Tutorial 7

Om Swostik

15.9(ii):

Asssume for contradiction there exists a FO sentence F without $=$ such that F doesn't satisfy any model with > 2 elements.

Let m be a model over $\{a, b\}$ such that $m \models F$. (with any signature)

Construct a model $m^{'}$ (over the same signature) such that $D_{m'} = \{a, b, c\}$ and $m^{'}(x) = m(x)$ $(\forall x \in FVars)$

And $m^{'}(c) = a$, $m^{'}$ is identity over $\{a, b\}$.

If $f \in R$, $f_{m'}(l_1, \ldots, l_n)$ is true if $f_m(m^{'}(l_1), \ldots, m^{'}(l_n))$ holds. $(l_i \in \{a, b, c\})$

For $g \in F$, $g_{m'}(l_1, \ldots, l_n) = g_m(m^{'}(l_1), \ldots, m^{'}(l_n))$ $(l_i \in \{a, b, c\})$

(Basically replace all occurences of c with a)

Observe that, since we don't have $=$, $m^{'} \models F$.

Contradiction. (since $|D_{m'}| = 3$)

16.6:

This is a tricky problem. It is quite tempting to try $\forall$-elim n times and substitute the appropriate terms (wrt σ)

However, this approach fails as the following example shows:

Consider the formula $\forall x_1, x_2, x_3 \; x_1 = x_3$ and let $\sigma = [x_1 \to x_2, x_2 \to x_1]$

If we try to apply $\forall$-elim starting from x_1 and substituting each term (according to σ), we get the formula $x_1 = x_3$.

However, $F\sigma = (x_2 = x_3)$ (As x_2 doesn't occur in F)

A possible solution to this issue is the following:

We use extra variables $y_1, y_2 \dots y_n$ (y_i are fresh variables not in F and distinct from x_i)

If σ maps x_i to t_i, we replace all occurences of x_i in t_i with y_i to create a new term $t_i^{'}$

We apply $\forall$-elim on the x_i's and replace with the corresponding terms $t_i^{'}$

We obtain a formula $F^{'}$ which has some y_i's occuring in it (we want to replace these y_i's with x_i's to obtain $F\sigma$)

Now apply $\forall$-intro for y_i's in order, starting from y_1 (Why can we do this?)

Now, apply $\forall$-elim again (on the the universally quantified y_i's) and replace each term with the corresponding x_i to get $F\sigma$ (Check that $\forall$-elim can be applied in this context)

With this, we get a proof of atmost $2n$ steps

16.8:

We will simulate $\forall$-elim using $\exists$-def and other proof rules

t refers to any arbitrary term

1. $\Sigma \vdash \forall x F(x)$ (Premise)
2. $\Sigma \vdash \neg\exists x(\neg F(x))$ ($\exists$-def-1)
3. $\Sigma \cup \{\neg F(t)\} \vdash \neg F(t)$ (Associativity)
4. $\Sigma \cup \{\neg F(t)\} \vdash \exists x(\neg F(x))$ ($\exists$-intro-3)
5. $\Sigma \cup \{\neg F(t)\} \vdash \neg\exists x(\neg F(x))$ (Monotonicity-2)
6. $\Sigma \vdash \neg\neg F(t)$ (By-Contra-4-5)
7. $\Sigma \vdash F(t)$ (Double-neg-elim-6)

18.10:

We are trying to prenex a formula F such that the sum of function parameters after skolemization is minimum.

First, we remove all occurences of $\implies$ from F. This can be done in linear time.

We end up a with a formula $F^{'}$ which has only $\vee$ and $\wedge$ as its binary connectives.

Now, the main idea is to divide $F^{'}$ into 'chunks' i.e divide $F^{'}$ into separetely quantified blocks, where we call each block a 'chunk'

For example, if $F^{'} = (\forall x, y \exists w(R(x, y, z))) \vee (\exists a, b \forall c(E(a, b, c)))$, then $F^{'}$ has two chunks.

These are $\forall x, y \exists w(R(x, y, z))$ and $\exists a, b \forall c(E(a, b, c))$.

We associate a sequence to each chunk in $F^{'}$. This is done by breaking each chunk into 'pieces'.

For each chunk, go through its quantifiers such that whenever a $\exists$ is encountered (or the end is reached), break that part and call it a 'piece'

In the example above, in the first chunk of $F^{'}$ we have only one piece i.e $\forall x, y \exists w$

while in the second chunk, we have three pieces $\exists a$, $\exists b$ and $\forall c$.

Note that if there are no $\exists$ occuring in the chunk, then the $\forall$s form one single piece.

For each piece in a given chunk, the value of that piece is the number of $\forall$'s occuring in it

The set of values of the pieces naturally generate a sequence for each chunk of the formula

In the example, the sequences are 2 and 0 0 1.

The algorithm goes as follows:

1. Store the sequences in separate linked lists with the pointer to the head of each list stored in another list (call it L, L is doubly linked to make deletions easier).

2. Compare the values at the head of each linked list (while going through L) and prenex the chunk with the minimum value and move the head of that list (with the minimum) to the list$\rightarrow$ next (if list $\rightarrow$ next $\neq$ NULL).

   If list $\rightarrow$ next $=$ NULL, delete the pointer to the head of that list from L

3. Stop iterating when L is empty

We obtain a prenexed formula G which we can guarantee will produce minimal number of parameters after skolemization.