Seminar Proof

KAIST SoC & Mathmatics 17 Kim Tae Young

August 1, 2019

1 Presburger Arithmetic

Theorem 1.1. *A theory is decidable if quantifier free formula of theory is decidable, and theory has quantifier elimination procedure.*

Proof. We prove by induction on depth of quantifier in F. When there are no quantifier, it is done by assumption.
Now suppose formula with n nested quantifier is decidable and F has n+1 quantifier depth. Then by quantifier elimination, we eliminate all deepest quantifiers. We have n quantifier depth formula equivalent to F, which is decidable by induction hypothesis. So Proof is done. □

Theorem 1.2. *Extended Presburger Arithmetic* $Th(\mathbb{N}, 0, 1, +, <, \{c : \cdot\}_{c>0})$ *has quantifier elimination procedure.*

Proof. First, we will find equivalent formula to $\exists x.F$ in form $\vee_i((\exists x.F_i)\wedge F_i')$, where F_i is in form $\exists x.(\wedge q_i(\vec{y_i} < a_i \cdot x)\wedge)$. Then we will find equivalent formula G.
The construction is as following.
First, we push all negation by applying de morgan's law, than find DNF form equivalent to it.
Then for each clause in DNF, if literal contains x it is in F_i, if not it is in F_i'. Now clauses in F_i will be either one of these.

$$\begin{aligned} c_i | a_i x + p_i(\vec{y_i}) \\ c_i \nmid a_i x + p_i(\vec{y_i}) \\ c_i < a_i x + p_i(\vec{y_i}) \\ c_i > a_i x + p_i(\vec{y_i}) \end{aligned}$$

Note that a_i is nonzero.
For $c_i \nmid a_i x + p_i(\vec{y_i})$, we can replace it by $\vee_{j=1}^{c_i-1} c_i | a_i x + p_i(\vec{y_i}) + j$. Then our formula is equivalent to some formula in form $\vee(\wedge F_i \wedge F_i')$ where F_i' is quantifier free, and $F_i = \exists x.(\wedge_{i\in L} q_i(\vec{y_i}) < a_i \cdot x_i) \wedge (\wedge_{j\in U} p_j(\vec{y_j}) > a_j \cdot x_j) \wedge (\wedge_{k\in D} c_k | a_i \cdot x_i + r_k(\vec{y_k}))$.
Claim : F_i is equivalent to $H = \wedge_{i\in L} \frac{b}{a_i} \cdot q_i(\vec{y_i}) < x \wedge \wedge_{j\in U} \frac{b}{a_j} \cdot p_j(\vec{y_j}) > x \wedge \wedge_{k\in D}(\frac{b}{a_k} \cdot c_k) | (x + \frac{b}{a_k} \cdot r_k(\vec{y_k})) \wedge b | x$. where $b = lcm(\{a_i : i \in L \cup U \cup D\})$
(Multiply $\frac{b}{a_i}$ to both side in each clause, and substitute bx to y, then we will get equivalent formula when added y is divisible by b.)
Then finally, by letting $c = lcm(\{b\} \cup \{b \cdot \frac{c_k}{a_k} : k \in D\})$

$$\exists x.H \equiv \begin{cases} \vee_{0\leq m<c} H[m/x] & \text{if } L = \emptyset \\ \vee_{i\in L} \vee_{q\leq m<c} H[(\frac{b}{a_i} \cdot q_i(\vec{y_i}) + m)/x] & \text{if } L \neq \emptyset \end{cases}$$

□

2 Computbility Theory

Theorem 2.1. *Halting Problem is Uncomputable.*

Proof. Suppose algorithm A computes Halting Problem, that it returns 1 if and only if algorithm B halts on input x. Consider new machine A', that gets input and duplicate it and input it as input to A. So A' looks like

```
x = input()
res = A(x, x)
if res == 1:
    infinite_loop()
else:
    halt()
```

Now think we input A' into A', then

- Suppose A' halts on input A'. Then res has value 1, so it enters infinite loop meaning it doesn't halt, which is contradiction.

- Suppose A' does not halt on input A'. Then res has value 0, so it halts. This is contradiction.

So our assumption is false, so Halting Problem is uncomputable. □

Theorem 2.2. *Post Correspondence Problem is Uncomputable.*

Proof. Here our proof is reduction to halting problem.
Our tile will be as following. For input $x = x_1, \dots, x_n$, initial state q_1
$w_1 = |q_1 x_1| \dots |x_n|\#, v_1 = |$
For every alphabet $\sigma \in \Sigma$, $w_\sigma = |\sigma, v_\sigma = \sigma|$
For each transistion

- $\delta(q_i, \sigma_i) = (q_j, \sigma_j, L)$
  $\forall \sigma \in \Sigma(|q_j\sigma|\sigma_j, \sigma|q_i\sigma_i|)$
  if $\sigma_j = \#$, $(|q_j\sigma|\sigma_j, \sigma|q_i\sigma_i|\#|)$
- $\delta(q_i, \sigma_i) = (q_j, \sigma_j, R)$
  $\forall \sigma \in \Sigma(|sigma_j|q_j\sigma, q_i\sigma_i|\sigma|)$
  if $\sigma_i = \#$, $(|\sigma_j|q_j\#, q_i\#|)$
- $\delta(q_i, \sigma_i) = (q_j, \sigma_j, S)$
  $(|q_j\sigma_j, q_i, \sigma_i|)$
  if $\sigma_i = \#, \sigma_j \neq \#$, $(q_j\sigma_j|\#, q_i\sigma_i|)$

For each halting state $q_f \in Q_F, \forall \sigma, \sigma' \in \Sigma, (|q_f\sigma', \sigma|q_f\sigma'|), (|\#|q_f\sigma', \#|q_f\sigma|\sigma'|), (|, q_f\#|)$
Then this problem has solution if and only if this turing machine halts for input x. □

3 General First Order Logic

Theorem 3.1. *Validity for General First Order Logic is undecidable.*

Proof. Given $(w_1, v_1), \dots, (w_n, v_n)$ where $w_i = w_{i,1}, \dots, w_{i,l_i}$, $v_i = v_{i,1}, \dots, v_{i,l'_i}$ Consider Formula defined as following over signature (ϵ, f_0, f_1, P) where ϵ is constant symbol, f_0, f_1 are unary function symbol, P is binary predicate symbol.

$$\begin{aligned}
F_1 &= \wedge_{i=1}^{n} P(f_{w_{i,1}}(\dots f_{w_{i,l_i}}(\epsilon)), f_{v_{i,1}}(\dots f_{v_{i,l'_i}}(\epsilon))) \\
F_2 &= \forall x.\forall y. \wedge_{i=1}^{n} P(x,y) \to P(f_{w_{i,1}}(\dots f_{w_{i,l_i}}(x)), f_{v_{i,1}}(\dots f_{v_{i,l'_i}}(y))) \\
F_3 &= \exists z.P(z,z) \\
F &= F_1 \wedge F_2 \to F_3
\end{aligned}$$

Then our post correspondence problem has solution if and only if this formula F has strcture of string concatenation and P denoting generated by each pair. Since if this formula is valid this formula is true for above structure, general first order logic validity is undecidable. $\square$