CSC438F/2404F **Solutions to Problem Set 2** **Fall, 2015**

Due: Friday, October 23, beginning of tutorial
Total marks: 35

[5] 1. Prove the soundness of the LK rule $\exists$ Left. That is, prove that the universal closure of the meaning of the bottom sequent is a logical consequence of the universal closure of the meaning of the top sequent.

Show where in your proof you are using the restriction that the free variable b must not occur in the conclusion.

(See pages 28-29 of the Notes. Note that the proof on page 29 for the $\forall$-right case is confusing, because the left and right side of $\models$ should be replaced by the universal closures of those formulas, so neither side has any free variables, so there is no need to discuss the object assignment σ.)

Solution:
We claim that it suffices to prove

$$\forall y[\neg(A(y) \wedge \bigwedge \Gamma) \vee \bigvee \Delta] \qquad \models \qquad [\neg(\exists x A(x) \wedge \bigwedge \Gamma) \vee \bigvee \Delta]$$

Here the intention is that we replace occurrences of b throughout by y, and we are using the restriction that b cannot occur in either Γ or Δ.

Now we simply note that the two formulas on either side of $\models$ are equivalent: The first is a prenex form of the second.

[10] 2. Give an LK proof of the sequent

$$\forall x(x+0=x) \rightarrow \forall x \forall y(x+(y+0)=x+y)$$

You do not need to put in weakenings or exchanges. Indicate which LK equality axioms you use.

Solution:
The LK equality axioms are

L1: $\rightarrow a=a$

L4: $a=a, b+0=b \rightarrow a+(b+0)=a+b$

Here is the LK proof:

L4 L1	cut
$b+0=b \rightarrow a+(b+0)=a+b$	$\forall$ left
$\forall x(x+0=x) \rightarrow a+(b+0)=a+b$	$\forall$ right twice
$\forall x(x+0=x) \rightarrow \forall x \forall y(x+(y+0)=x+y)$	

[5] 3. Let $\mathcal{L}$ be a language consisting of a countably infinite set $\{c_1, c_2, \ldots\}$ of constant symbols and the binary predicate symbol P, and also $=$. Let Γ be the set of sentences

$$\Gamma = \{c_i \neq c_j \mid i, j \in \mathbb{N} \text{ and } i < j\}$$

Let A be an $\mathcal{L}$ sentence such that $\Gamma \models A$. Prove that A has a finite model.

Solution:
By compactness, since $\Gamma \models A$ it follows that $\Gamma_0 \models A$ for some finite subset $\Gamma_0 \subseteq \Gamma$. Let n be the largest index i such that c_i occurs in Γ_0.

Define a finite structure $\mathcal{M}$ as follows:

$M = \{0, 1, ..., n\}$
$c_i^{\mathcal{M}} = i$, $0 \leq i \leq n$
$P^{\mathcal{M}} = \varnothing$.

Then $\mathcal{M} \models \Gamma_0$, and since $\Gamma_0 \models A$, it follows that $\mathcal{M} \models A$.

4. Let us call a formula A of the predicate calculus *simple* if all predicate symbols in A are unary, and A has no function symbols (and no constants).

[10] a) Show that if A is a satisfiable simple formula with n predicate symbols, then A is satisfied by some structure whose universe has at most 2^n elements. (Hint: Start with a model for A with universe M, and define a certain equivalence relation on M. Show that there are at most 2^n equivalence classes. Now show how to define a model for A whose universe is the set M' of these equivalence classes. It may help to look at the proof of Lemma 2 on page 45 of the Notes.)

[5] b) Show that the set of valid simple formulas is decidable. (That is, give an algorithm which, given a simple formula A, determines whether A is valid. Your algorithm should halt on all inputs.)

Solution:
REMARK: Unary predicate symbols are sometimes called monadic.

a) Let $\mathcal{M}$ be a structure with universe M, and let σ_0 be an object assignment to M such that

$$\mathcal{M} \models A[\sigma_0] \tag{1}$$

Let $P_1, ..., P_n$ be the predicate symbols in A. Define an equivalence relation $\sim$ on M as follows: $u \sim v$ iff for each $i \in \{1, ..., n\}$,

$$u \in P_i^{\mathcal{M}} \text{ iff } v \in P_i^{\mathcal{M}}$$

Thus each element in M has an associated n-tuple of bits indicating its membership in each of the n sets $P_1^{\mathcal{M}}, ..., P_n^{\mathcal{M}}$. Two elements are equivalent iff their associated n-tuples are the same. Hence there are at most 2^n equivalence classes.

Now define a new structure $\hat{\mathcal{M}}$ as follows. The universe $\hat{M}$ is the set of equivalence classes under $\sim$. For each $u \in M$ let $[u]$ be its associated equivalence class. Then for $i = 1, ..., n$ define $[u] \in P_i^{\hat{\mathcal{M}}}$ iff $u \in P_i^{\mathcal{M}}$. Notice that this definition does not depend on our choice of the representative u in the equivalence class $[u]$.

For each object assignment σ to M define the object assignment $\hat{\sigma}$ to $\hat{M}$ by

$$\hat{\sigma}(x) = [\sigma(x)]$$

for each variable x.

Lemma: *For every formula B in the language of A, and for every object assignment σ to M,*

$$\mathcal{M} \models B[\sigma] \text{ iff } \hat{\mathcal{M}} \models B[\hat{\sigma}]$$

The proof is by structural induction on B. The base case is when B is an atomic formula $P_i x$, and the Lemma follows from our definition of $\hat{\mathcal{M}}$ and $\hat{\sigma}$. The induction step is straightforward from the Basic Semantic Definition.

The problem follows from the Lemma and our assumption (1).

REMARK: The Lemma above is really a simplified version of Lemma 2 in the proof of the Equality Theorem (see page 45 of the Notes). If $=$ is interpreted as $\sim$ in the valuation σ, then $\mathcal{M}$ satisfies all equality axioms for the language of A, augmented by the symbol $=$.

b) Notice that A is valid iff $\neg A$ is unsatisfiable.

Given a simple formula A with n monadic predicate symbols $P_1, \ldots, P_n$, for each structure $\mathcal{M}$ with universe $M = \{1, 2, \ldots, 2^n\}$, where $\mathcal{M}$ assigns a relation $P_i^{\mathcal{M}}$ to each predicate symbol P_i in A, determine whether $\mathcal{M}$ satisfies A. Accept A iff every such structure satisfies A.

The correctness of the algorithm follows from the proof of part a). Note the proof of part a) can be modified to show that if A is satisfied by some structure, then it is satisfied by a structure with exactly 2^n elements. (Just pad out one the equivalence classes to add more elements.)