[ \text{a deterministic finite-state transducer: } w \in \Sigma^* \longmapsto \mathcal{A}(w) \in \Delta^* \]
The Truth-Table Reduction by Oracle Mealy Machines

$A^{L_2}$ is a Mealy Machine $A$ with an oracle $L_2 \subseteq \Delta^*$:

- For each state $q$ of $A$:
  - $r_q$ suffixes $s_{q,1}, \ldots, s_{q,r_q} \in \Delta^*$
  - Truth table $T_q : \{0, 1\}^{r_q} \rightarrow \{0, 1\}$ with $r_q$ variables

$w \in \Sigma^*$ is accepted by $A^{L_2}$ iff $w$ brings $A$ to the state $q$ such that

$$T_q \left( A(w) \cdot s_{q,1} \in L_2, A(w) \cdot s_{q,2} \in L_2, \ldots, A(w) \cdot s_{q,r_q} \in L_2 \right) = 1$$

$L_1 \leq_{tt} L_2$: $L_1 \subseteq \Sigma^*$ is truth-table reducible to $L_2 \subseteq \Delta^*$ iff

$L_1 = \mathcal{L}(A^{L_2})$ is accepted by some Mealy machine $A^{L_2}$ with oracle $L_2$

Proposition: The relation $\leq_{tt}$ is a preorder.
Why $L_\# = \{0^n1^n \mid n \geq 1\}$ Is the Simplest DCFL’ Language?

any reduced context-free grammar $G$ generating a non-regular language $L \subseteq \Delta^*$

is self-embedding: there is a self-embedding nonterminal $A$ admitting the derivation

$A \Rightarrow^* xAy$ for some non-empty strings $x, y \in \Delta^+$ (Chomsky, 1959)

$G$ is reduced $\longrightarrow S \Rightarrow^* vAz$ and $A \Rightarrow^* w$ for some $v, w, z \in \Delta^*$

$\longrightarrow S \Rightarrow^* vx^mwy^nz \in L$ for every $m \geq 0$ \hspace{1cm} (1)

??? a conceivable (one-one) reduction from $L_\#$ to $L$: for every $m, n \geq 1$,

$0^m1^n \in \{0, 1\}^* \longmapsto vx^mwy^nz \in \Delta^*$

(the inputs outside $0^+1^+$ are mapped onto some fixed string outside $L$)

since $0^m1^n \in L_\#$ implies $vx^mwy^nz \in L$ by (1)

!!! however, the opposite implication may not be true:
Why $L\#$ Is the Simplest DCFL’ language? (cont.)

*** however, the opposite implication may not be true:

for the DCFL' language $L_1 = \{a^m b^n \mid 1 \leq m \leq n\}$ over $\Delta = \{a, b\}$

there are no words $v, x, w, y, z \in \Delta^*$ such that for every $m, n \geq 1$,

$v x^m w y^n z \in L_1$ would ensure $m = n$

nevertheless, already two inputs $a^m b^{n-1} \in L_1$ and $a^m b^n \in L_1$ decides $m ? n$

$\longrightarrow$ the truth-table reduction from $L\#$ to $L_1$ with two queries to the oracle $L_1$:

$0^m 1^n \in \{0, 1\}^* \iff v x^m w y^{n-1} z \in \Delta^*, \ v x^m w y^n z \in \Delta^*$

where $x = a, \ y = b, \ v = w = z = \varepsilon$ is the empty string

satisfying $0^m 1^n \in L\#$ iff $(v x^m w y^{n-1} z \notin L_1 \text{ and } v x^m w y^n z \in L_1)$

this can be generalized to any DCFL’ language $L$:
The Main Technical Result

**Theorem:** Let $L \subseteq \Delta^*$ be a non-regular deterministic context-free language over an alphabet $\Delta$. There exist non-empty words $v, x, w, y, z \in \Delta^+$ and a language $L' \in \{L, \overline{L}\}$ (where $\overline{L} = \Delta^* \setminus L$ is the complement of $L$) such that

1. **either** for all $m, n \geq 0$, $vx^mwy^n z \in L'$ iff $m = n$,
2. **or** for all $m, n \geq 0$, $vx^mwy^n z \in L'$ iff $m \leq n$.

1. 2.

|   | 0   | 1   | 2   | 3   | ... |
|---|-----|-----|-----|-----|-----|
| 0 | $\in L'$ | $\notin L'$ | $\notin L'$ | $\notin L'$ |     |
| 1 | $\notin L'$ | $\in L'$ | $\notin L'$ | $\notin L'$ |     |
| 2 | $\notin L'$ | $\notin L'$ | $\in L'$ | $\notin L'$ |     |
| 3 | $\notin L'$ | $\notin L'$ | $\notin L'$ | $\in L'$ |     |
| ... | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |

In particular, for all $m \geq 0$ and $n > 0$,

$$(vx^mwy^{n-1}z \notin L' \text{ and } vx^mwy^nz \in L') \iff m = n.$$
The Truth-Table Reduction From $L_\#$ to Any DCFL’ $L$

implemented by a Mealy machine $A^L$ with two queries to the oracle $L$:

For any DCFL’ language $L \subseteq \Delta^*$, Theorem provides $v, x, w, y, z \in \Delta^+$ and $L' \in \{L, \overline{L}\}$, say $L' = L$ (analogously for $L' = \overline{L}$), such that

$$(vx^mwy^{n-1}z \notin L \text{ and } vx^mwy^nz \in L) \iff m = n. \quad (2)$$

$A^L$ transforms the input $0^m1^n$ to the output $A(0^m1^n) = vx^mwy^{n-1} \in \Delta^+$ (the inputs outside $0^+1^+$ are rejected), while moving to the state $q$ with $r_q = 2$ suffixes $s_{q,1}, s_{q,2}$ and the truth table $T_q : \{0, 1\}^2 \rightarrow \{0, 1\}$

$$
\begin{array}{l|l|l}
A(0^m1^n) \cdot z & A(0^m1^n) \cdot yz & T_q \\
\hline
vx^mwy^{n-1}z \in L & vx^mwy^nz \in L & \\
0 & 1 & 1 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 0 & 0 \\
\end{array}
$$

It follows from (2) that $L(A^L) = L_\#$, i.e. $L_\# \leq_{tt} A^L$. 11/19
Ideas of the Proof of the Theorem
(inspired by some ideas on regularity of pushdown processes due to Jančar, 2020)

• any non-regular DCFL language $L \subseteq \Delta^*$ is accepted by a deterministic pushdown automaton $\mathcal{M}$ by the empty stack

• since $L \notin \text{REG}$, there is a computation by $\mathcal{M}$, reaching configurations with an arbitrary large stack which is being erased afterwards, corresponding to $v, x, w, y, z \in \Delta^+$ such that $vx^mwynz \in L$ for all $m \geq 1$

• in addition, we aim to ensure that for all $m \geq 0$ and $n > 0$,

  $(vx^mwyn^{-1}z \notin L' \text{ and } vx^mwynz \in L') \iff m = n$

\[ (L' = L) \]

case “$m = n$”: REJ REJ ACC REJ REJ REJ REJ

case “$m \leq n$”: REJ REJ ACC ACC ACC ACC ACC
Ideas of the Proof of the Theorem

(inspired by some ideas on regularity of pushdown processes due to Jančar, 2020)

- any non-regular DCFL language $L \subseteq \Delta^*$ is accepted
  by a deterministic pushdown automaton $M$ by the empty stack

- since $L \notin \text{REG}$, there is a computation by $M$, reaching configurations with
  an arbitrary large stack which is being erased afterwards,
  corresponding to $v, x, w, y, z \in \Delta^+$ such that $vx^mwy^mz \in L$ for all $m \geq 1$

- in addition, we aim to ensure that for all $m \geq 0$ and $n > 0$,
  $(vx^mwy^{n-1}z \notin L' \text{ and } vx^mwy^nz \in L') \iff m = n$

- we study the computation of $M$ on an infinite word that traverses infinitely
  many pairwise non-equivalent configurations

- we use a natural congruence property of language equivalence on the set of
  configurations (determinism of $M$ is essential)

- we apply Ramsey’s theorem for extracting the required $v, x, w, y, z \in \Delta^+$
  from the infinite computation
Basic Properties of DCFL’-Simple Problems

DCFLS is the class of DCFL’-simple problems

Proposition:

- \( \text{REG} \subsetneq \text{DCFLS} \subsetneq \text{DCFL} \),
  - e.g. \( L_\# \in \text{DCFLS} \), \( L_R = \{wcw^R \mid w \in \{a,b\}^*\} \notin \text{DCF} \)

- The class DCFLS is closed under complement and intersection with regular languages.
- The class DCFLS is not closed under concatenation, intersection, and union.
An Application of DCFL’-Simple $L\#$ in Neural Networks
(this application has originally inspired the concept of a DCFL’-simple problem)

The Computational Power of Neural Networks (NNs)
(discrete-time recurrent NNs with the saturated-linear activation function)
depends on the information contents of weight parameters:

- **integer** weights: finite automaton (FA) (Minsky, 1967)
- **rational** weights: Turing machine (TM) (Siegelmann, Sontag, 1995)
  polynomial time $\equiv$ the complexity class $P$
- arbitrary **real** weights: “super-Turing” computation (Siegelmann, Sontag, 1994)
  polynomial time $\equiv$ the nonuniform complexity class $P/poly$
  exponential time $\equiv$ any I/O mapping
- increasing **Kolmogorov complexity** of real weights
  polynomial time $\equiv$ a proper hierarchy of nonuniform complexity classes
  between $P$ and $P/poly$ (Balcázar, Gavaldà, Siegelmann, 1997)

???
the gap in the analysis between realistic **integer** and **rational** weights
w.r.t. Chomsky hierarchy: regular vs. recursively enumerable languages
A Neural Network Model with Increasing Analogicity

from integer to rational weights

\( \alpha \text{ANN} = \) a binary-state NN with integer weights

+ \( \alpha \) extra analog-state neurons with rational weights

\[
\alpha \text{ANN} = \begin{cases} 
\text{binary-state } & \text{NN with integer weights} \\
+ \alpha \text{ extra analog-state } & \text{neurons with rational weights}
\end{cases}
\]

\[
\alpha = 2
\]

\[
w_{ji} \in \begin{cases} 
\mathbb{Q} & j = 1, \ldots, \alpha \\
\mathbb{Z} & j = \alpha + 1, \ldots, s \\
i \in \{0, \ldots, s\}
\end{cases}
\]
A Neural Network Model with Increasing Analogicity
from binary (\{0, 1\}) to analog ([0, 1]) states of neurons

$$\alpha \text{ANN} = \text{a binary-state NN with integer weights} + \alpha \text{ extra analog-state neurons with rational weights}$$

$$y_{j}^{(t+1)} = \sigma_{j} \left( \sum_{i=0}^{s} w_{ji} y_{i}^{(t)} \right) \quad j = 1, \ldots, s$$

updating the states of neurons

$$\sigma_{j}(\xi) = \begin{cases} 
1 & \text{for } \xi \geq 1 \\
\xi & \text{for } 0 < \xi < 1 \\
0 & \text{for } \xi \leq 0 
\end{cases} \quad j = 1, \ldots, \alpha$$

saturated-linear function

$$H(\xi) = \begin{cases} 
1 & \text{for } \xi \geq 0 \\
0 & \text{for } \xi < 0 
\end{cases} \quad j = \alpha + 1, \ldots, s$$

Heaviside function
The Analog Neuron Hierarchy (Šíma, 2019, 2020)

the computational power of NNs increases with the number $\alpha$ of extra analog neurons:

$$F_A \equiv 0_{\text{ANN}} \subseteq 1_{\text{ANN}} \subseteq 2_{\text{ANN}} \subseteq 3_{\text{ANN}} \subseteq \ldots \equiv \text{TM}$$

$\uparrow$ \hspace{1cm} \times \hspace{1cm} \uparrow$

integer weights \hspace{1cm} Chomsky hierarchy \hspace{1cm} rational weights

- $L_\# = \{0^n1^n | n \geq 1\} \not\in 1_{\text{ANN}} \subset \text{CSL}$ (Context-Sensitive Languages)
- $L_1 = \left\{ x_1 \ldots x_n \in \{0, 1\}^* \left| \sum_{k=1}^{n} x_{n-k+1} \left(\frac{3}{2}\right)^{-k} < 1 \right\} \right\} \in 1_{\text{ANN}} \setminus \text{CFL}$
- DCFL $\subset 2_{\text{ANN}}$
- $3_{\text{ANN}} \equiv \text{TM}$
The Technique of Expanding a Lower Bound

- $L_\# \notin 1\text{ANN}$ with a nontrivial proof (based on the Bolzano–Weierstrass theorem) which can hardly be generalized to another DCFL’ language
- $L_\#$ is DCFL’-simple under $\leq_{\text{tt}}$
- the reduction $\leq_{\text{tt}}$ to any $L \in 1\text{ANN}$ can be implemented by $1\text{ANN}$

$\rightarrow$ the known lower bound $L_\# \notin 1\text{ANN}$ for a single DCFL’-simple problem $L_\#$ is extended to the whole class $\text{DCFL'} \cap 1\text{ANN} = \emptyset$

Comments:

- If any DCFL’ language proves to be CFL’-simple, then $\text{CFL'} \cap 1\text{ANN} = \emptyset$.
- $L_\#$ is not CSL’-simple since $L_\# \leq_{\text{tt}} L_1 \in 1\text{ANN}$ would imply $L_\# \in 1\text{ANN}$
A Summary

• We have introduced a new notion of $\mathcal{C}$-simple problems which is a conceptual counterpart to $\mathcal{C}$-hard problems.

• We have shown $L_\# = \{0^n1^n \mid n \geq 1\}$ to be a DCFL’-simple problem under the truth-table reduction by oracle Mealy machines:

$$\rightarrow L_\# \text{ is the simplest DCFL’ problem}$$

• We have proposed a new proof technique of expanding a lower bound known for a single $\mathcal{C}$-simple problem to the whole class of problems $\mathcal{C}$, which has been illustrated by a nontrivial application to the analysis of neural networks:

DCFL’-simple $L_\# \notin 1ANN \rightarrow \text{DCFL’} \cap 1ANN = \emptyset$

Open Problems

• Is $L_\#$ CFL’-simple or UCFL’-simple (Unambiguous CFL’)?

$$(\rightarrow (U)\text{CFL’} \cap 1ANN \neq \emptyset)$$

• Examples of nontrivial $\mathcal{C}$-simple problems for other complexity classes $\mathcal{C}$ under suitable reductions?