Logics for Polynomial Time

Tutorial Part 1

Anuj Dawar

University of Cambridge

JAF, Fontainebleau, 17 June 2009

Descriptive Complexity

Descriptive Complexity provides an alternative perspective on Computational Complexity.

Computational Complexity

- Measure use of resources (space, time, *etc.*) on a machine model of computation;
- Complexity of a language—i.e. a set of strings.

Descriptive Complexity

- Complexity of a class of structures—e.g. a collection of graphs.
- Measure the complexity of describing the collection in a formal logic, using resources such as variables, quantifiers, higher-order operators, *etc.*

There is a fascinating interplay between the views.

First-Order Logic

For a first-order sentence φ, we ask what is the *computational complexity* of the problem:

Given: a structure $\mathbb{A}$

Decide: if $\mathbb{A} \models \varphi$

In other words, how complex can the collection of finite models of φ be?

In order to talk of the complexity of a class of finite structures, we need to fix some way of representing finite structures as strings.

Encoding Structures

We use an alphabet $\Sigma = \{0, 1, \#, -\}$.

For a structure $\mathbb{A} = (A, R_1, \ldots, R_m, f_1, \ldots, f_l)$, fix a linear order $<$ on $A = \{a_1, \ldots, a_n\}$.

R_i (of arity k) is encoded by a string $[R_i]_<$ of 0s and 1s of length n^k.

f_i is encoded by a string $[f_i]_<$ of 0s, 1s and $-$s of length $n^k \log n$.

$$[\mathbb{A}]_< = \underbrace{1 \cdots 1}_{n} \#[R_1]_< \# \cdots \#[R_m]_< \#[f_1]_< \# \cdots \#[f_l]_<$$

The exact string obtained depends on the choice of order.

Naïve Algorithm

The straightforward algorithm proceeds recursively on the structure of φ:

- Atomic formulas by direct lookup.
- Boolean connectives are easy.
- If $\varphi \equiv \exists x\, \psi$ then for each $a \in \mathbb{A}$ check whether

$$(\mathbb{A}, c \mapsto a) \models \psi[c/x],$$

  where c is a new constant symbol.

This runs n time $O(ln^m)$ and $O(m \log n)$ space, where m is the nesting depth of quantifiers in φ.

$$\mathrm{Mod}(\varphi) = \{\mathbb{A} \mid \mathbb{A} \models \varphi\}$$

is in *logarithmic space* and *polynomial time*.

Second-Order Logic

There are computationally easy properties that are not definable in first-order logic.

- There is no sentence φ of first-order logic such that $\mathbb{A} \models \varphi$ if, and only if, $|A|$ is even.
- There is no formula $\varphi(E, x, y)$ that defines the transitive closure of a binary relation E.

Consider second-order logic, extending first-order logic with *relational quantifiers* — $\exists X \varphi$

Examples

Evennness

This formula is true in a structure if, and only if, the size of the domain is even.

$$\exists B \exists S \quad \forall x \exists y B(x,y) \wedge \forall x \forall y \forall z B(x,y) \wedge B(x,z) \rightarrow y = z$$
$$\forall x \forall y \forall z B(x,z) \wedge B(y,z) \rightarrow x = y$$
$$\forall x \forall y S(x) \wedge B(x,y) \rightarrow \neg S(y)$$
$$\forall x \forall y \neg S(x) \wedge B(x,y) \rightarrow S(y)$$

Examples

Transitive Closure

This formula is true of a pair of elements a, b in a structure if, and only if, there is an E-path from a to b.

$$\exists P \quad \forall x \forall y \, P(x,y) \rightarrow E(x,y)$$
$$\exists x P(a,x) \wedge \exists x P(x,b) \wedge \neg \exists x P(x,a) \wedge \neg \exists x P(b,x)$$
$$\forall x \forall y (P(x,y) \rightarrow \forall z (P(x,z) \rightarrow y = z))$$
$$\forall x \forall y (P(x,y) \rightarrow \forall z (P(z,x) \rightarrow y = z))$$
$$\forall x ((x \neq a \wedge \exists y P(x,y)) \rightarrow \exists z P(z,x))$$
$$\forall x ((x \neq b \wedge \exists y P(y,x)) \rightarrow \exists z P(x,z))$$

Examples

3-Colourability

The following formula is true in a graph (V, E) if, and only if, it is 3-colourable.

$$\begin{aligned}
\exists R \exists B \exists G \quad & \forall x (Rx \vee Bx \vee Gx) \wedge \\
& \forall x (\quad \neg(Rx \wedge Bx) \wedge \neg(Bx \wedge Gx) \wedge \neg(Rx \wedge Gx)) \wedge \\
& \forall x \forall y (Exy \rightarrow (\quad \neg(Rx \wedge Ry) \wedge \\
& \qquad\qquad\qquad\quad \neg(Bx \wedge By) \wedge \\
& \qquad\qquad\qquad\quad \neg(Gx \wedge Gy)))
\end{aligned}$$

Fagin's Theorem

Theorem (Fagin)

A class $\mathcal{C}$ of finite structures is definable by a sentence of *existential second-order logic* if, and only if, it is decidable by a *nondeterminisitic machine* running in polynomial time.

$$\text{ESO} = \text{NP}$$

One direction is easy: Given $\mathbb{A}$ and $\exists P_1 \ldots \exists P_m \varphi$.

a nondeterministic machine can guess an interpretation for $P_1, \ldots, P_m$ and then verify φ.

Fagin's Theorem

Given a machine M and an integer k, there is an ESO sentence φ such that $\mathbb{A} \models \varphi$ if, and only if, M accepts $[\mathbb{A}]_<$, for some order $<$ in n^k steps.

$\exists <$ $\exists$State $\exists$Head $\exists$Tape

$<$ is a linear order $\wedge$

$\left.\begin{array}{l} \text{State}(t+1, s_1) \rightarrow \text{State}(t, s) \vee \ldots \\ \wedge \text{State}(t+1, s_2) \rightarrow \ldots \\ \wedge \text{Tape}(t+1, p) \leftrightarrow \text{Head}(t, p) \ldots \\ \wedge \text{Head}(t+1, h+1) \leftrightarrow \ldots \\ \wedge \text{Head}(t+1, h-1) \leftrightarrow \ldots \end{array}\right\}$ encoding transitions of M

$\wedge$at time 0 the tape contains a description of $\mathbb{A}$

$\wedge \text{State}(\text{max}, s)$ for some accepting s

Fagin's Theorem

State, Tape and Head are $2k$-ary relations, that use the lexicographic order on k-tuples.

To state that Tape encodes the input structure:

$$\begin{aligned} \forall \mathbf{x} \quad & \mathbf{x} < n \rightarrow \mathsf{Tape}(0, \mathbf{x}) \\ & \mathbf{x} < n^a \rightarrow (\mathsf{Tape}(0, \mathbf{x} + n) \leftrightarrow R_1(\mathbf{x}|_a)) \\ \ldots & \end{aligned}$$

where,

$$\mathbf{x} < n^a \quad : \quad \bigwedge_{i \leq (k-a)} x_i = 0$$

Is there a logic for P?

The major open question in *Descriptive Complexity* (first asked by Chandra and Harel in 1982) is whether there is a logic $\mathcal{L}$ such that

> for any class of finite structures $\mathcal{C}$, $\mathcal{C}$ is definable by a sentence of $\mathcal{L}$ if, and only if, $\mathcal{C}$ is decidable by a deterministic machine running in polynomial time.

Formally, we require $\mathcal{L}$ to be a *recursively enumerable* set of sentences, with a computable map taking each sentence to a Turing machine M and a polynomial time bound p such that (M, p) accepts a *class of structures*.

(Gurevich 1988)

Enumerating Queries

For a given structure $\mathbb{A}$ with n elements, there may be as many as $n!$ distinct strings $[\mathbb{A}]_<$ encoding it.

Given $(M_0, p_0), \ldots, (M_i, p_i), \ldots$—an enumeration of polynomially-clocked Turing machines.

Can we enumerate a subsequence of those that compute graph properties, i.e. are *encoding invariant*, while including all such properties?

Recursive Indexability

We say that P is *recursively indexable*, if there is a recursive set $\mathcal{I}$ and a Turing machine M such that:

- on input $i \in \mathcal{I}$, M produces the code for a machine $M(i)$ and a polynomial p_i
- $M(i)$, accepts a class of structures in P.
- $M(i)$ runs in time bounded by p_i
- for each class of structures $C \in$ P, there is an i such that $M(i)$ accepts C.

Canonical Labelling

We say that a machine M *canonically labels* graphs, if

- on any input $[G]_{\leq}$, the output of M is $[G]_{\leq'}$ for some ordering $\leq'$; and
- if $[G]_{\leq_1}$ and $[G]_{\leq_2}$ are two encodings of the same graph, then $M([G]_{\leq_1}) = M([G]_{\leq_2})$.

It is an open question whether such a polynomial-time exists.

If so, then P is recursively indexable, by enumerating machines $M \to M_i$.

If not, P $\neq$ NP.

Interpretations

Given two relational signatures σ and τ, where $\tau = \langle R_1, \ldots, R_r \rangle$, and arity of R_i is n_i

A *first-order interpretation of τ in σ* is a sequence:

$$\langle \pi_U, \pi_1, \ldots, \pi_r \rangle$$

of first-order σ-formulas, such that, for some k,:

- the free variables of π_U are among $x_1, \ldots, x_k$,
- and the free variables of π_i (for each i) are among $x_1, \ldots, x_{k \cdot n_i}$.

k is the width of the interpretation.

Interpretations II

An interpretation of τ in σ maps σ-structures to τ-structures.

If $\mathbb{A}$ is a σ-structure with universe A, then

$\pi(\mathbb{A})$ is a structure $(B, R_1, \ldots, R_r)$ with

- $B \subseteq A^k$ is the relation defined by π_U.
- for each i, R_i is the relation on B defined by π_i.

Reductions

Given:

- C_1 – a class of structures over σ; and
- C_2 – a class of structures over τ

π is a *first-order reduction* of C_1 to C_2 if, and only if,

$$\mathbb{A} \in C_1 \Leftrightarrow \pi(\mathbb{A}) \in C_2.$$

If such a π exists, we say that C_1 is first-order reducible to C_2.

Bi-interpretation with Graphs

For any σ and any class C of σ-structures, there is a class D of *graphs* (i.e. structures over a signature containing just one binary relation) such that:

- C is first-order reducible to D; and
- D is first-order reducible to C.

This follows from a standard model-theoretic bi-interpretation.

NP-complete Problems

First-order reductions are, in general, much weaker than *polynomial-time reductions* and (in the absence of order and airthmetic on the structures) even weaker than AC_0-reductions.

Nonetheless, there are NP-complete problems under such reductions.

Every problem in NP is first-order reducible to *SAT*

(Lovàsz and Gàcs 1977)

Hamiltonicity and *Clique* are NP-complete via first-order reductions

(Dahlhaus 1984)

But, *3-colourability* is not NP-complete via first-order reductions.

(D.-Grädel 1995)

and the question is open for *3SAT*.

P-complete Problems

If there is any problem that is complete for P with respect to first-order reductions, then there is a logic for P.

If Q is such a problem, we form, for each k, a quantifier Q^k.

The sentence

$$Q^k(\pi_U, \pi_1, \ldots, \pi_s)$$

for a k-ary interpretation $\pi = (\pi_U, \pi_1, \ldots, \pi_s)$ is defined to be true on a structure $\mathbb{A}$ just in case

$$\pi(\mathbb{A}) \in Q.$$

The collection of such sentences is then a logic for P.

Conversely,

Theorem

If the polynomial time properties of graphs are recursively indexable, there is a problem complete for P under first-order reductions.

(D. 1995)

Proof Idea:

Given a recursive indexing $((M_i, p_i)|i \in \omega)$ of P

Encode the following problem into a class of finite structures:

$$\{(i, x)|M_i \text{ accepts } x \text{ in time bounded by } p_i(|x|)\}$$

To ensure that this problem is still in P, we need to pad the input to have length $p_i(|x|)$.

Constructing the Complete Problem

Suppose M is a machine which on input $i \in \omega$ gives a pair (M_i, p_i) as in the definition of recursive indexing. Let g a recursive bound on the running time of M.

Q is a class of structures over the signature $(V, E, \preceq, I)$.

$\mathbb{A} = (A, V, E, \preceq, I)$ is in Q if, and only if,

1. $\preceq$ is a linear pre-order on A;
2. if $a, b \in I$, $a \preceq b$ and $b \preceq a$, i.e. I picks out one equivalence class from the pre-order (say the i^{th});
3. $|A| \geq p_i(|V|)$;
4. the graph (V, E) is accepted by M_i; and
5. $g(i) \leq |A|$.

Summary

The following are equivalent:

- P is recursively indexable.
- There is a logic capturing P of the form FO$(\mathbf{Q})$, where $\mathbf{Q}$ is the collection of vectorisations of a single quantifier.
- There is a complete problem in P under first order reductions.