Computational Complexity and Logical Definability

or

« How logic can help
to *discover efficient generic algorithms*
or to *understand complexity classes* »

Etienne Grandjean, GREYC
CNRS / Université de Caen / ENSICAEN

Journée LIPN du 5 juillet 2011
en l'honneur de Christian Lavault

A natural question

- What relationships exist between
  - Algorithmic complexity
    - How a problem PB is difficult to solve?

and

  - Descriptive complexity
    - How the problem PB is difficult to define?

Two kinds of results

- Algorithmic meta theorems
  - Or « generic algorithms of small complexity for solving a class of problems definable in some logic »
- Logical characterizations of complexity classes
  - Or « complexity class = logically definable class »

The logics involved

- First-order logic: FO
- Second-order logic: SO
  - And its restrictions
    - Existential second-order: ESO
    - Monadic second-order: MSO
    - Existential monadic second-order: EMSO

Examples: First-Order Logic

- The fact that a graph G = (V,E) satisfies the First-Order (FO) sentence

  $\exists x \; \exists y \; \exists z \;\; E(x,y) \wedge E(y,z) \wedge E(z,x)$

means that

  G contains a triangle

- We say that the problem TRIANGLE is defined in First-Order logic and denote
  - TRIANGLE $\in$ FO
- *Notice:* each problem defined in FO is computable in PTIME. Why?

Examples: Second-Order Logic

- The fact that a graph G = (V,E) satisfies the Second-Order (SO) sentence

  $\forall U$

  « Any set U of vertices »

  $([\exists x\ U(x) \wedge \forall x \forall y\ ((U(x) \wedge E(x,y)) \rightarrow U(y))]$

  « that is nonempty and closed for neighbours »

  $\rightarrow \forall x\ U(x))$

  « contains all the vertices of G »

means that G is connex

- We say that problem CONNEX is defined in SO (and MSO) logic, denoted CONNEX $\in$ SO and CONNEX $\in$ MSO
- At the opposite: CONNEX $\notin$ FO

More examples in SO

- **Problem 3-COL**
  - *Input:* a graph G=(V,E)
  - *Question:* Can G be coloured with 3 colours?
- Problem 3-COL is defined by the Existential Second-Order (ESO) sentence

  ∃ C1 ∃ C2 ∃ C3

  $[\ \forall x\ (C1(x) \vee C2(x) \vee C3(x)) \wedge$ the Ci's are pairwise disjoint

  $\wedge\ \forall x\ \forall y\ \bigwedge_i\ (Ci(x) \wedge E(x,y)) \rightarrow \neg\ Ci(y)\ ]$

Therefore 3-COL ∈ ESO

and 3-COL ∈ EMSO (Existential Monadic Second-Order)

More examples in ESO

- **Problem HAMILTON**
  - *Input:* a graph G=(V,E)
  - *Question:* has G a Hamiltonian path?
- Problem HAMILTON is defined by the Existential Second-Order (ESO) sentence

  ∃ binary relation <

  [< is a linear order (of the vertices)

  ∧ (∀x ∀y (y is the successor of x for <) → E(x,y))]
- Therefore HAMILTON ∈ ESO
- Notice: HAMILTON and 3-COL are NP-complete, hence are hard problems

A natural question

- What relationships exist between
  - Algorithmic complexity
    - How a problem PB is difficult to solve?

  and

  - Descriptive complexity
    - How the problem PB is difficult to define?

Two kinds of results

- Algorithmic meta theorems
  - Or « generic algorithms of small complexity for solving a class of problems definable in some logic »
- Logical characterizations of complexity classes
  - Or « complexity class = logically definable class »

Algorithmic meta theorems using logic

- They are results of the form

  « Each problem **definable** in a certain logic on a certain class of structures **is solved efficiently** » (Martin Grohe, 2007)

Algorithmic meta theorems using logic
A preliminary example: the TRIANGLE problem in a cubic graph

- *Algorithm*
  - *Input:* a cubic graph G=(V,E)
    (each vertex of G has degree 3, i.e. 3 neighbours)
  - For each vertex *a* of G do
    - For each neighbour *b* of *a* do
      - For each neighbour *c* of *b* except *a* do
        - If *c* is a neighbour of *a* then
          Output « G has a triangle »
  - Output « G has no triangle ».
- *Complexity*
  - The internal test is performed in constant time and is repeated 6|V| times
  - Hence, the whole algorithm runs in linear time

An algorithmic meta theorem using logic

- *Seese's Theorem (1996):* Each graph problem definable in FO is solved in linear time on any class of graphs of bounded degree

- *Application:* the TRIANGLE problem is defined by an FO sentence

  $\exists x\ \exists y\ \exists z\ \ E(x,y) \wedge E(y,z) \wedge E(z,x)$

  and hence, is solved on a cubic graph G in time O(|G|)

Another algorithmic meta theorem using logic

- *Courcelle's Theorem (1990):* Each graph problem definable in MSO is solved in linear time on any class of graphs of bounded tree-width

- *An application:* the KERNEL problem in a directed graph G = (V,E), defined by the MSO (even EMSO) sentence

  $$\exists \mathbf{K}\ [\ \forall x\ \forall y\ ((K(x) \wedge K(y)) \rightarrow \neg\ E(x,y))$$
  $$\wedge\ \forall x\ (\neg K(x) \rightarrow \exists y\ (K(y) \wedge E(x,y)))\ ]$$

  is solved in time O(|G|) if the graph G has tree-width bounded by some fixed k.

  This is not trivial even for k = 1 !

Meta theorems using logic
Why are they interesting ?

- They are general: they allow to establish that some large class of problems are solved efficiently; typically, in linear time
- They allow to establish that some specific problem is solved by an efficient algorithm by defining it in some logic (FO, MSO, etc.)
  - *Examples:* Problems TRIANGLE in a cubic graph or KERNEL in a tree-like graph can both be solved in linear time by Seese's and Courcelle's Theorems, respectively

Do the converse of meta theorems hold?

- In general, No !
  - For example, there are graph problems solved in linear time in cubic graphs that are not definable in FO
  - Both converses of Seese's and Courcelle's Theorems fail !

Are meta theorems optimal?

Courcelle's Theorem is optimal in the following sense

Theorem (Grohe,1999): Let C be a class of graphs (or, more generally, of structures).

The following three assertions are equivalent (under the assumption P≠NP)

- C is of **bounded tree-width**, i.e. there is some k such that the tree-width of each structure G of C is at most k
- Each problem **definable in MSO** is solved in **linear time** on each structure of C
- Each problem **definable in MSO** is solved in **polynomial time** on each structure of C

- *Notice:* here, linear time is equivalent to polynomial time!

Can we exactly characterize a complexity class in logic ?

Some results of the form

Complexity class = Logically definable class

Existential Second-Order Logic is very expressive

- ESO logic allows to define NP-complete problems such as HAMILTON by

  ∃ binary relation <

  [< is a linear order (of the vertices)

  ∧ (∀x ∀y (y is the successor of x for <) → E(x,y))]

- An ESO sentence is a formula of the form

  $\exists R_1 \ldots \exists R_k\, \psi$

  where each R_i is a relation variable of fixed arity and ψ is an FO sentence

ESO logic exactly characterizes NP

Fagin's Theorem (1974): NP = ESO

That means:

A problem is NP

if and only if

it is definable in ESO

Proof of ESO $\leq$ NP

Let PB be a problem, e.g. a graph problem, defined by an ESO sentence $\exists R_1 \ldots \exists R_k\ \psi$

Here is a nondeterministic algorithm that decides whether a graph $G = (V,E)$ belongs to PB, i.e. satisfies $\exists R_1 \ldots \exists R_k\ \psi$

Algorithm

- **Guess some relations** $R_1, \ldots, R_k$ (in time $O(|V|^r)$ where r is the maximal arity of the R_i's)
- **Check** whether the « expanded » structure $(G, R_1, \ldots, R_k)$ **satisfies the FO sentence** ψ: this is performed in deterministic polynomial time

So, this nondeterministic algorithm decides problem PB in polynomial time: hence, problem PB is NP

Sketch of proof of the converse: NP ≤ ESO

- Let PB be an NP problem
- An input G belongs to PB *iff* it **has an accepting computation** C of polynomial time and then **of polynomial size**
- Such an accepting computation can be encoded by a list of relations $R_1 \ldots R_k$
- There is an **FO sentence ψ that exactly defines** the correct accepting computations of input G
- In other words, the list of relations $R_1 \ldots R_k$ **encodes a correct accepting computation** of G *if and only if* $(G, R_1 \ldots R_k)$ **satisfies** ψ
- In other words, G **has an accepting computation** *if and only if* G **satisfies the ESO sentence** $\exists R_1 \ldots \exists R_k\ \psi$
- That means PB **is defined by this ESO sentence**

The central rôle of Second-Order logic

- The SO logic and its restrictions
  - Existential Second-Order Logic (ESO)
  - Monadic Second-Order Logic (MSO)

play a key rôle in describing computations and complexity classes

Here are analogues of Fagin's Theorem for some classical complexity classes included in NP

Theorem (Grädel, 1992)

- **PTIME** = ESO(Horn-clauses) = SO(Horn-clauses)
- **NLOGSPACE** = ESO(2-clauses) = SO(2-clauses)

Idea: A deterministic computation is easily described by Horn clauses

Similarly, an NLOGSPACE computation is described by 2-clauses (clauses of 2 litterals)

A striking refinement of Fagin's Theorem

Fagin's Theorem can be also refined for precise nondeterministic time bounds

- using the RAM model of computation
- and the ESOF logic (Existential Second-Order logic with Functions), i.e. sentences of the form

$$\exists f_1 \ldots \exists f_k \, \psi$$

  - where the f_i's are function variables (of any arity) instead of (or in complement of) relation variables
  - and ψ is an FO sentence

A striking refinement of Fagin's Theorem

Theorem (Grandjean, 1990, and Grandjean, Olive, 2004)

- NLINTIME = ESOF(1 var) = ESOF($\forall^1$) = ESOF($\forall^1$, arity 1)
- And more generally, for each integer d ≥1,

  NTIME(n^d) = ESOF(d var) = ESOF($\forall^d$) = ESOF($\forall^d$, arity d)

Here, ESOF(d var) denotes the class of ESOF sentences with at most d distinct first-order variables. ESO($\forall^d$) and ESOF($\forall^d$, arity d) are defined similarly.

In simplified words,

the degree of the nondeterministic polynomial time

is exactly

the number of first-order variables

Why those results are interesting?

- They yield straightforward (or almost straightforward) completeness results in complexity for natural problems, typically for problems in propositional logic:
  - Fagin's Theorem immediately implies Cook and Levin's Theorem
    - SAT is NP-complete
  - Grädel's Theorem immediately implies that
    - HORN-SAT is PTIME-complete
    - 2-SAT is NLOGSPACE-complete

Hint (for proving Cook's Theorem from Fagin's Theorem):

- Unfold the FO subformula ψ of the ESO formula $\exists R_1 \ldots \exists R_k\ \psi$ as a conjunction over all the possible assignments of its (first-order) variables
- This gives a propositional formula of polynomial size

Why those results are interesting?

Even for nondeterministic linear time (NLINTIME)

The characterization

$$\text{NLINTIME} = \text{ESOF}(\forall^1, \text{arity } 1)$$

also implies by a (sophisticated) unfolding of the unique first-order variable x of any ESOF($\forall^1$, arity 1) sentence

$$\exists f_1 \ldots \exists f_k \ \forall x \ \psi \quad \text{(where } \psi \text{ is quantifier-free)}$$

that the classical problem

RISA (Reduction of Incompletely Specified finite state Automata)

is NLINTIME-complete

Remark: Since the linear time complexity class DTIME(n) (for Turing machines) *is strictly included in* NLINTIME, this implies a complexity lower bound:

RISA $\notin$ DTIME(n), i.e. RISA cannot be solved in linear time on any Turing machine

Why those results are interesting? Robustness and machine independence

- Those characterizations show how the complexity classes involved, NP, PTIME, NLOGSPACE, NLINTIME are robust,
  - not only from a computational point of view (they have many equivalent definitions)
  - but also from a logical point of view
- They are in fact machine independent:
  - NP is the set of problems definable in ESO
  - NLINTIME is the set of problems definable in ESO (with function variables) using only 1 first-order variable
- As their machine counterparts, the logical classes involved are robust, i.e. their ability to define problems does not change for a number of extensions and restrictions (normalizations), typically

$$\text{ESOF}(1 \text{ var}) = \text{ESOF}(\forall^1) = \text{ESOF}(\forall^1, \text{arity } 1)$$

Conclusion

We have presentented two kinds of results that involve logic in algorithmics and complexity theory:

- Algorithmic meta theorems
  - Or « generic algorithms of small complexity for solving a class of problems definable in some logic »
- Logical characterizations of complexity classes
  - Or « complexity class = logically definable class »

Conclusion: the state of art

Our initial question: What relationships exist between

- Algorithmic complexity, and
- Descriptive complexity

is still widely open !

Typically, we have

- ESO = NP and also SO = PH (the Polynomial Hierarchy beyond NP)
- but know no similar equality for the class FO (the class of problems defined in First-Order logic) or MSO