1 3F-1 Bookkeeping

- 0 pts Correct

Answer 3F-1

Rule for $e_1 e_2$

$$\frac{\vdash \mathsf{e_1 \text{ matches } } s \mathsf{\text{ leaving }} s' \quad \vdash \mathsf{e_2 \text{ matches }} s' \mathsf{\text{ leaving }} s''}{\vdash \mathsf{e_1 e_2 \text{ matches }} s \mathsf{\text{ leaving }} s''}$$

Rules for $e_1|e_2$

$$\frac{\vdash \mathsf{e_1 \text{ matches }} s \mathsf{\text{ leaving }} s'}{\vdash \mathsf{e_1|e_2 \text{ matches }} s \mathsf{\text{ leaving }} s'}$$

$$\frac{\vdash \mathsf{e_2 \text{ matches }} s \mathsf{\text{ leaving }} s'}{\vdash \mathsf{e_1|e_2 \text{ matches }} s \mathsf{\text{ leaving }} s'}$$

Rules for $e*$

$$\frac{\vdash \texttt{empty} \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} s}{\vdash \mathsf{e* \text{ matches }} s \mathsf{\text{ leaving }} s}$$

$$\frac{\vdash \mathsf{e \text{ matches }} s \mathsf{\text{ leaving }} s' \quad \vdash \mathsf{e* \text{ matches }} s' \mathsf{\text{ leaving }} s''}{\vdash \mathsf{e* \text{ matches }} s \mathsf{\text{ leaving }} s''}$$

Exercise 3F-2. Regular Expression and Sets [5 points]. We want to update our operational semantics for regular expressions to capture multiple suffices. We want our new operational semantics to be deterministic — it return the set of all possible answers from the single-answer operational semantics above. We introduce a new judgment:

$$\vdash e \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S$$

And use rules of inference like the following:

$$\frac{}{\vdash \text{"x"} \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} \{s' \mid s = \text{"x"} :: s'\}} \qquad \frac{}{\vdash \mathsf{\text{empty matches }} s \mathsf{\text{ leaving }} \{s\}}$$

$$\frac{\vdash e_1 \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S \quad \vdash e_2 \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S'}{\vdash e_1 \mid e_2 \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S \cup S'}$$

You must do one of the following:

- *either* give operational semantics rules of inference for $e*$ and $e_1 e_2$. You may *not* place a derivation inside a set constructor, as in: $\{x \mid \exists y.\ \vdash e \mathsf{\text{ matches }} x \mathsf{\text{ leaving }} y\}$. Each inference rules must have a finite and fixed set of hypotheses.

- *or* argue in one or two sentences that it cannot be done correctly in the given framework. Back up your argument by presenting two attempted but "wrong" rules of inference and show that each one is either unsound or incomplete with respect to our intuitive notion of regular expression matching.

Part of doing research is getting stuck. When you get stuck, you must be able to recognize whether "you are just missing something" or "the problem is actually impossible".

2 3F-2 Regular Expressions, Large Step

- 0 pts Correct

Answer 3F-2
Rule for $e_1 e_2$

$$\frac{\vdash \mathsf{e_1 \ matches} \ s \ \mathsf{leaving} \ S_1 \quad \vdash \mathsf{e_2 \ matches} \ s' \in S_1 \ \mathsf{leaving} \ S_2}{\vdash \mathsf{e_1 e_2 \ matches} \ s \ \mathsf{leaving} \ S_2}$$

Rules for $e*$

$$\frac{\vdash \mathtt{empty} \ \mathsf{matches} \ s \ \mathsf{leaving} \ \{s\}}{\vdash \mathsf{e*} \ \mathsf{matches} \ s \ \mathsf{leaving} \ \{s\}}$$

$$\frac{\vdash \mathsf{e \ matches} \ s \ \mathsf{leaving} \ S_1 \quad \vdash \mathsf{e*} \ \mathsf{matches} \ s' \in S_1 \ \mathsf{leaving} \ S_2}{\vdash \mathsf{e*} \ \mathsf{matches} \ s \ \mathsf{leaving} \ S_2}$$

Exercise 3F-3. Equivalence [7 points]. In the class notes (usually marked as "optional material" for the lecture component of the class but relevant for this question) we defined an equivalence relation $c_1 \sim c_2$ for IMP commands. Computing equivalence turned out to be undecideable: $c \sim c$ iff c halts. We can define a similar equivalence relation for regular expressions: $e_1 \sim e_2$ iff $\forall s \in S. \ \vdash e_1 \ \mathsf{matches} \ s \ \mathsf{leaving} \ S_1 \ \wedge \ \vdash e_2 \ \mathsf{matches} \ s \ \mathsf{leaving} \ S_2 \implies S_1 = S_2$ (note that we are using an "updated" operational semantics that returns the set of all possible matched suffices, as in the previous problem).

You must *either* claim that $e_1 \sim e_2$ is undecideable by reducing it to the halting problem *or* explain in two or three sentences how to compute it. You may assume that I the reader is familiar with the relevant literature.

Answer 3F-3 The equivalence of two regular expressions is decidable. Given two regular expressions e_1 and e_2, one can expand or rewrite them using the abstract grammar that defines regular expressions to see if they can be expressed in the same way. Essentially, you compute the result of the two regular expressions as some combination of elements such as "x", empty, ["x" - "y"], etc, to see if the two are equivalent.

Exercise 3C. SAT Solving. Download the Homework 3 code pack from the course web page. Update the skeletal SMT solver so that it correctly integrates the given DPLL-style CNF SAT solver with the given theory of bounded arithmetic. In particular, you must update only the Main.solve function. Your updated solver must be correct. This notably implies that it must correctly handle all of the included test cases — we use `diff` for some testing, but if you change only the listed method you should end up with the same answers as the reference.

In addition, create an example "tricky" input that can be parsed by our test harness.

Submit your `.ml` and `.input` files.

Exercise 3F-4. SAT Solving [6 points]. Why do the last two included tests take such a comparatively long time? Impress me with your knowledge of DPLL(T) — feel free to use information from the assigned reading or related papers, not just from the lecture slides. I

3 3F-3 Regular Expressions and Sets

- 0 pts Correct

Answer 3F-2

Rule for e_1e_2

$$\frac{\vdash \mathsf{e_1 \text{ matches } } s \mathsf{\text{ leaving }} S_1 \quad \vdash \mathsf{e_2 \text{ matches }} s' \in S_1 \mathsf{\text{ leaving }} S_2}{\vdash \mathsf{e_1e_2 \text{ matches }} s \mathsf{\text{ leaving }} S_2}$$

Rules for $e*$

$$\frac{\vdash \mathtt{empty} \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} \{s\}}{\vdash \mathsf{e* \text{ matches }} s \mathsf{\text{ leaving }} \{s\}}$$

$$\frac{\vdash \mathsf{e \text{ matches }} s \mathsf{\text{ leaving }} S_1 \quad \vdash \mathsf{e* \text{ matches }} s' \in S_1 \mathsf{\text{ leaving }} S_2}{\vdash \mathsf{e* \text{ matches }} s \mathsf{\text{ leaving }} S_2}$$

Exercise 3F-3. Equivalence [7 points]. In the class notes (usually marked as "optional material" for the lecture component of the class but relevant for this question) we defined an equivalence relation $c_1 \sim c_2$ for IMP commands. Computing equivalence turned out to be undecideable: $c \sim c$ iff c halts. We can define a similar equivalence relation for regular expressions: $e_1 \sim e_2$ iff $\forall s \in S.\ \vdash e_1 \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S_1 \ \wedge \ \vdash e_2 \mathsf{\text{ matches }} s \mathsf{\text{ leaving }} S_2 \implies S_1 = S_2$ (note that we are using an "updated" operational semantics that returns the set of all possible matched suffices, as in the previous problem).

You must *either* claim that $e_1 \sim e_2$ is undecideable by reducing it to the halting problem *or* explain in two or three sentences how to compute it. You may assume that I the reader is familiar with the relevant literature.

Answer 3F-3 The equivalence of two regular expressions is decidable. Given two regular expressions e_1 and e_2, one can expand or rewrite them using the abstract grammar that defines regular expressions to see if they can be expressed in the same way. Essentially, you compute the result of the two regular expressions as some combination of elements such as "x", empty, ["x" - "y"], etc, to see if the two are equivalent.

Exercise 3C. SAT Solving. Download the Homework 3 code pack from the course web page. Update the skeletal SMT solver so that it correctly integrates the given DPLL-style CNF SAT solver with the given theory of bounded arithmetic. In particular, you must update only the Main.solve function. Your updated solver must be correct. This notably implies that it must correctly handle all of the included test cases — we use `diff` for some testing, but if you change only the listed method you should end up with the same answers as the reference.

In addition, create an example "tricky" input that can be parsed by our test harness.

Submit your `.ml` and `.input` files.

Exercise 3F-4. SAT Solving [6 points]. Why do the last two included tests take such a comparatively long time? Impress me with your knowledge of DPLL(T) — feel free to use information from the assigned reading or related papers, not just from the lecture slides. I

4 3F-4 Equivalence

- 0 pts Correct

am looking for a reasonably detailed answer. Include a discussion of which single module you would rewrite first to improve performance, as well as how you would change that module.

Potential bonus point: The provided code contains at least one fairly egregious defect. Comment.

Answer 3F-4 The last two test cases take a lot of time because the algorithm as implemented does not efficiently prune the search for integer values that can satisfy the arithmetic clauses. The `Arith.arith` function tries all values for each variable within the bounded range [-127, 128]. Even with just three variables, that could mean that the program has to check well over 16 million possibilities. However, the program could significantly reduce the search space by prematurely terminating some for loops based on information gathered when checking to see if a particular assignment of values to variables creates a valid model.

For example, if we check the assignment of values a = 1, b = 2, and c = 3 and find that the model is not valid because of some condition on a and b, it is a waste of time to modify the value of c and model check again. However, as it is written, the code will continue to try different values of c while keeping the values of a and b the same. This results in the unnecessary enumeration of invalid candidate models.

Submission. Turn in the formal component of the assignment as a single PDF document via the `gradescope` website. Your name and Michigan email address must appear on the first page of your PDF submission but may not appear anywhere else. Turn in the coding component of the assignment via the `autograder.io` website.

5 3F-5 SAT Solving

- 0 pts Correct