A potpourri from Descriptive Complexity

Antonis Achilleos

June 26, 2008

Outline

Once again: What is that?

First-order logic

- Games, limits and complexity of
- Add-ons: Built-in relations

Fagin's Theorem, $\mathcal{NP}$ and $\mathcal{PH}$

$\mathcal{P}$, $\mathcal{L}$ and stuff

- Fixed Points
- $\mathcal{P}$
- $\mathcal{L}, \mathcal{NL}$ and transitive closures

Regular things

- $MSO, \exists MSO$ and Automata

Descriptive Complexity

- In Computational Complexity, we deal with the classification of problems (properties of strings, graphs, etc.) in complexity classes and many times we try to find the relationship between these classes.
- In logic (sometimes) we try to express properties of structures in a given language and find the limits of the language.
- For example: Tarski's inexpressibility of Truth.
- Model Theory
- *Finite* Model Theory deals with *finite* structures (can mostly be thought of as graphs). These are more appropriate if we want to imagine them as inputs to a computational problem.

Descriptive Complexity

- When dealing with finite structures, many things are different from the infinite case.
- First-order logic is no longer uncomputable. On the contrary many relatively easy problems cannot be expressed by it.
- More expressive languages are needed.
- Descriptive Complexity studies the relationship between these logics and complexity classes.
- If a property of a finite structure (decision problem) can be expressed by a formula of logic L, what is its computational complexity?
- What logic is needed to express all properties in a specific complexity class?

Structures, vocabularies, graphs and strings

- Vocabularies are...
- Structures are...
- Graphs are...
- And I'm sure you thought you knew what strings are...

Outline

Ehrenfeucht - Fraisse and Pe(e)bble games

- Players: Player I - Player II, or Spoiler - Duplicator
- They play on two structures, say $\mathcal{A}$ and $\mathcal{B}$.
- The game is $\mathcal{G}^k$: k rounds
- Spoiler tries to show that the two structures are not identical and Duplicator tries to respond to Spoiler's challenges.
- Spoiler moves first each round, say round i.
- S picks an element from $\mathcal{A}$ or $\mathcal{B}$ and calls it a_i, or b_i accordingly.
- D responds with an element from the other structure, s.t. both a_i, and b_i are defined. and they keep placing pebbles...
- They play for n rounds.
- If the induced substructures on the elements chosen are isomorphic, with the isomorphism mappint a_i to b_i, D wins. Otherwise, S does.

Ehrenfeucht - Fraisse and Pe(e)bble games

- Players: Player I - Player II, or Spoiler - Duplicator
- They play on two structures, say $\mathcal{A}$ and $\mathcal{B}$.
- The game is $\mathcal{G}^k$: k rounds
- Spoiler tries to show that the two structures are not identical and Duplicator tries to respond to Spoiler's challenges.
- Spoiler moves first each round, say round i.
- S picks an element from $\mathcal{A}$ or $\mathcal{B}$ and calls it a_i, or b_i accordingly.
- D responds with an element from the other structure, s.t. both a_i, and b_i are defined. and they keep placing pebbles...
- They play for n rounds.
- If the induced substructures on the elements chosen are isomorphic, with the isomorphism mappint a_i to b_i, D wins. Otherwise, S does.

Ehrenfeucht - Fraisse and Pe(e)bble games

- Players: Player I - Player II, or Spoiler - Duplicator
- They play on two structures, say $\mathcal{A}$ and $\mathcal{B}$.
- The game is $\mathcal{G}^k$: k rounds
- Spoiler tries to show that the two structures are not identical and Duplicator tries to respond to Spoiler's challenges.
- Spoiler moves first each round, say round i.
- S picks an element from $\mathcal{A}$ or $\mathcal{B}$ and calls it a_i, or b_i accordingly.
- D responds with an element from the other structure, s.t. both a_i, and b_i are defined. and they keep placing pebbles...
- They play for n rounds.
- If the induced substructures on the elements chosen are isomorphic, with the isomorphism mappint a_i to b_i, D wins. Otherwise, S does.

Ehrenfeucht - Fraisse and Pe(e)bble games

- Players: Player I - Player II, or Spoiler - Duplicator
- They play on two structures, say $\mathcal{A}$ and $\mathcal{B}$.
- The game is $\mathcal{G}^k$: k rounds
- Spoiler tries to show that the two structures are not identical and Duplicator tries to respond to Spoiler's challenges.
- Spoiler moves first each round, say round i.
- S picks an element from $\mathcal{A}$ or $\mathcal{B}$ and calls it a_i, or b_i accordingly.
- D responds with an element from the other structure, s.t. both a_i, and b_i are defined. and they keep placing pebbles...
- They play for n rounds.
- If the induced substructures on the elements chosen are isomorphic, with the isomorphism mappint a_i to b_i, D wins. Otherwise, S does.

Ehrenfeucht - Fraisse and Pe(e)bble games

- Players: Player I - Player II, or Spoiler - Duplicator
- They play on two structures, say $\mathcal{A}$ and $\mathcal{B}$.
- The game is $\mathcal{G}^k$: k rounds
- Spoiler tries to show that the two structures are not identical and Duplicator tries to respond to Spoiler's challenges.
- Spoiler moves first each round, say round i.
- S picks an element from $\mathcal{A}$ or $\mathcal{B}$ and calls it a_i, or b_i accordingly.
- D responds with an element from the other structure, s.t. both a_i, and b_i are defined. and they keep placing pebbles...
- They play for n rounds.
- If the induced substructures on the elements chosen are isomorphic, with the isomorphism mappint a_i to b_i, D wins. Otherwise, S does.

Ehrenfeucht - Fraisse and Pe(e)bble games

- Ehrenfeucht - Fraisse games are a way to show inexpressibility results about first-order logic.
- S wins G_k iff a first order sentence with at most k quantifier alterations can distinguish between the structures. $(\exists\vec{x_1}\forall\vec{x_2}\cdots\forall\vec{x_k}\phi(\vec{x}_1,\ldots,\vec{x}_k))$
- Pebble games are similar, but instead of choosing $a_i's$ and $b_i's$, they place pairs of pebbles. Pebbles are finite and can be reused. The game of k moves and m pebbles is G_k^m.
- S wins G_k^m iff a first order sentence with at most k quantifier alterations and m variables can distinguish between the structures.

To use an old example...

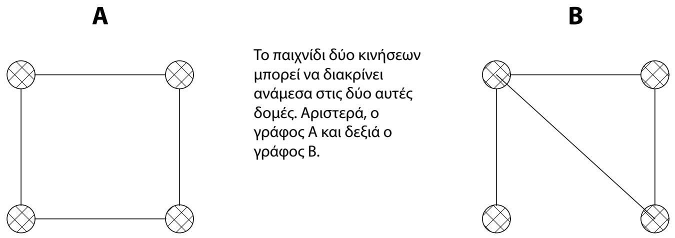

Figure: A two-move game.

By hand:
The $k-$move game shows that graph connectivity and other properties are not expressible in first order logic.

Complexity

- FO is the class of problems that correspond to first order sentences. This will be used sloppily...
- $FO \subseteq \mathcal{L}$, and the reason is that each sentence has a fixed number of quantifiers.
- so, exhaustive search of the structure will do: $k \cdot \log n$ space is needed.
- Also, $FO \neq \mathcal{L}$, from the previous example (the “by hand“ one)

Outline

Built-in relations: Bit, $\leq$ and others

- What is a node in a graph?
- Possible answers:
  1. A node is ... a node
  2. A node is a natural number from $\{1, 2, 3, \ldots, n\}$
- But $1 + 1 = 2$ and $2 \leq 5$, while the $2nd$ bit of 3 is 1...
- Can we use these relations of the natural numbers? Yes. (Do we want to??)
- If $P_1, \ldots, P_k$ are relations in $\mathbb{N}$, $FO(P_1, \ldots, P_k)$ is first-order logic on a vocabulary extended by $P_1, \ldots, P_k$. Structures will have finite subsets from $\mathbb{N}$ as universes and the new symbols will be interpreted accordingly.

AC^0

- *Non-uniform* AC^0 is the class of problems (languages) decided by families of constant-depth polynomial-sized circuits (with unbounded fan-in $\vee$ and $\wedge$ gates).
- *Uniform* AC^0 is the same, but with uniform families of circuits.
- Uniformity: Generated (think of it as "described") by DLOGTIME Turing machines with random access on the input tape.
- It turns out that $FO(all)$ (yes, we include all possible relations from $\mathbb{N}$) is equivalent to non-uniform AC^0. (Perhaps we allowed too much in our language...)
- Also, $FO(+, \times) = FO(Bit, <) = FO(Bit) =$ uniform AC^0. No, I will not prove this.

so, why not include these relations??

Fagin's Theorem. Yes, you *have* seen it before...

Theorem (Fagin's Theorem - 1973)

$\mathcal{NP}$ is equivalent to the class of problems expressible in Second Order Existential Logic ($\exists SO$):

$$\mathcal{NP} = \exists SO$$

Which means that a problem (class of structures) $\mathcal{C}$ is in $\mathcal{NP}$, iff there exists a second order formula $\exists \vec{S} \phi(\vec{S})$, where $\phi(\vec{S})$ is first order, such that for all instances of $\mathcal{C}$ (structures in $\mathcal{C}$), $\mathcal{A}$,

$$\mathcal{A} \in \mathcal{C} \Leftrightarrow \mathcal{A} \models \exists \vec{S} \phi(\vec{S})$$

Furthermore, the theorem still holds for $\phi \in \Pi_2$.

The proof of the theorem...

- ...is long, complicated and nearly boring
- The idea of the proof is...
- Existential quantification can be used to say "There exists a computation, a polynomial...",
- The first-order part can describe the transition function of the TM and limit the steps of the computation by the polynomial.
- Interestingly, we can use built-in relations to make the first-order part universal.

From Fagin's Theorem, $\mathcal{PH} = SO$

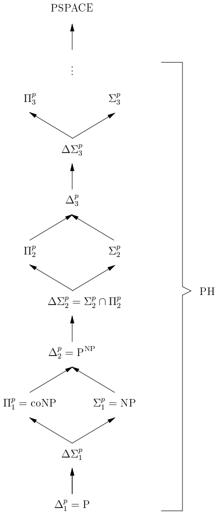

Figure: The Polynomial Time Hierarchy

Outline

Ο Τελεστής ελάχιστου σταθερού σημείου (LFP)

Definition (Μονοτότονη απεικόνιση)

Μια απεικόνιση $(\phi^{\mathcal{A}})$ λέγεται *μονότονη* αν για κάθε R, S,

$$R \subseteq S \rightarrow (\phi^{\mathcal{A}})(R) \subseteq (\phi^{\mathcal{A}})(S)$$

Theorem (Knaster-Tarski)

Έστω R ένα νέο σχεσιακό σύμβολο τάξης k, και έστω $\phi(R, x_1, \ldots, x_k)$ ένας μονότονος πρωτοβάθμιος τύπος. Τότε, για κάθε πεπερασμένη δομή $\mathcal{A}$, το ελάχιστο σταθερό σημείο της $\phi^{\mathcal{A}}(S)$ υπάρχει και ισούται με $(\phi^{\mathcal{A}})^r(\emptyset)$ όπου το r είναι το ελάχιστο για το οποίο $(\phi^{\mathcal{A}})^r(\emptyset) = (\phi^{\mathcal{A}})^{r+1}(\emptyset)$. Επιπλέον, αν $n = \|\mathcal{A}\|$, τότε $r \leq n^k$.

- Με $(LFP_{R^k x_1 \ldots x_k} \phi)$ θα συμβολίζουμε αυτό το ελάχιστο σταθερό σημείο.

LFP

Example

$$REACH \equiv (LFP_{Rxy}\phi)(s,t)$$

$\acute{}Οπου$ $\phi(R,x,y) \equiv x = y \vee \exists z(E(x,z) \wedge R(z,y))$

Definition (FO(LFP))

$FO(LFP)$ είναι το κλείσιμο της Πρωτοβάθμιας Λογικής με τον τελεστή ελάχιστου σταθερού σημείου

Theorem (Θεώρημα Κανονικής Μορφής)

Έστω ϕ τύπος στην $FO(LFP)$. Τότε υπάρχει Πρωτοβάθμιος τύπος ψ και μια σειρά από μεταβλητές $\bar{c}$, ώστε,

$$\phi \equiv (LFP\psi)(\bar{c})$$

Outline

And LFP is...

- $FO(LFP) = P$, when we restrict ourselves on finite *ordered* structures...
- And of course, $FO(LFP)(<) = P$.
- The proof looks like the one of Fagin's Theorem, but here the ordering of the structure plays a significant role. And of course, some things need to be modified to keep the formula positive...
- Without the ordering, we cannot even describe the parity of a set. (Proof? in a while)
- So, once again, what could possibly be wrong with built-in relations and more specifically, why not impose a linear ordering to our structures?

An answer to a question, which brings another question...

- $LFP(<)$ is not a logic, because its sentences are not preserved under isomorphisms.
- But problems in $\mathcal{P}$ do not depend on the (built-in) ordering of a structure. Can't we keep the order-invariant sentences from $LFP(<)$?
- The order-invariant sentences from $LFP(<)$ are not a logic either, because it is an undecidable set.
- In fact, it is an open question, whether a logic exists that captures exactly $\mathcal{P}$.
- A negative answer directly implies $\mathcal{P} \neq \mathcal{NP}$, from Fagin's Theorem.
- A positive answer might help proving this, using a game, like Ehrenfeucht - Fraisse games

Limiting LFP

- $L^k_{\infty,\omega}$ is the extension of FO with infinite disjunctions and conjunctions available, but with only k variables allowed.
- $L^\omega_{\infty,\omega} = \bigcup_k L^k_{\infty,\omega}$
- $LFP \subset L^k_{\infty,\omega}$
- Infinite move pebble games - $L^k_{\infty,\omega}$
- Using games, EVENNESS is not in $L^\omega_{\infty,\omega}$.

Outline

Other fixed-point logics: TC, DTC

- TC stands for transitive closure of a relation (defined by a formula).
- DTC stands for *deterministic* transitive closure: exactly one path.
- Formulas: $TC_{\vec{u}\vec{v}}\phi(\vec{u}\vec{v})$, $DTC_{\vec{u}\vec{v}}\phi(\vec{u}\vec{v})$
- Logics: $FO(TC)$, $FO(DTC), FO(TC)$
- Problems: $(s-t)-$ REACHABILITY, DETERMINISTIC $(s-t)-$ REACHABILITY: $NL, L-$ complete by FO-reductions (what are those?)
- (If you believe that, then) easily, $FO(TC) = \mathcal{NL}$, $FO(DTC) = \mathcal{L}$ (with $BIT, \leq$)

Outline

Once again: What is that?

First-order logic

- Games, limits and complexity of
- Add-ons: Built-in relations

Fagin's Theorem, $\mathcal{NP}$ and $\mathcal{PH}$

$\mathcal{P}$, $\mathcal{L}$ and stuff

- Fixed Points
- $\mathcal{P}$
- $\mathcal{L}, \mathcal{NL}$ and transitive closures

Regular things

- $MSO, \exists MSO$ and Automata

Monadic Second Order Logic on strings

- MSO is SO, with only arity 1 second order quantifiers.
- Similarly, $\exists MSO$: only existential quantifiers.
- We will consider only strings with finite unary relations, P_a, $a \in \Sigma$.
- $P_a(n)$ means that there is an a at the n'th position of the string.

$MSO, \exists MSO$ and Automata

- $Regular \subseteq \exists MSO$ (on strings)
- Proof (idea of a): from a DFA we construct an $\exists MSO$ sentence, where the existential second order quantifiers provide relations of arity 1, that represent the states of the automaton. The first order part ensures that they behave like a DFA: that q_0 is the initial state, that transitions are performed correctly, that only one state satisfies each position of the input, and that the state at the last position of the string has an accepting state.
- Also, $MSO \subseteq Regular$. (without proof, though...)
- Therefore, it follows that on strings, $MSO = \exists MSO$