CS 476 Homework #8 Due 10:45am on 3/31

Note: Answers to the exercises listed below, as well as the relevant Maude code for Problem 2, should emailed in *typewritten form* (latex formatting preferred) by the deadline mentioned above to `abir1@illinois.edu` (as explained below).

1. (1.1) Let (Σ, E) be an equational theory and G a set of Σ-equations. Prove that if $\mathcal{T}_{\Sigma/E} \models G$, then we have a Σ-algebra isomorphism:

   $$\mathcal{T}_{\Sigma/E} \cong \mathcal{T}_{\Sigma/E \cup G}$$

   Hint: Use initiality!

   (1.2) Let (Σ, E) and Let (Σ, E') be two equational theories. You are asked to answer a very simple question: Suppose $\mathcal{T}_{\Sigma/E} \cong \mathcal{T}_{\Sigma/E'}$. Does this imply that $(\Sigma, E) \equiv (\Sigma, E')$? Give a precise justification for your answer.

   Hints: (i) Recall that the definition of theory equivalence $(\Sigma, E) \equiv (\Sigma, E')$ was given in Homework#7. (ii) Your answer to (1.1) above should give you plenty of food for thought in answering (1.2).

2. We can exploit the result (1.1) above to make inductive theorem proving easier as follows. Suppose that for a functional module, say, `fmod` (Σ, E) `endfm` we have proved that $u = v$ is an inductive theorem. Then of course we have proved that $\mathcal{T}_{\Sigma/E} \models u = v$, and therefore, by (1.1), that $\mathcal{T}_{\Sigma/E} \cong \mathcal{T}_{\Sigma/E \cup \{u=v\}}$. But if $u = v$ (or otherwise $v = u$) has good properties as either a rewrite rule or an equational axiom, we can then *add it to the module* `fmod` (Σ, E) `endfm` and use such an enriched module with the added rule or axiom as a more powerful module in which to prove *other properties*.

   For example, in Lecture 12 we proved *associativity and commutativity* of natural number addition. Why not add them as *axioms* to then prove *distributivity of multiplication*? (such axioms are just what is needed). That is, in the following module, where such axioms have been added and multiplication is defined:

```
fmod NATURAL is
  sort Natural .
  op 0 : -> Natural [ctor] .
  op s : Natural -> Natural [ctor] .
  op _+_ : Natural Natural -> Natural [assoc comm] .
  op _*_ : Natural Natural -> Natural .
  vars N M : Natural .
  eq N + 0 = N .
  eq N + s(M) = s(N + M) .
  eq N * 0 = 0 .
  eq N * s(M) = N + (N * M) .
endfm
```

   use the Maude ITP to prove the distributivity property goal:

```
(goal dist : NATURAL
|- A{N:Natural ; J:Natural ; K:Natural}
((N * (J + K)) = ((N * J) + (N * K))) .)
```

By why not doing the same again! Suppose that we now want to prove that multiplication is *associative*. If you try to do it by hand, you will soon see that you need to use distributivity. But we have already *proved* distributivity! Why not add it as a rule to the above module to get the extended module:

```
fmod NATURAL+DIST is
  sort Natural .
  op 0 : -> Natural [ctor] .
  op s : Natural -> Natural [ctor] .
  op _+_ : Natural Natural -> Natural [assoc comm] .
  op _*_ : Natural Natural -> Natural .
  vars N M J K : Natural .
  eq N + 0 = N .
  eq N + s(M) = s(N + M) .
  eq N * 0 = 0 .
  eq N * s(M) = N + (N * M) .
  eq N * (J + K) = (N * J) + (N * K) .  *** dist added as proved lemma
endfm
```

Use now the Maude ITP to prove in this extended module the associativity of multiplication goal:

```
(goal *-assoc : NATURAL+DIST
|- A{N:Natural ; J:Natural ; K:Natural}
(((N * J) * K) = (N * (J * K))) .)
```

You can just cut and paste the two modules and their corresponding goals from the pdf, but the files for these will also be posted with this homework. You should include a screenshot of your interactions with the ITP and email your code as well to `abir2@illinois.edu`.