**Exercise1**
This Homework took me 15 hours. The programming assignments were straight forward. I am getting used to learning to think formally. So although it was time-consuming this time, I feel like I am improving.
In my spare time, I like to sing and build Internet applications (not together). I try and contribute to community driven open source projects like Sproutcore, Rails, Firefox and more recently Open MRS.

**Exercise3**
*->concatenation*

$$\frac{r_1 \ matches \ s \ leaving \ s'' \quad r_2 \ matches \ s'' \ leaving \ s'}{r_1 r_2 \ matches \ s \ leaving \ s'}$$

*->or*

$$\frac{r_2 \ matches \ s \ leaving \ s'}{r_1|r_2 \ matches \ s \ leaving \ s'} \qquad \frac{r_1 \ matches \ s \ leaving \ s'}{r_1|r_2 \ matches \ s \ leaving \ s'}$$

*->kleene star*

$$\frac{r \ matches \ s \ leaving \ s'' . r^* \ matches \ s'' leaving \ s'}{r^* \ matches \ s \ leaving \ s'} \qquad \frac{}{r^* \ matches \ s \ leaving \ s}$$

**Exercise4**

$$R[[r_1 r_2]](s) = R[[r_1]](s) \cup R[[r_2]](s)$$
$$R[[r_1 r_2]](s) = \bigcup R[[r_2]](x)\{\forall x \ belongsto \ R[[r_1]](s)\}$$

Kleene star is unwound to be

$$R[[r^*]](s) = \bigcup_{x \ belongsto \ R[[r]](s)} R[[r^*]](x)$$

where the context Fx=$R[[r^*]](x)$.

The important difference here is that Fx operates on x that belongs to R[[r]](s) and is bound by the existential quantifier of the existential quantifier $\exists Y \, R[r^*](x) = Y$ Y being a non empty set.

Hence

$$R[r^*](s) = \bigcup_{0<=i<=k} R_i[r^*](s)$$

where

$$R_k[r^*](s) = \bigcup_{x \ belongsto \ R[[r]](s)} R_{k-1}[r^*](x)$$

with the base case being

$R_0[r^*](s) = s$

**Exercise5**

In the given framework and constraints, it is not possible to build deterministic rules for for regexes returning multiple suffices:

1) In order to ensure that the rules are deterministic, we need to ensure that conclusions are never the same for 2 different hypotheses since this may lead to non unique, non-deterministic derivations. Consider the Kleene*

Since r* unconditionally matches any string leaving the entire string as suffix, we can attempt to combine the rules as follows:

$$\frac{r\ matches\ s\ leaving\ S' \quad r^*\ matches\ x\ leaving S\{x|x\ belongsto\ S\}}{r^* matches\ s\ leaving\ (S \cup \{s\})}$$

where {s} is a set containing the entire string (to incorporate the tautology).

If we write a derivation tree for the *regex a* and string "aaaaaaa"*, this rule will put us into **infinite recursion** since the **tautology was removed in an attempt to combine the rules and make the derivation tree deterministic** for any input. This is a case of the rules being incomplete since one of the tautologies (2nd rule for Kleene *) is not seen as a theorem.

2)Consider the concatenation rule. Since there is a constraint that there can be no derivation inside the set-constructor, if we were to remove the derivation and treat x to be picked at random or in the "for all x" manner, we get:

$$\frac{r_1\ matches\ s\ leaving\ S' \quad r_2\ matches\ x\ leaving\ S\{x|x\ belongsto\ S'\}}{r_1 r_2\ matches\ s\ leaving\ S}$$

However this rule that we have stated above assumes that we will always find an 'x' in any set S-prime.

1) Without the qualifiers, picking an x becomes non deterministic in the premises.

2) If we were to make this deterministic (say always pick x to be the nth element by ordering the set), the following problem arises

Consider matching the regex "a*b" onto string "aab" (Should result in a match with single suffix 'b').

S′={aac,ac,c}

if x was either element1 or element2 of the above set, r2 would not match x and we will find no derivation and conclude with a mismatch.

**Hence unless the correct x is picked by chance (or we are allowed to add**

**derivative qualifiers to the set constructor), our regex will not match our string** -- Thus leaving the rule **unsound** in specific cases**.**

---

**Exercise 6**

I choose not to do this problem. You may assume that I am familiar with the relevant literature.