Computability and Definability

by

J.A. Makowsky

Department of Computer Science
Technion - Israel Institute of Technology
Haifa, Israel

janos@@cs.technion.ac.il
http://cs.technion.ac.il/~janos

Course given the Technion
Haifa, Israel

Spring Semester, 2001

PRELUDE

Outlook
References

Computing devices

Machine:

Input → [Machine M] → Output

Finite Automaton
Turing Machine (with resource bounds)
Register Machine (with resource bounds)
Boolean and Algebraic Circuits

Transducer:

In-structure → [Machine T] → Out-structure

Acceptor:

Input → [Machine A] → $\{0, 1\}$

Counter:

Input → [Machine C] → $\mathbb{N}$

Combinatorial algorithms, I

Acceptors:
Deciding properties of a graph
Connected, cycle-free, hamiltonian, 3-colorable

Graph → [Machine A] → $\{0, 1\}$

Transducers:
Finding configurations in a graph
Connected component, (hamiltonian) cycle, 3-coloring

Graph → [Machine A] → Graph

Counters:
Counting configurations in a graph
Connected components, (hamiltonian) cycles,

Graph → [Machine A] → $\mathbb{N}$

Input for Machines

For Finite Automata and Turing Machines the input is **coded** in (finite) words over some alphabet Σ.

For Boolean circuits the input is **coded** as Boolean vectors in $\bigcup_n \{0,1\}^n$.

For Algebraic circuits over a field or ring $\mathcal{R}$, the input is **coded** as vector over $\bigcup_n \mathcal{R}^n$.

For Register Machines we may have specialized registers for specific data types, including words, natural numbers, real numbers, finite relations, etc.....

Combinatorial algorithms, II

We shall also look at machines

Evaluating cost functions in a graph
Size of connected components,
size or cost of hamiltonian cycles, cost of 3-coloring

Optimizing with respect to cost functions
size of largest connected component,
cost of cheapest hamiltonian cycle, etc

Here the input is a weighted graph with weights in some ring $\mathcal{R}$ and the output may be a value in $\mathcal{R}$, a natural number or a boolean value.

The appropriate computation devices are
in the non-uniform case
Valiant's model of algebraic circuits
or, in the uniform case,
the model **BSS** (due to Blum, Shub and Smale).

Complexity theory

Each machine type uses resources:

- Computing time
- Number of gates
- Space on tape
- Number of auxiliary registers
- Content size of registers

Computability:
there is a machine which does the job

Complexity classes:
computability within certain
resource restrictions

What do we compute ?

The machine M computes (accepts) "what it computes (accepts)".

The machine M accepts exactly the same inputs as the machine M'.

The machine M accepts exactly the inputs which have "property P".

What are properties ?

Concrete graphs (in $\mathbb{R}^3$)

A **concrete** graph G is given by a finite set of *points* V in $\mathbb{R}^3$ and a finite set E of *ropes* linking two points v_1, v_2. The ropes are continuous curves which do not intersect.

Plane graphs

A **plane** graph G is given by a finite set of *points* V in $\mathbb{R}^2$ and a finite set E of *arcs* linking two points v_1, v_2. The arcs are continuous curves which do not intersect.

Abstract graphs

An **abstract** graph G is given by a finite set of *vertices* V and a finite set E of *edges* linking two vertices v_1, v_2.
Here $E \subseteq V^{(2)}$ where $V^{(2)}$ denotes the set of unordered pairs of elements of V.

$V = \{1, \ldots, 6\}$

$$E = \begin{cases} \{(1,2),(2,3),(3,1)\} \cup \\ \{(4,5),(5,6),(6,4)\} \cup \\ \{(1,6),(6,3),(3,5),(5,2),(2,4),(4,1)\} \end{cases}$$

Graph properties

This graph is (*give definition*):

- 4-regular
- Chordal (triangulated)
- 3-colorable
- Eulerian
- Hamiltonian
- Planar

k-regular graphs

We say that v_1 **and** v_2 **are neighbors** if they are connected by a rope (arc, edge). We can write for this $E(v_1, v_2)$.

A graph is k-**regular** if every vertex has exactly k distinct neighbors. For $k = 4$ we can write

$$Diff(x_0, x_1, \ldots, x_n) = \left(\bigwedge_{i<j\leq n} x_i \neq x_j \right) =$$
$$(v_0 \neq v_1 \wedge v_0 \neq v_2 \wedge \ldots v_{n-2} \neq v_{n-1} \wedge v_{n-1} \neq v_n)$$

$$\forall v_0 \, (\exists v_1 \exists v_2 \exists v_3 \exists v_4 \, (Diff(x_0, x_1, \ldots, x_4) \rightarrow$$
$$\wedge \, (E(v_0, v_1) \wedge E(v_0, v_2) \wedge E(v_0, v_3) \wedge E(v_0, v_4)))$$
$$\wedge \, (\forall v_1 \forall v_2 \forall v_3 \forall v_4 \forall v_5 \, (Diff(x_0, x_1, \ldots, x_5) \rightarrow$$
$$\neg E(v_0, v_1) \vee \neg E(v_0, v_2) \vee \neg E(v_0, v_3) \vee \neg E(v_0, v_4) \vee \neg E(v_0, v_5)))$$

which is a well formed formula in **First Order Logic** in the vocabulary of abstract graphs with one binary relation symbol E.

Chordal graphs

A graph $G_1 = (V_1, E_1)$ is an **induced subgraph** of $G = (V, E)$ if $V_1 \subseteq V$ and $E_1 = E \cap V_1^2$.

A graph is a **simple cycle of length** k of it is of the form:

A graph G is **chordal** if there is no induced subgraph of G isomorphic to a simple cycle of length ≥ 4.

Can we say this in First Order Logic ?

k-colorable graphs

A subset V_1 of a graph $G = (V, E)$ is **independent** if it induces a graph of points without neighbors (nor loops).

A graph is k-**colorable** if its vertices can be partitioned into k independent sets.

$$Part(X_1, X_2, X_3) = ((X_1 \cup X_2 \cup X_3 = V) \wedge$$
$$((X_1 \cap X_2) = (X_2 \cap X_3) = (X_3 \cap X_1) = \emptyset))$$

$$Ind(X) := (\forall v_1 \in X)(\forall v_2 \in X) \neg E(v_1, v_2)$$

With this 3-colorable can be expressed as

$$\exists X_1 \exists X_2 \exists X_3 \, (Part(X_1, X_2, X_3) \wedge Ind(X_1) \wedge Ind(X_2) \wedge Ind(X_3))$$

We have expressed 3-colorability by a formula in **(Monadic) Second Order Logic.**

Can we express this in First Order Logic ?

Eulerian graphs

A graph $G = (V, E)$ is **Eulerian** if we can follow each rope exactly once, pass through all the ropes, and return to the point of departure.

Equivalently:
Can we order all the edges of E

$$e_1, e_2, e_3, \ldots e_m$$

and choose beginning and end of th edge $e_i = (u_i, v_i)$ such that for all i, $v_i = u_{i+1}$ and $v_m = u_1$.

Theorem (Euler):
A connected graph G is Eulerian iff each vertex has even degree.

How can we express this more conveniently ?

Hamiltonian graphs

A graph $G_1 = (V_1, E_1)$ is a **subgraph of** $G = (V, E)$ if $V_1 \subseteq V$ and $E_1 \subseteq E \cap V_1^2$.
G_1 is a **spanning subgraph** if $V_1 = V$.
Recall G_1 is an induced subgraph if $E_1 = E \cap V_1^2$.

A graph with n vertices is Hamiltonian if it contains a spanning subgraph which is a cycle of size n.

We define formulas:

$Conn(V_1, E_1)$: (V_1, E_1) is connected.
$Cycle(V_1, E_1)$: is a cycle.

Proposition: A graph is a cycle iff it is connected and each vertex has exactly two neighbors.

$$HAM := \exists V_1 \exists E_1 \, (Cycle(V_1, E_1) \wedge E_1 \subseteq E \wedge V_1 = V)$$

Planar graphs

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are **isomorphic** if there is a $1-1$ function

$$f : V_1 \to V_2$$

such that for each pair $(u_1, v_1) \in V_1^2$ we have that

$$(u_1, v_1) \in E_1 \text{ iff } (f(u_1), f(v_1)) \in E_2$$

A graph is **planar** iff it is isomorphic to a plane graph.

This definition involves the geometry of the Euclidean plane.

How can we express planarity **without** geometry ?

Logics

- Propositional Logic
  Atomic statements, Closure under
  Boolean operations $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$
- Predicate Logic or First Order Logic, **FOL**
  Relational language, quantification over
  individual variables only,
- Second Order Logic, **SOL**
  Relational language, quantification over individual and **typed relation variables**,
- Monadic Second Order Logic, **MSOL**
  Relational language, quantification over individual and **unary relation variables**,

Theme 1:

Can you say it
in
First Order,
Second Order or
Monadic Second Order
Logic ?

Complexity classes

$\mathbf{L}, \mathbf{NL}, \mathbf{P}, \mathbf{NP}, \mathbf{PH}, \sharp\mathbf{P}, \mathbf{PSpace}$

Büchi-Trachtenbrot (1961):
Regular languages = MSOL definable classes of words.

Fagin-Christen (1974);
Class of graphs is in $\mathbf{NP}$ iff it is $\exists SOL$ definable.

*****

Can we characterize other complexity classes, such as $\mathbf{L}, \mathbf{NL}, \mathbf{P}, \mathbf{PSpace}$?

Theme 2:

Definability vs. Computability

Can we use logical methods to learn more about $\mathbf{P} =_? \mathbf{NP}$?

Complexity of SOL-properties

Fagin, Christen:
The **NP**-properties of classes of τ-structures are exactly the $\exists SOL$-definable properties.

Meyer,Stockmeyer:
The **PH**-properties (in the *polynomial hierarchy*)
of classes of τ-structures are exactly the SOL-definable properties.

Makowsky, Pnueli:
For every level Σ_n^P of $\mathbf{PH}$ there are $MSOL$-definable classes which are complete for it.

HEX and Geography, I

HEX: Undirected graph G and two vertices s, t.
Players color alternately vertices in $V - \{s, t\}$ white and black respectively.
Player I tries to construct a white path from s to t and Player II tries to prevent this.

HEX: The class of graphs which allow a Winning Strategy for I.

Geography: Directed graph G. Players choose alternately new edges starting at the end point of the last chosen edge. The first who cannot find such an edge has lost.

GEOGRAPHY: The class of graphs which allow a Winning Strategy for I.

HEX and Geography, II

Even, Tarjan: HEX is **PSPACE**-complete.

Schaefer: GEOGRAPHY is **PSPACE**-complete.

Short versions: Fix $k \in \mathbb{N}$.
SHORT-HEX, SHORT-GEOGRAPHY asks whether Player I can win in k moves.

SHORT-HEX and SHORT-GEOGRAPHY are FOL-definable for fixed k (and therefore solvable in **P**).

Separating Complexity Classes, I

- HEX is SOL-definable iff $\mathbf{PSpace} = \mathbf{PH}$.
- Every sentence $\phi \in SOL(\tau)$ is equivalent (over finite structures) to an existential sentence $\psi \in SOL(\tau)$ iff $\mathbf{NP} = \mathbf{CoNP}$.
  Note we allow arbitrary arities of the quantified relation variables.
  Over infinite structures this is known to be false (Rabin)
- If there is a $\phi \in SOL(\tau)$ which is **not** equivalent to an existential sentence, then $\mathbf{P} \neq \mathbf{NP}$.
  And there should be such a sentence

Separating Complexity Classes, II

Problem: Is there a logic for $\mathbf{P}$?

- Have to make precise what is a logic ?
- Over **ordered finite** structures, YES.
  Immerman and Vardi: Fixed Point Logic
  Graedel: Horn Fragment of SOL
- For structures without order this is one of the main open problems in Finite Model Theory

Logical Methods: Model Theory

- Definability :
  Can we say it?
- Non-definability:
  Can we say that we cannot say it?
- Inductivity:
  Can we pass from simple to complex structures?

References: Textbooks

[Pa94] C. Papadimitriou,
Computational Complexity,
Addison-Wesley 1994

[EF95] H.-D. Ebbinghaus and Jörg Flum,
Finite Model Theory,
Perspectives in Mathematical Logic,
Springer 1995, ISBN 3-540-60149-X

[Bo99] B. Bollobas,
Modern Graph Theory,
Graduate Texts in Mathematics, Springer 1999.

[Di97] R. Diestel,
Graph Theory,
Graduate Texts in Mathematics, Springer 1997.