An in-between "implicit" and "explicit" complexity: Automata

Kick-off Elica

Clément Aubert

LIPN — Paris 13
6 February 2015

> classes. By implicit, we here mean that classes are not given by constraining the amount of resources a *machine* is allowed to use, but rather by imposing linguistic constraints on the way *algorithms* are formulated. This idea has de-

(Dal Lago, 2011, p. 90)

Question

Is there an in-between?

Answer

One can also restrict the *quality* of the resources.

ICC is

machine-independent **and** without explicit bounds.

Machine-dependant

Turing machine,
Random access machine,
Counter machine, . . .

Machine-dependant

Turing machine,
Random access machine,
Counter machine, . . .

Machine-independant

Bounded recursion on notation (Cobham, 1965),
Bounded linear logic (Girard et al., 1992),
Bounded arithmetic (Buss, 1986), . . .

Machine-dependant	**Machine-independant**
Turing machine, Random access machine, Counter machine, . . .	Bounded recursion on notation (Cobham, 1965), Bounded linear logic (Girard et al., 1992), Bounded arithmetic (Buss, 1986), . . .

The rules for storage naturally induce polynomials:

Storage $\dfrac{!_{\vec{y}}\Gamma \vdash A}{!_{x\vec{y}}\Gamma \vdash !_x A}$ *Weakening* $\dfrac{\Gamma \vdash B}{\Gamma, !_0 A \vdash B}$

Contraction $\dfrac{\Gamma, !_x A, !_y A \vdash B}{\Gamma, !_{x+y} A \vdash B}$ *Dereliction* $\dfrac{\Gamma, A \vdash B}{\Gamma, !_1 A \vdash B}$.

(Girard et al., 1992, p. 18)

	Machine-dependant	Machine-independant
Explicit bounds	Turing machine, Random access machine, Counter machine, . . .	Bounded recursion on notation (Cobham, 1965), Bounded linear logic (Girard et al., 1992), Bounded arithmetic (Buss, 1986), . . .

	Machine-dependant	Machine-independant
Explicit bounds	Turing machine, Random access machine, Counter machine, . . .	Bounded recursion on notation (Cobham, 1965), Bounded linear logic (Girard et al., 1992), Bounded arithmetic (Buss, 1986), . . .
Implicit bounds		Descriptive complexity (Fagin, 1973), Recursion on notation (Bellantoni and Cook, 1992), Tiered recurrence (Leivant, 1993), . . .

	Machine-dependant	**Machine-independant**
Explicit bounds	Turing machine, Random access machine, Counter machine, . . .	Bounded recursion on notation (Cobham, 1965), Bounded linear logic (Girard et al., 1992), Bounded arithmetic (Buss, 1986), . . .
Implicit bounds	Automaton, Auxiliary pushdown machine, (Boolean circuit,) . . .	Descriptive complexity (Fagin, 1973), Recursion on notation (Bellantoni and Cook, 1992), Tiered recurrence (Leivant, 1993), . . .

	Machine-dependant	**Machine-independant**
Explicit bounds	Turing machine, Random access machine, Counter machine, . . .	Bounded recursion on notation (Cobham, 1965), Bounded linear logic (Girard et al., 1992), Bounded arithmetic (Buss, 1986), . . .
ounds	Automaton, Auxiliary pushdown machine,	Descriptive complexity (Fagin, 1973), Recursion on notation (Bellantoni and Cook,

related to the foregoing question. More specifically, we have attempted to characterize several tape and time complexity classes of Turing machines in terms of devices whose definitions involve only ways in which their infinite memory may be manipulated and no restrictions are imposed on the amount of memory that they use. The basic model

(Ibarra, 1971, p. 88)

Definition (2NFA(k, j))

For $k \geqslant 1$, $j \geqslant 0$, a 2-*way non-deterministic finite automaton with* k*-heads and* j *pushdown stacks* is a tuple $M = \{\mathbf{S}, A, B, \rhd, \lhd, \boxdot, \sigma\}$ where:

- $\mathbf{S}$ is the finite set of *states*;
- A is the *input alphabet*, B is the *stack alphabet*;
- $\rhd$ and $\lhd$ are the *left* and *right endmarkers*, $\rhd, \lhd \notin A$;
- $\boxdot$ is the *bottom symbol of the stack*, $\boxdot \notin B$;
- $\sigma \subseteq (\mathbf{S} \times (A \cup \{\rhd, \lhd\})^k \times (B \cup \{\boxdot\})^j) \times (\mathbf{S} \times \{-1, 0, +1\}^k \times \{\texttt{pop}, \texttt{peek}, \texttt{push}(b)\}^j)$

$\mathbf{2NFA(k, l)} = \{\mathcal{L}(M) \mid M \text{ a } 2\mathrm{NFA}(k, j)\}$

$\mathbf{2NFA(*, l)} = \cup_{k \geqslant 1} \mathbf{2NFA(k, j)}$

Theorem (Main characterizations)

Automata	***Language / Predicate***
$\mathbf{2NFA}(\mathbf{1},\mathbf{2})$	*Computable*
$\mathbf{2NFA}(*,\mathbf{1})$	*Polynomial time*
$\mathbf{2NFA}(*,\mathbf{0})$	*Logarithmic space*
$\mathbf{2NFA}(\mathbf{1},\mathbf{1})$	*Context-free*
$\mathbf{2NFA}(\mathbf{1},\mathbf{0})$	*Regular*

$\mathbf{2NFA}(\mathbf{1},\mathbf{2}) \supseteq$ Computable.

Let P be computable

Let M be a Turing Machine that computes it

Take M' with 1 read-only head and 1 read-write tape

Simulate the content of the tape with the pushdown tapes

□

$\mathbf{2NFA}(\mathbf{1},\mathbf{2}) \subseteq$ Computable.

Finite automata are restrictions of Turing Machines. □

2NFA$(*, \mathbf{1}) \supseteq$ **Polynomial Time.**

Restrict the Turing Machine. □

2NFA$(*, \mathbf{1}) \subseteq$ **Polynomial Time.**

A plain simulation? □

2NFA$(*, 1) \supseteq$ Polynomial Time.

Restrict the Turing Machine. □

2NFA$(*, 1) \subseteq$ Polynomial Time.

A plain simulation? □

> In fact, given a 2N PDA (or 2D PDA) P, it is possible to effectively design from P a 2N PDA (or 2D PDA) P', which in addition to stimulating the behavior of P scanning any input string $w = a_2 \cdots a_{n-1}$, also counts from 1 to 2^n between each move made by P. The number of

Aho, Hopcroft, and Ullman, 1968, p. 197

Your attention please

Memoization will be explained on the white board.

Your attention please

Memoization will be explained on the white board.

Some FA with exponential runs can be simulated in linear time (Cook, 1971).

$\mathbf{2NFA}(*, \mathbf{0}) \subseteq$ Logarithmic space.

Write and update the heads' addresses in log-space. □

$\mathbf{2NFA}(*, \mathbf{0}) \supseteq$ Logarithmic space.

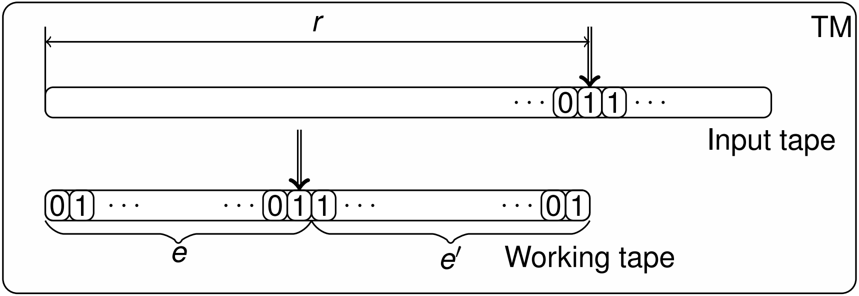
Encode the content of the tapes as positions (tallies). □

Remark (Hofmann and Schöpp, 2010)

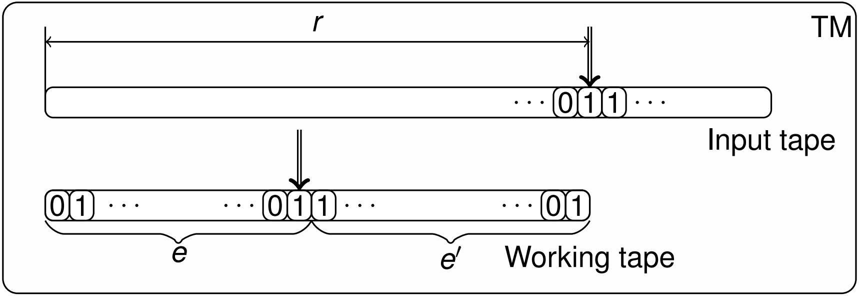
The input tape *must* be ordered.

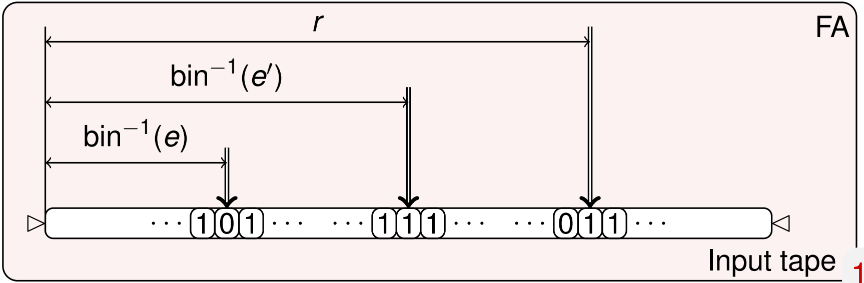
Reminder

A binary string of length $\log(|n|)$ cannot express an integer greater than $|n|$.



TM
r
· · · 0 1 1 · · ·
Input tape
0 1 · · · · · · 0 1 1 · · · · · · 0 1
e
e′
Working tape


r
TM
· · · 0 1 1 · · ·
Input tape
0 1 · · · · · · 0 1 1 · · · · · · 0 1
e
e′
Working tape


r
FA
$\mathrm{bin}^{-1}(e')$
$\mathrm{bin}^{-1}(e)$
· · · 1 0 1 · · · · · · 1 1 1 · · · · · · 0 1 1 · · ·
Input tape

$\mathbf{2NFA}(\mathbf{1}, \mathbf{1})$ = Context-free
and
$\mathbf{2NFA}(\mathbf{1}, \mathbf{0})$ = Regular.

So well-known it became the definition. □

$\mathbf{2NFA}(\mathbf{1}, \mathbf{1})$ = Context-free
and
$\mathbf{2NFA}(\mathbf{1}, \mathbf{0})$ = Regular.

So well-known it became the definition. □

FA exist in deterministic and 1-way variants.

$\mathbf{2NFA}(\mathbf{1},\mathbf{1})$ = Context-free
and
$\mathbf{2NFA}(\mathbf{1},\mathbf{0})$ = Regular.

So well-known it became the definition. □

FA exist in deterministic and 1-way variants.

Theorem (Equalities)

$$\mathbf{2NFA}(*,\mathbf{0}) = \mathbf{NL}$$

$$\mathbf{2DFA}(*,\mathbf{0}) = \mathbf{L}$$

$$\mathbf{2NFA}(*,\mathbf{1}) = \mathbf{2DFA}(*,\mathbf{1}) = \mathbf{P}$$

$$\mathbf{1DFA}(\mathbf{1},\mathbf{0}) = \mathbf{2DFA}(\mathbf{1},\mathbf{0}) = \mathbf{1NFA}(\mathbf{1},\mathbf{0}) = \mathbf{2DFA}(\mathbf{1},\mathbf{0})$$

Theorem (Inequalities)

$k > 1, j \leqslant 1$

$\_\_\mathbf{FA}(\mathbf{k}, \mathbf{j}) \subsetneq \_\_\mathbf{FA}(\mathbf{k}+\mathbf{1}, \mathbf{j})$	(Too many to list)
$\mathbf{1DFA}(\mathbf{k}, \mathbf{0}) \subsetneq \mathbf{2DFA}(\mathbf{k}, \mathbf{0})$	(Holzer et al., 2008)
$\mathbf{1NFA}(\mathbf{k}, \mathbf{0}) \subsetneq \mathbf{2NFA}(\mathbf{k}, \mathbf{0})$	(")
$\mathbf{1DFA}(\mathbf{k}, \mathbf{0}) \subsetneq \mathbf{1NFA}(\mathbf{2}, \mathbf{0})$	(Yao and Rivest, 1978)
$\mathbf{1NFA}(\mathbf{k}, \mathbf{0}) \subseteq \mathbf{2DFA}(\mathbf{k}, \mathbf{0})$	(Sudborough, 1977b)
$\mathbf{2NFA}(\mathbf{k}, \mathbf{0}) \subseteq \mathbf{2NFA}(\mathbf{2} \times \mathbf{k}, \mathbf{1})$	(")
$\mathbf{2NFA}(\mathbf{k}, \mathbf{1}) \subseteq \mathbf{2DFA}(\mathbf{4} \times \mathbf{k}, \mathbf{1})$	(Sudborough, 1977a)
$\mathbf{1DFA}(\mathbf{k}, \mathbf{1}) \subsetneq \mathbf{1NFA}(\mathbf{k}, \mathbf{1})$	(Chrobak, 1986)
$\mathbf{1DFA}(\mathbf{k}, \mathbf{1}) \subsetneq \mathbf{1NFA}(\mathbf{k}, \mathbf{1})$	(")
$\mathbf{NL} = \mathbf{L}$ *iff* $\mathbf{1NFA}(\mathbf{2}, \mathbf{0}) \subseteq \mathbf{2DFA}(\mathbf{k}, \mathbf{0})$	(Sudborough, 1973)

Automata are

- well-studied[1];
- closely related to complexity;
- a place where many fundamental techniques have been discovered;
- *implicit* in an *unexpected* way;
- *explicit* in an *expected* way.

The *quality* of their storage impacts directly on their expressive power.

[1] Over-studied?

Inspiration to formalize computation in GoI (Aubert and Seiller, 2014; Aubert and Seiller, 2015).

Theorem (Aubert, Bagnol, Pistone, and Seiller, 2014)

"Logic programs where all the variables are at the same height characterize Logarithmic space."

Theorem (Aubert, Bagnol, and Seiller, 2015)

"Logic programs using only unary functions characterize polynomial time."

Inspiration to formalize computation in GoI (Aubert and Seiller, 2014; Aubert and Seiller, 2015).

Theorem (Aubert, Bagnol, Pistone, and Seiller, 2014)

"Logic programs where all the variables are at the same height characterize Logarithmic space."

Theorem (Aubert, Bagnol, and Seiller, 2015)

"Logic programs using only unary functions characterize polynomial time."

Thanks!

Aho, Alfred V., John E. Hopcroft, and Jeffrey D. Ullman (1968). “Time and Tape Complexity of Pushdown Automaton Languages”. In: *Inform. Control* 13.3, pp. 186–206.

Aubert, Clément, Marc Bagnol, and Thomas Seiller (2015). *Memoization for Unary Logic Programming: Characterizing Ptime*. Research Report, submitted to LICS 2015, p. 12. arXiv: `1501.05104 [cs.LO]`.

Aubert, Clément and Thomas Seiller (2014). “Characterizing co-NL by a group action”. In: *Mathematical Structures in Computer Science* FirstView, pp. 1–33.

– (2015). “Logarithmic Space and Permutations”. In: *Information and Computation*. to appear.

- Aubert, Clément, Marc Bagnol, Paolo Pistone, and Thomas Seiller (2014). "Logic Programming and Logarithmic Space". In: *APLAS*. Ed. by Jacques Garrigue. Vol. 8858. LNCS. Springer, pp. 39–57.
- Bellantoni, Stephen J. and Stephen Arthur Cook (1992). "A New Recursion-Theoretic Characterization of the Polytime Functions (Extended Abstract)". In: *STOC*. Ed. by S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis. ACM, pp. 283–93.
- Buss, Samuel R. (1986). *Bounded Arithmetic*. Ed. by Bibliopolis. Vol. 3. Studies in Proof Theory. Lecture Notes.
- Chrobak, Marek (1986). "Hierarchies of One-Way Multihead Automata Languages". In: *Theoret. Comput. Sci.* 48.3, pp. 153–181.

- Cobham, Alan (1965). "The intrinsic computational difficulty of functions". In: *Logic, methodology and philosophy of science: Proceedings of the 1964 international congress held at the Hebrew university of Jerusalem, Israel, from August 26 to September 2, 1964*. Ed. by Yehoshua Bar-Hillel. Studies in Logic and the foundations of mathematics. North-Holland Publishing Company, pp. 24–30.
- Cook, Stephen Arthur (1971). "Linear Time Simulation of Deterministic Two-Way Pushdown Automata". In: *IFIP Congress (1)*. Ljubljana, Yugoslavia, August 23-28, 1971: North-Holland, pp. 75–80.
- Dal Lago, Ugo (2011). "A Short Introduction to Implicit Computational Complexity". In: *ESSLLI*. Ed. by Nick Bezhanishvili and Valentin Goranko. Vol. 7388. LNCS. Springer, pp. 89–109.

- Fagin, Ronald (1973). "Contributions to the Model Theory of Finite Structures". PhD thesis. University of California, Berkeley.
- Girard, Jean-Yves, Andre Scedrov, and Philip J. Scott (1992). "Bounded linear logic: a modular approach to polynomial-time computability". In: *Theoret. Comput. Sci.* 97.1, pp. 1–66.
- Hofmann, Martin and Ulrich Schöpp (2010). "Pure Pointer Programs with Iteration". In: *ACM Transactions on Computational Logic* 11.4.
- Holzer, Markus, Martin Kutrib, and Andreas Malcher (2008). "Multi-Head Finite Automata: Characterizations, Concepts and Open Problems". In: *CSP*. Ed. by Turlough Neary, Damien Woods, Anthony Karel Seda, and Niall Murphy. Vol. 1. EPTCS, pp. 93–107.

Ibarra, Oscar H. (1971). "Characterizations of Some Tape and Time Complexity Classes of Turing Machines in Terms of Multihead and Auxiliary Stack Automata". In: *J. Comput. Syst. Sci.* 5.2, pp. 88–117.

Leivant, Daniel (1993). "Stratified Functional Programs and Computational Complexity". In: *POPL*. Ed. by Mary S. Van Deusen and Bernard Lang. Charleston, South Carolina, USA: ACM Press, pp. 325–333.

Sudborough, Ivan Hal (1973). "On Tape-Bounded Complexity Classes and Multi-Head Finite Automata". In: *SWAT (FOCS)*. IEEE Computer Society, pp. 138–144.

– (1977a). "Separating Tape Bounded Auxiliary Pushdown Automata Classes". In: *STOC*. Ed. by John E. Hopcroft, Emily P. Friedman, and Michael A. Harrison. ACM, pp. 208–217.

- Sudborough, Ivan Hal (1977b). "Some Remarks on Multihead Automata". In: *Theor. Inform. Appl.* 11.3, pp. 181–195.
- Yao, Andrew Chi-Chih and Ronald L. Rivest (1978). "$k + 1$ Heads Are Better Than k". In: *J. ACM* 25.2, pp. 337–340.