Complexity Theory
Lecture 12

Anuj Dawar

University of Cambridge Computer Laboratory

Easter Term 2015

http://www.cl.cam.ac.uk/teaching/1415/Complexity/

Time Hierarchy Theorem

For any constructible function f, with $f(n) \geq n$, define the f-bounded *halting language* to be:

$$H_f = \{[M], x \mid M \text{ accepts } x \text{ in } f(|x|) \text{ steps}\}$$

where $[M]$ is a description of M in some fixed encoding scheme.

Then, we can show

$H_f \in \mathsf{TIME}(f(n)^3)$ and $H_f \notin \mathsf{TIME}(f(\lfloor n/2 \rfloor))$

Time Hierarchy Theorem

For any constructible function $f(n) \geq n$, $\mathsf{TIME}(f(n))$ is properly contained in $\mathsf{TIME}(f(2n+1)^3)$.

Strong Hierarchy Theorems

For any constructible function $f(n) \geq n$, $\mathsf{TIME}(f(n))$ is properly contained in $\mathsf{TIME}(f(n)(\log f(n)))$.

Space Hierarchy Theorem

For any pair of constructible functions f and g, with $f = O(g)$ and $g \neq O(f)$, there is a language in $\mathsf{SPACE}(g(n))$ that is not in $\mathsf{SPACE}(f(n))$.

Similar results can be established for nondeterministic time and space classes.

Consequences

- For each k, $\mathsf{TIME}(n^k) \neq \mathsf{P}$.
- $\mathsf{P} \neq \mathsf{EXP}$.
- $\mathsf{L} \neq \mathsf{PSPACE}$.
- Any language that is EXP-complete is not in P.
- There are no problems in P that are complete under linear time reductions.

Descriptive Complexity

Descriptive Complexity is an attempt to study the complexity of problems and classify them, not on the basis of how difficult it is to *compute* solutions, but on the basis of how difficult it is to *describe* the problem.

This gives an alternative way to study complexity, independent of particular machine models.

Based on *definability in logic*.

Graph Properties

As an example, consider the following three decision problems on *graphs*.

1. Given a graph $G = (V, E)$ does it contain a *triangle*?
2. Given a directed graph $G = (V, E)$ and two of its vertices $a, b \in V$, does G contain a *path* from a to b?
3. Given a graph $G = (V, E)$ is it *3-colourable*? That is,

   is there a function $\chi : V \to \{1, 2, 3\}$ so that whenever $(u, v) \in E$, $\chi(u) \neq \chi(v)$.

Graph Properties

1. Checking if G contains a triangle can be solved in *polynomial time* and *logarithmic space*.

2. Checking if G contains a path from a to b can be done in *polynomial time*.

   Can it be done in *logarithmic space*?

   Unlikely. It is NL-complete.

3. Checking if G is 3-colourable can be done in *exponential time* and *polynomial space*.

   Can it be done in *polynomial time*?

   Unlikely. It is NP-complete.

Logical Definability

In what kind of formal language can these decision problems be *specified* or *defined*?

The graph $G = (V, E)$ contains a triangle.

$$\exists x, y, z \in V (x \neq y \land y \neq z \land x \neq z \land E(x, y) \land E(x, z) \land E(y, z))$$

The other two properties are *provably* not definable with only first-order quantification over vertices.

First-Order Logic

Consider *first-order predicate logic*.

A collection of variables $x, y, \ldots$, and formulas:

$$E(x, y) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \exists x\phi \mid \forall x\phi$$

Any property of graphs that is expressible in *first-order logic* is in L.

The problem of deciding whether $G \models \phi$ for a first-order ϕ is in time $O(ln^m)$ and $O(m \log n)$ space.

where, l is the *length* of ϕ and n the *order* of G and m is the nesting depth of quantifiers in ϕ.

Complexity of First-Order Logic

The straightforward algorithm proceeds recursively on the structure of ϕ:

- Atomic formulas by direct lookup.
- Boolean connectives are easy.
- If $\phi \equiv \exists x\, \psi$ then for each v in G check whether

$$(G, x \mapsto v) \models \psi.$$

Second-Order Quantifiers

3-Colourability and *Reachability* can be defined with quantification over *sets of vertices*.

$$
\begin{aligned}
\exists R \subseteq V\, \exists B \subseteq V\, \exists G \subseteq V \\
& \forall x (Rx \lor Bx \lor Gx) \land \\
& \forall x (\neg (Rx \land Bx) \land \neg (Bx \land Gx) \land \neg (Rx \land Gx)) \land \\
& \forall x \forall y (Exy \to (\neg (Rx \land Ry) \land \\
& \qquad\qquad\quad \neg (Bx \land By) \land \\
& \qquad\qquad\quad \neg (Gx \land Gy)))
\end{aligned}
$$

$$\forall S \subseteq V (a \in S \land \forall x \forall y ((x \in S \land E(x, y)) \to y \in S) \to b \in S)$$

Existential Second-Order Logic

Second-order logic is obtained by adding to the defining rules of first-order logic two further clauses:

atomic formulae – $X(t_1, \ldots, t_a)$, where X is a *second-order variable*

second-order quantifiers – $\exists X \phi$, $\forall X \phi$

Existential Second-Order Logic (ESO) consists of formulas of the form

$$\exists X_1 \cdots \exists X_k \phi$$

where ϕ is *first-order*

Fagin's Theorem

Theorem (Fagin)
A class of graphs is definable by a formula of *existential second-order logic* if, and only if, it is decidable by a *nondeterminisitic machine* running in polynomial time.

$$\mathsf{ESO} = \mathsf{NP}$$

One direction is easy: Given G and $\exists X_1 \ldots \exists X_k \phi$.

> a nondeterministic machine can guess an interpretation for $X_1, \ldots, X_k$ and then verify ϕ.

The other direction requires a proof similar to Cook's theorem.

A Logic for P?

Is there a logic, intermediate between first and second-order logic that expresses exactly graph properties in P?

This is an open question, still the subject of active research.

The End

Please provide *feedback*, using the link sent to you by e-mail.