Non-regular complexity

Szilárd Zsolt Fazekas[1]

Akita University

One FLAT World Seminar
April 10, 2024

[1] Supported by JSPS Kakenhi Grant 23K10976

“Relative complexity”

Given computational model $\mathcal{A}$, model $\mathcal{B}$ is an extension of $\mathcal{A}$ if comp. steps possible in $\mathcal{A}$ are also possible in $\mathcal{B}$, and $\mathcal{B}$ allows some operations not available $\mathcal{A}$.

Operations available in the extensions but not in the original model are a computational resource and can be analyzed quantitatively. The used amount of this ‘extra’ resource can be thought of as the complexity of a system from $\mathcal{B}$ relative to model $\mathcal{A}$.

Complexity measure

Let $C(w)$ be the computation (derivation, run of automaton) of a system M of type $\mathcal{B}$ for some input $w \in L(M)$:

$$C(w) : c_1 \vdash c_2 \vdash \cdots \vdash c_n.$$

$$notA_M^{\theta}(w) = |\{i \mid c_i \vdash c_{i+1} \text{ uses an operation } \theta \text{ not available in } \mathcal{A}\}|$$

$$notA_M^{\theta}(n) = \max_{|w|=n} \{notA(w)\}$$

For $w \notin L(M)$ and for n such that no word of length n is in $L(M)$, the measures are set to 0.

Non-regular complexity

Here we focus on cases when $\mathcal{A}$ is a model which generates/accepts regular languages.

- $\mathcal{A} =$ regular grammars, $\mathcal{B} =$ context-free grammars
- $\mathcal{A} =$ FA, $\mathcal{B} =$ one-way jumping automata
- $\mathcal{A} =$ FA, $\mathcal{B} =$ automata with translucent letters

Questions

1. Where is the boundary of regularity?
2. Are there systems/languages with intermediate complexity, i.e., more than minimal (constant) and less than maximal (typically linear)?
3. Is the complexity of a given system/language decidable?

$\mathcal{A}$= regular grammars, $\mathcal{B}$ = context-free grammars

On the degrees of non-regularity and non-context-freeness [Bordihn and Mitrana, '20]

Count the number of non-regular production rules used in the derivations (degree of non-regularity).

$dnreg_G(w, D) =$ number of non-regular steps in derivation D of w.

$$dnreg_G(w) = \begin{cases} \min\{dnreg_G(w, D) \mid D \text{ is a derivation of } w\} & w \in L(G) \\ 0 & w \notin L(G) \end{cases}$$

$$dnreg_G(n) = \max\{dnreg_G(w) \mid |w| = n\}$$

$$DNREG(f(n)) = \{L \mid L = L(G) \text{ for CFG } G \text{ with } dnreg_G(n) \in O(f(n))\}$$

$\mathcal{A}$= regular grammars, $\mathcal{B}$ = context-free grammars

- DNREG(1)=REG.
- For any context-free grammar G and positive integer c, it is decidable whether $dnreg_G(n) \leq c$.
- Given an unambiguous context-free grammar G, one can algorithmically decide whether $dnreg_G(n) \in O(1)$.
- Given a linear context-free grammar G, it is undecidable whether $dnreg_G(n) \in O(1)$.

$\mathcal{A}$= regular grammars, $\mathcal{B}$ = context-free grammars

- CF=DNREG(n).
- For every deterministic context-free grammar G with $L(G)$ non-regular, $dnreg_G(n) \in \Omega(n)$.
- DNREG($\sqrt{n}$)\DNREG(1)$\neq \emptyset$ and DNREG($\log n$)\DNREG(1)$\neq \emptyset$.

Probably DNREG(n)\DNREG($f(n)$)$\neq \emptyset$, for any sublinear function $f(n)$ (language of palindromes...)

$\mathcal{A}$= DFA, $\mathcal{B}$= one-way jumping DFA

$M = (Q, \Sigma, R, s, F)$, as in a (partially defined) DFA.

Elements of R are transition rules $\mathbf{p}a \to \mathbf{q} \in R$

Configurations of M are strings in $Q\Sigma^*$.

A $\circlearrowright_R$**DFA** transition ($\vdash$) can be :

(i) $\mathbf{p}ax \Rightarrow \mathbf{q}x,$ if $\mathbf{p}a \to \mathbf{q} \in R$ (*sequential trans.*) or

(ii) $\mathbf{p}yax \circlearrowright \mathbf{q}xy,$ when $y \in (\Sigma \setminus \Sigma_p)^*, \mathbf{p}a \to \mathbf{q}$. (*a jump*)

$L(M) = \{x \in \Sigma^* \mid \exists \mathbf{f} \in F : \mathbf{s}x \vdash^* \mathbf{f}\}$.

Example

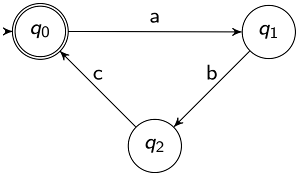
Let M be a $\circlearrowright_R$DFA given by

$$M = (\{\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2\}, \{a, b, c\}, R, \mathbf{q}_0, \{\mathbf{q}_0\}),$$

where R consists of the rules $\mathbf{q}_0 a \to \mathbf{q}_1$, $\mathbf{q}_1 b \to \mathbf{q}_2$ and $\mathbf{q}_2 c \to \mathbf{q}_0$.

Accepted language is $\{w \in \{a, b, c\}^* \mid |w|_a = |w|_b = |w|_c\}$

$$\mathbf{q}_0 acbcab \vdash \mathbf{q}_1 bcabc \vdash \mathbf{q}_2 cabc \vdash \mathbf{q}_0 abc \vdash \mathbf{q}_1 bc \vdash \mathbf{q}_2 c \vdash \mathbf{q}_0$$



Accepting power

- **REG** $\subsetneq$ $\circlearrowright_R$**DFA**.
- **CF** and $\circlearrowright_R$**DFA** are incomparable.
- $\circlearrowright_R$**DFA** $\subsetneq$ **CS**.
- $\circlearrowright_R$**DFA** $\subseteq$ DTIME(n^2).

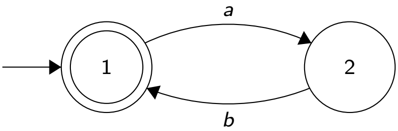


Figure: $\circlearrowright_R$**DFA** $\mathcal{A}$ accepting $\{w \mid |w|_a = |w|_b\}$.

Sweeps:

position :	0	1	2	3	4	5	6	7
input	**a**	**a**	**a**	**a**	**b**	**b**	**b**	**b**
after sweep 1	ε	**a**	**a**	**a**	ε	**b**	**b**	**b**
after sweep 2	ε	ε	**a**	**a**	ε	ε	**b**	**b**
after sweep 3	ε	ε	ε	**a**	ε	ε	ε	**b**
after sweep 4	ε	ε	ε	ε	ε	ε	ε	ε

Figure: The computation table for $a^4 b^4$ by $\mathcal{A}$.

Sweep complexity [F. et al., 2022]

The *jump complexity* (*sweep complexity*) of an automaton M is $jc_M(n)$ ($sc_M(n)$) is the maximum number of jumps (sweeps) that M makes on processing inputs $w \in L(M)$ of length n.

$JUMP(f(n))$ ($\mathrm{SWEEP}(f(n))$) is the class of languages accepted by $\circlearrowright_R$DFA with $jc_M(n)$ ($sc_M(n)$) in $\mathcal{O}(f(n))$.

Jump complexity

Jump complexity is between $O(1)$ and $O(n)$, but we do not know more in the general case.[2]

For jumps of limited length, we have machines with jump complexity $\Theta(\log n)$.

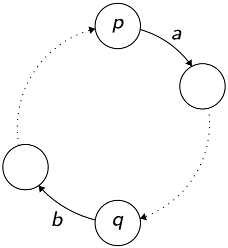
[2] By the slightly different definition used in FMW'22, it can be $O(n^2)$

Sweep complexity

For any $\circlearrowright_R$**DFA** $\mathcal{A}$ and any constant k, the set of words accepted by $\mathcal{A}$ in at most k sweeps is regular. [F., Yamamura, 2016]

Lemma (F., Mercaș, Wu, 2022)

If a $\circlearrowright_R$**DFA** *has superconstant sweep complexity, then it has two reachable and co-reachable states* $\mathbf{p}$ *and* $\mathbf{q}$ *such that* $\mathbf{p}$ *is* a*-deficient,* $\mathbf{q}$ *is* b*-deficient, for some* $a, b \in \Sigma$ *with* $a \neq b$*, and* $\mathbf{p}buav \vdash^* \mathbf{q}av \vdash^* \mathbf{p}$*, for some* $u, v \in \Sigma^*$*.*



Logarithmic complexity

Sweep complexity revisited [F., Mercaș, 2023]

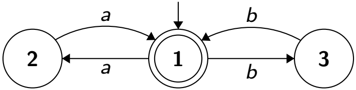


Figure: $L(\mathcal{B}) = \{w \in \{a,b\}^* \mid |w|_a \equiv 0 \bmod 2, |w|_b \equiv 0 \bmod 2\}$.

$L(\mathcal{B})$ is regular.

The sweep complexity of $\mathcal{B}$ is $\Theta(\log n)$.

Linear complexity

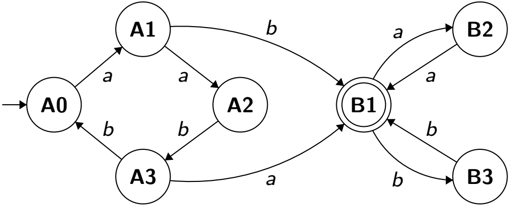
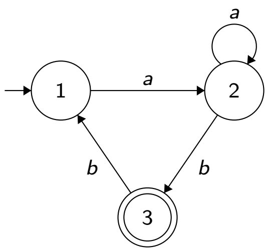


Figure: $L(\mathcal{C}) = \{w \in \{a, b\}^* \mid |w|_a \equiv 1 \mod 2, |w|_b \equiv 1 \mod 2\}$

$L(\mathcal{C})$ is regular.

The sweep complexity of $\mathcal{C}$ is $\Theta(n)$.

Non-REG language accepted with sublinear sweep complexity



The $\circlearrowright_R$**DFA** $\mathcal{D}$ accepts a non-regular language.

The sweep complexity of $\mathcal{D}$ is $\Theta(\log n)$.

Separating complexity classes

$\mathrm{SWEEP}(1) \subsetneq \mathrm{SWEEP}(\log n)$.

Any automaton which accepts $L_{ab} = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$ has sweep complexity $\Theta(n)$.

If $f : \mathbb{N} \Rightarrow \mathbb{N}$ with $f(n) \in o(n)$ then $\mathrm{SWEEP}(f(n)) \subsetneq \mathrm{SWEEP}(n)$.

$\mathcal{A}$= NFA, $\mathcal{B}$= NFA with translucent letters

Jump complexity of finite automata with translucent letters
[Mitrana, Păun, Păun, Sanchez Couso, 2024]

$M = (Q, \Sigma, R, s, F)$, as in a (partially defined) DFA.

Transition rules are $\mathbf{p}a \to \mathbf{q} \in R$, configurations are strings in $Q\Sigma^*$.

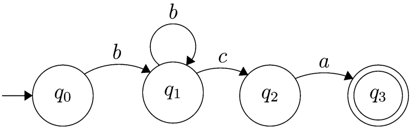
A transition can be either:

(i) $\mathbf{p}ax \Rightarrow \mathbf{q}x,$ if $\mathbf{p}a \to \mathbf{q} \in R$ (*sequential trans.*) or

(ii) $\mathbf{p}xay \circlearrowright \mathbf{q}xy,$ when $x \in (\Sigma \setminus \Sigma_p)^*, \mathbf{p}a \to \mathbf{q}$. (*a jump*)

Jump complexity

- Given an NFATL M and a positive integer c, the language the language accepted by M with at most c jumps ($L(M, c)$) is regular.
- There are NFATL with $\Omega(n)$ jump complexity accepting regular languages.



- Any NFATL accepting $L = \{w \mid |w|_a - |w|_b \in \{0, 1\}\}$ has jump complexity $\Omega(n)$, so $JCL(n) \setminus JCL(f(n)) \neq \emptyset$ for any sublinear function $f(n)$.
- $JCL(\log n) \setminus JCL(1) \neq \emptyset$.

$\mathcal{A}$= DFA, $\mathcal{B}$= DFA with translucent letters

- Ongoing work with MPPC
- It looks like there is a gap between $JCL(1)$ and $JCL(n)$
- In some cases complexity is probably decidable

Overall

- $O(1)$ complexity implies that the language generated/accepted is regular
- in some cases the hierarchy collapses to $O(1)$ and $O(n)$, but at least in the nondeterministic case there are intermediate classes
- $O(1)$ (and maybe $O(n)$) complexity is decidable for some models

What is next?

- Are there machines with arbitrary (constructible) sublinear complexity ($\Theta(\log^k n)$ and $\Theta(n^\epsilon)$)?
- Is it decidable, given a machine or language and a function $f(n)$, whether the machine/language has $\Theta(f(n))$ sweep complexity?
- Investigate similar models $\rightarrow$ General framework for 'non-regular' complexity

General framework

- Perhaps a single tape Turing machine, or a model like iterated finite transducers

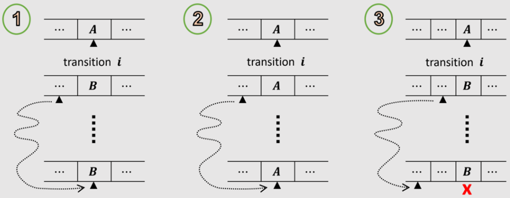


Figure. Three types of transitions (i):

1. **non-regular step**: the cell content changed and the head returns to it
2. **regular step**: the content did not change
3. **regular step**: the content changed, but the head does not return to read it

- Parameterise the complexity classes by number of rewrites allowed per position, per sweep, in total…

Thank you!

F., S. Z., Mercaș, R., and Wu, O. (2022).
Complexities for jumps and sweeps.
Journal of Automata, Languages and Combinatorics, 27(1-3):131–149.