Logic and Computation: I

Part 3 First order logic and decision problems

Kazuyuki Tanaka

BIMSA

December 22, 2022

北京雁栖湖应用数学研究院
YANQI LAKE BEIJING INSTITUTE OF MATHEMATICAL SCIENCES AND APPLICATIONS

Logic and Computation I

- **Part 1. Introduction to Theory of Computation**
- **Part 2. Propositional Logic and Computational Complexity**
- **Part 3. First Order Logic and Decision Problems**

Part 3. Schedule

- Dec. 8, (1) What is first-order logic?
- Dec.13, (2) Skolem's theorem
- Dec.15, (3) Gödel's completeness theorem
- Dec.20, (4) Ehrenfeucht-Fraïssé's theorem
- Dec.22, (5) Presburger arithmetic
- Dec.27, (6) Peano arithmetic and Gödel's first incompleteness theorem

Presburger arithmetic

Recap

- We consider a language of finitely many relation symbols and constants.
- The (quantifier) rank of a formula measures the entanglement of quantifiers appearing in it. For example, the rank of $\forall y(\forall x \exists y(x = y) \wedge \forall z(z > 0))$ is 3.
- By $\mathcal{A} \equiv_n \mathcal{B}$, we mean that structures $\mathcal{A}, \mathcal{B}$ satisfy the same formulas with rank $\leq n$.
- Given an $\mathcal{A}$ and n, there is the **Scott-Hintikka sentence** $\varphi^n_{\mathcal{A}}$ of rank n such that $\mathcal{B} \models \varphi^n_{\mathcal{A}} \Leftrightarrow \mathcal{B} \equiv_n \mathcal{A}$.
- By $\mathcal{A} \simeq^n \mathcal{B}$, we mean that player II has a winning strategy in $\mathrm{EF}_n(\mathcal{A}, \mathcal{B})$, where n is the round of the game.
- **EF theorem** For all $n \geq 0$, $\mathcal{A} \equiv_n \mathcal{B}$ iff $\mathcal{A} \simeq^n \mathcal{B}$.
- **Corollary** The following are equivalent.
  (1) For any n, there exist $\mathcal{A} \in K$ and $\mathcal{B} \notin K$ such that $\mathcal{A} \equiv_n \mathcal{B}$.
  (2) K is not an elementary class (K cannot be defined by a first-order formula).

Further readings

Jouko Väänänen, Models and Games, Cambridge University Press, 2011.

Dense linear order without end points (DLO)

- The typical models of DLO are $(\mathbb{Q}, <)$ and $(\mathbb{R}, <)$. $(\mathbb{Z}, <)$ is LO but discrete (not dense) since no element exists between n and $n+1$.
- Let $\mathcal{A}, \mathcal{B}$ be two models of DOL. Player II has a winning strategy in $\mathrm{EF}_n(\mathcal{A}, \mathcal{B})$ for all n. Suppose a partial isomorphism between $a_1 < a_2 < \cdots < a_n$ in A and $b_1 < b_2 < \cdots < b_n$ in B are constructed by the players up to the round n. If Player I chooses x_{n+1} between $a_i < a_{i+1}$ (or $b_i < b_{i+1}$), then Player II can extend the partial isomorphism by choosing y_{n+1} between $b_i < b_{i+1}$ (or $a_i < a_{i+1}$).
- Then, for all $n \geq 0$, $\mathcal{A} \simeq^n \mathcal{B}$. By the EF theorem, for all n, $\mathcal{A} \equiv_n \mathcal{B}$, and hence $\mathcal{A} \equiv \mathcal{B}$. In particular, $(\mathbb{Q}, <) \equiv (\mathbb{R}, <)$.
- Then, DLO is a complete theory. Therefore, it is decidable.
  - If it is not complete, then there is a sentece σ which is neither provable nor disprovable.
  - That is, both $\mathrm{DLO} \cup \{\neg\sigma\}$ and $\mathrm{DLO} \cup \{\sigma\}$ are consistent. So, each has its own model, but they are no longer elementary equivalent, which is a contradiction.
- A complete theory is characterized as Th$(\mathcal{A})$ for its arbitrary model $\mathcal{A}$. DLO is often treated as Th$(\mathbb{Q}, <)$.

Theorem (1)

DLO is a PSPACE-complete problem.

Proof. First, we show that DLO is PSPACE-hard, by reducing TQBF to DLO in polynomial time. It was shown in Part 2 of this course, TQBF (true quantified Boolean formula) is PSPACE-complete.

- Let A be a QBF and transform it to a PNF $Q_1 x_1 Q_2 x_2 \ldots Q_n x_n B(x_1, x_2, \ldots, x_n)$, where $B(x_1, x_2, \ldots, x_n)$ is a Boolean formula.
- Then, define a DLO formula $A_<$ as follows.

$$Q_1 x_1 Q_1 y_1 Q_2 x_2 Q_2 y_2 \ldots Q_n x_n Q_n y_n B(x_1 < y_1, x_2 < y_2, \ldots, x_n < y_n).$$

- For example, for a QBF $A \equiv \forall x_1 \exists x_2 \forall x_3 ((x_1 \wedge x_2) \vee \neg x_3)$, $A_<$ in DLO is

$$\forall x_1 \forall y_1 \exists x_2 \exists y_2 \forall x_3 \forall y_3 (((x_1 < y_1) \wedge (x_2 < y_2)) \vee \neg (x_3 < y_3)).$$

- An atomic formula $x_i < y_i$ in $A_<$ simply plays the role of variable x_i in A. Then A is true in a simple Boolean algebra $\{0, 1\}$ iff $A_<$ is true in any model of DLO.
- Since the lengths of A and $A_<$ differ only by constant multiples, TQBF is reduced to DLO in polynomial time.

Next, we show that DLO is PSPACE, following the proof that TQBF is PSPACE.

- First, assume a DLO formula is given in PNF $Q_1 x_1 Q_2 x_2 \ldots Q_n x_n \; C(x_1, x_2, \ldots, x_n)$ (with no quantifiers in $C(x_1, x_2, \ldots, x_n)$).
- In general, we can determine the truth value of $C(x_1, x_2, \ldots, x_n)$ by specifying elements of DLO substituting for variables $x_1, x_2, \ldots, x_n$. Here only the relations of the elements are enough to determine the truth value.
- Now, we first fix x_1 is arbitrarily. Next, the necessary information on x_2 is whether it is larger, smaller, or equal to x_1.
- If Q_2 is $\forall$ ($\exists$), all the three cases (one of the three cases) must hold. Without loss of generality, we may assume $x_1 < x_2$.
- Next, there are five cases for x_3 as illustrated by the red arrows:

  If Q_3 is $\forall$ ($\exists$), all the five cases (one of the five cases) should hold.
- Since the number of cases for variable x_i is less than $2i - 1$, there are less than $(2n-1)!$ cases to check in total.
- In order to execute this computation, we need $\log((2n-1)!) = O(n \log n)$ space to keep records. Thus, it is textsfDSPACE$(n \log n)$, hence also PSPACE.

We next apply the EF theorem to the problem of length of finite linear orders.

Lemma (Gurevich)

For any $m > 0$, for any two finite linear sequences L_1, L_2 of length 2^m or greater, $L_1 \equiv_m L_2$.

Proof.

- A finite linear order of length n is denoted by $[n] = (n, <)$, where n of $(n, <)$ is identified with $\{0, 1, \ldots, n-1\}$.
- For each k, we introduce a threshold absolute value $|x|_k$ by $|x|_k = |x|$ if $|x| < 2^k$; $|x|_k = \infty$, otherwise.
- Select l elements from $[n]$ and arrange them in ascending order as $\vec{a} = (a_1, a_2, \ldots, a_l)$.
- Similarly, select l elements from $[n']$ and arrange as $\vec{b} = (b_1, b_2, \ldots, b_l)$.
- Let I_k the a collection of all partial isomorphisms $\vec{a} \mapsto \vec{b}$ that satisfy the following conditions: if $a_0 = b_0 = 0$, $a_{l+1} = n$, $b_{l+1} = n'$, then

  for any $i \leq l$, $|a_{i+1} - a_i|_k = |b_{i+1} - b_i|_k$ holds.
- Note that by $\varnothing \in I_k$ we mean $|n|_k = |n'|_k$.

- Now, suppose $\vec{a} \mapsto \vec{b} \in I_k$. We can show that for any $a \in n$, there exists a $b \in n'$ such that $\vec{a}a \mapsto \vec{b}b \in I_{k-1}$ holds. Here, $\vec{a}a$ and $\vec{b}b$ are rearranged in order.
- First consider the case $|a_{i+1} - a_i|_k = |b_{i+1} - b_i|_k < \infty$. If $a_{i+1} > a > a_i$, then $|a_{i+1} - a|_{k-1} < \infty$ or $|a - a_i|_{k-1} < \infty$ holds, and so the value of b is also uniquely determined by b_{i+1} or b_i.
- Next assume $|a_{i+1} - a_i|_k = |b_{i+1} - b_i|_k = \infty$. If $a_{i+1} > a > a_i$ then $|a_{i+1} - a|_{k-1} = \infty$ or $|a - a_i|_{k-1} = \infty$ holds; if one is $< \infty$, then the value of b is uniquely determined by the corresponding b_{i+1} or b_i; if both are ∞, the value of b can also be taken from both sides to ∞.
- Therefore, if $n = n'$ or $n, n' \geq 2^m$, then we obtain $\varnothing \in I_m$.
- In particular, if $n, n' \geq 2^m$, then $[n] \equiv_m [n']$. $\square$

Theorem (2)

For finite linear orders, there is no first-order formula expressing the parity of its length.

Proof Assume, for the sake of contradiction, we have such a formula φ. Let $\mathrm{qr}(\varphi) = m$. Then by the above lemma, for linear sequences longer than 2^m, we cannot tell whether its length is even or odd, which is a contradiction. $\square$

The connectivity of graphs

- We can show the connectivity of graphs cannot be defined by a first-order formula by reducing the parity problem of linear orders to it. We first make a special graph from a linear order.
- In the linear order $<$, let $\mathrm{succ}(x, y) \equiv (x < y) \wedge \forall z(z \leq x \vee y \leq z)$ and $\mathrm{succ2}(x, y) \equiv \exists z(\mathrm{succ}(x, z) \wedge \mathrm{succ}(z, y))$.
- Also let $\mathrm{first}(x) \equiv \neg\exists y\ \mathrm{succ}(y, x)$, and $\mathrm{last}(x) \equiv \neg\exists y\ \mathrm{succ}(x, y)$
- Then, define $\mathrm{edge}(x, y)$ as follows.
  $$\mathrm{edge}(x, y) \equiv \mathrm{succ2}(x, y) \vee$$
  $$((\exists z(\mathrm{succ}(x, z) \wedge \mathrm{last}(z)) \wedge \mathrm{first}(y))) \vee (\mathrm{last}(x) \wedge (\exists z(\mathrm{first}(z) \wedge \mathrm{succ}(z, y))))$$
  By this formula, we make a graph by connecting every other points in a line by an edge, and also by going back to the first point from the second last point and also to the second point from the last point.
- If a linear order has even number of points, the graph becomes two cycles (disconnected), and if odd number, it results in a single cycle.
- In other words, if the connectivity of a graph can be defined, then the parity of the length of a linear order can be defined, which is a contradiction.

Homework

Given a finitely connected graph, the existence of an Eulerian cycle in it cannot be described in first-order logic.

- To expand the scope of application of the EF theorem, we would like to consider structures with functions.
- Rewriting functions as relations requires the use of extra quantifiers for function composition, and the need to use more complicated formulas for atomic formulas involving functions.
- However, there is not much problem when dealing with arbitrary ranks. For example, the following argument is possible for groups.
- $G_1 \equiv G_2 \Rightarrow G_1 \times H \equiv G_2 \times H$ for three groups G_1, G_2, H. For this proof, we observe that II's winning play $\vec{g_1} \leftrightarrow \vec{g_2}$ in $\mathrm{EF}_n(G_1, G_2)$ can be modified as II's winning play $(\vec{g_1}, \vec{h}) \leftrightarrow (\vec{g_2}, \vec{h})$ in $\mathrm{EF}_n(G_1 \times H, G_2 \times H)$.

Presburger arithmetic

- There are various methods of applying computational models such as automata to solve decision problems.
- As a typical example, let us consider its application to first-order Presburger arithmetic, which has only addition operation on natural numbers. The technique here will be extended to second-order logic in the next semester.
- Presburger arithmetic is a first order theory for structure $\mathcal{N} = (\mathbb{N}, 0, 1, +)$ in the language $\mathcal{L}_{\mathrm{P}} = \{0, 1, +\}$.
- We want to find a method to determine whether or not $\mathcal{N} \models \sigma$ holds for a sentence σ in the language $\mathcal{L}_{\mathrm{P}}$.
- Note that in Presburger arithmetic, $<$ is defined as $x < y \leftrightarrow \exists z(x + z + 1 = y)$. The congruence relation $\equiv_k$ is also defined. Then Presburger arithmetic with $<$ and $\equiv_k$ admits the elimination of quantifiers, which is another method of solving the decision problem.

- First, let us consider how to express the sequence of natural numbers $(n_1, n_2, \ldots, n_s)$ (where $s > 0$) in terms of a word that recognized by the automaton.
- The alphabet Ω_s is a set of vertical vectors of length s with elements $0, 1$. So, Ω_s consists of 2^s symbols defined by

$$\vec{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_s \end{bmatrix} \quad \text{where } b_1, b_2, \ldots, b_s = 0 \text{ or } 1.$$

  We may also write $\vec{b} = {}^t[b_1, b_2, \ldots, b_s]$.
- A word $\vec{b}_1 \vec{b}_2 \ldots \vec{b}_t$ over Ω_s can be expressed as

$$\begin{bmatrix} b_{11} \\ b_{12} \\ \vdots \\ b_{1s} \end{bmatrix} \begin{bmatrix} b_{21} \\ b_{22} \\ \vdots \\ b_{2s} \end{bmatrix} \cdots \begin{bmatrix} b_{t1} \\ b_{t2} \\ \vdots \\ b_{ts} \end{bmatrix} = \begin{bmatrix} b_{11} b_{21} \ldots b_{t1} \\ b_{12} b_{22} \ldots b_{t2} \\ \vdots \\ b_{1s} b_{2s} \ldots b_{ts} \end{bmatrix}$$

- An s-tuple $(n_1, n_2, \ldots, n_s)$ of natural numbers are represented by $\vec{b}_1\vec{b}_2\ldots\vec{b}_t$ as follows.

$$\begin{gathered}
n_1 = b_{11} + b_{21} \cdot 2 + \cdots + b_{t1} \cdot 2^{t-1} \\
n_2 = b_{12} + b_{22} \cdot 2 + \cdots + b_{t2} \cdot 2^{t-1} \\
\vdots \\
n_s = b_{1s} + b_{2s} \cdot 2 + \cdots + b_{ts} \cdot 2^{t-1}
\end{gathered}$$

- In other words, the binary representation of natural number n_i is $b_{ti}b_{(t-1)i}\ldots b_{1i}$.

- So, if we add the zero vector $\vec{0}$ to the right of the word $\vec{b}_1\vec{b}_2\ldots\vec{b}_t$, the resulting sequence $\vec{b}_1\vec{b}_2\ldots\vec{b}_t\vec{0}$ represents the same sequence $(n_1, n_2, \ldots, n_t)$ of natural numbers. But if we add $\vec{0}$ to the left of $\vec{b}_1\vec{b}_2\ldots\vec{b}_t$, the resulting sequence $\vec{0}\ldots\vec{b}_t$ represents $(2n_1, 2n_2, \ldots, 2n_s)$.

- Note that the zero vector $\vec{0}$ is different from the empty string

$$\varepsilon = \begin{bmatrix} \\ \\ \\ \end{bmatrix}.$$

- Since an s-tuple of natural numbers $(n_1, n_2, \ldots, n_s)$ (where $s > 0$) can be expressed as words over Ω_s, we next consider the set of $(n_1, n_2, \ldots, n_s)$ that satisfies a given formula $\varphi(x_1, x_2, \ldots, x_s)$ and whether an automaton can accept the language of words representing such a set.
- First, an atomic formula in Presburger arithmetic is expressed as follows.

$$a_1x_1 + a_2x_2 + \cdots + a_sx_s = b, \qquad \cdots \qquad (\star)$$

where a_ix_i is short for $\pm\underbrace{(x_i + x_i + \cdots + x_i)}_{|a_i| \text{ copies}}$ and b for $\pm\underbrace{(1 + 1 + \cdots + 1)}_{|b| \text{ copies}}$.

- Note that a_i's and b may be negative because terms are transposed to express a formula as $(\star)$.
- Also, we may assume $s > 0$, since by setting $a_i = 0$, you can add the variable x_i meaninglessly.

- Let $\vec{c} = {}^t[c_1, c_2, \ldots, c_s]$ be the first letter of the word representing the solution $(n_1, n_2, \ldots, n_s)$ of Equation $(\star)$.
- Then, let $(n'_1, n'_2, \ldots, n'_s)$ be the sequence of numbers represented by the remaining strings excluding $\vec{c}$. Then for each i,

$$n_i = c_i + 2n'_i.$$

Hence,

$$a_1 n'_1 + a_2 n'_2 + \cdots + a_s n'_s = \frac{b - \Sigma_i a_i c_i}{2}.$$

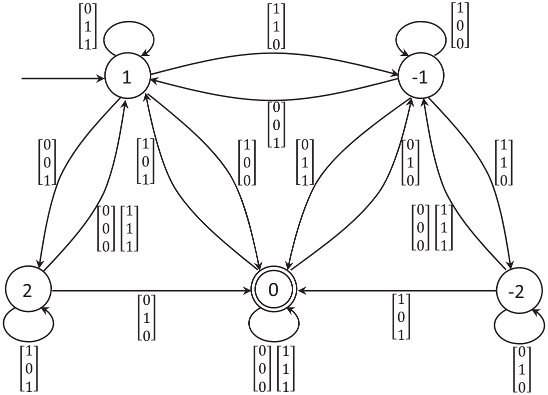
- Let $M = |b| + \Sigma_i |a_i|$. For any ${}^t[c_1, c_2, \ldots, c_s] \in \Omega$, $|\sum_i a_i c_i| \leq \Sigma_i |a_i| \leq M$. Then for any $b' \in [-M, M]$, $\frac{b' - \Sigma_i a_i c_i}{2} \in [-M, M]$.
- Now define an automaton $\mathcal{M} = (Q, \Omega_s, \delta, q_0, F)$ for Equation $(\star)$ by:
    - the set of states Q are the integer in the interval $[-M, M]$.
    - transition function $\delta : Q \times \Omega \to Q$ is

$$\delta(q, \vec{c}) = \frac{q - \Sigma_i a_i c_i}{2}$$

    - the initial state $q_0 = b$,
    - the set of final states $F = \{0\}$.

Example

The transition of an automaton for $x_1 + 2x_2 - 3x_3 = 1$ is shown as follows. If the next state of $\delta(q, \vec{c})$ is not indicated, it will enter the deadlock state $\perp \notin F$, in which the automaton cannot leave.



- At 1, if the first input symbol is ${}^t[0,0,0]$, it immediately enter the deadlock becasue of no outgoing arrow. For such a input, n_1, n_2, and n_3 are all multiples of 2, and so they can not be a solution of $x_1 + 2x_2 - 3x_3 = 1$.
- On the other hand, it accepts the word ${}^t[1,1,0]{}^t[0,1,1]$, which represents $(n_1, n_2, n_3) = (1, 3, 2)$.

- An automaton thus defined accepts the language of words representing s-tuples $(n_1, n_2, \ldots, n_s)$ that satisfy the atomic formula $\varphi(x_1, x_2, \ldots, x_s)$.
- It is also easy to extend an automaton expressing an atomic formula to that for a Boolean combination of them, since the class of regular languages is closed under Boolean operations.
- It is also easy to add quantifiers. If $\mathcal{M} = (Q, \Omega_s, \delta, q_0, F)$ is a deterministic automaton corresponding to a formula $\varphi(x_1, x_2, \ldots, x_s)$, then a nondeterministic automaton $\mathcal{M}' = (Q, \Omega_{s-1}, \delta', \{q_0\}, F)$ corresponding to $\exists x_1 \varphi(x_1, x_2, \ldots, x_s)$ can be constructed as follows.

$$\delta'(q, {}^t[c_2, \ldots, c_s]) = \{\delta(q, {}^t[b, c_2, \ldots, c_s]) : b = 0, 1\}$$

Then $\mathcal{M}'$ accepts a word representing $(n_2, \ldots, n_s)$ iff $\mathcal{M}$ accepts a word representing $(n_1, n_2, \ldots, n_s)$ for some n_1. Note that a nondeterministic automaton can always be transformed into a deterministic automaton.

- The universal quantifier $\forall x$ can be rewritten as $\neg\exists x\neg$.
- Thus, for every formula $\varphi(x_1, x_2, \ldots, x_s)$ in Pressburger arithmetic, we can construct an automaton accepting the language of words representing s-tuples $(n_1, n_2, \ldots, n_s)$ that satisfy the formula $\varphi(x_1, x_2, \ldots, x_s)$.
- For a sentence σ, it can be treated by adding a meaningless variable, and the truth of the sentence can be determined by whether the language accepted by automaton is empty or Ω_1^*.
- Therefore, we obtain the following theorem.

Theorem

Presburger arithmetic is decidable.

Summary

- By the EF theorem, DLO is decidable.
- DLO is PSPACE-complete. TQBF is polynomial-time reducible to DLO.
- (Gurevich) For any $m > 0$, for any two finite linear sequences L_1, L_2 of length 2^m or greater, $L_1 \equiv_m L_2$.
- For finite linear orders, there is no first-order formula expressing the parity of its length.
- The connectivity of a graph cannot be defined by a first-order formula.
- For every formula $\varphi(x_1, x_2, \ldots, x_s)$ in Presburger arithmetic, we can construct an automaton accepting the language of words representing s-tuples $(n_1, n_2, \ldots, n_s)$ that satisfy the formula $\varphi(x_1, x_2, \ldots, x_s)$.
- Presburger arithmetic is decidable.

Thank you for your attention!