1 3F-1 Bookkeeping

- **0 pts** Correct

3F-2.

$\frac{\vdash e_1 \text{ matches } s_1 \text{ leaving } s_2 \quad \vdash e_2 \text{ matches } s_2 \text{ leaving } s_3}{\vdash e_1 e_2 \text{ matches } s_1 \text{ leaving } s_3.}$

$\frac{\vdash e_1 \text{ matches } s \text{ leaving } s_1}{\vdash e_1 | e_2 \text{ matches } s \text{ leaving } s_1.}$

$\frac{\vdash e_2 \text{ matches } s \text{ leaving } s_1.}{\vdash e_1 | e_2. \text{ matches } s. \text{ Leaving } s_1.}$

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

$\frac{\vdash e \text{ matches } s \text{ leaving } s_1 \quad \vdash e* \text{ matches } s_1 \text{ leaving } s_2}{\vdash e* \text{ matches } s \text{ leaving } s_2.}$

2 3F-2 Regular Expressions, Large Step

- **0 pts** Correct

3F-3.

It cannot be done correctly with the given framework. For $e_1e_2$, $e_2$ has to match the string set which is a result of $e_1$'s matching. There is no provided rules for e matching a string set. and we're not allowed to use a derivation inside a set constructor. As a result, we can't deal with the set, using the existing rules.
provided by the evaluation of $e_1$.

I provide 2 wrong attempts here.

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

There is no rule of matching a string set using e, it's basically same as using a derivation inside of a set constructor. We are not allowed to do that.

$$\frac{\vdash e_1 \text{ matches } s \text{ leaving } S1 \quad \vdash e_2 \text{ matches } s \text{ leaving } S_2}{\vdash e_1e_2 \text{ matches } s \text{ leaving } S_1 \cap S_2}$$

It will return incorrect results. because it ignores the consecutive order requirement.

I also provide another wrong attempt about $e^*$ rule

$$\frac{\vdash e \text{ matches } s \text{ leaving } S \quad \vdash e^* \text{ matches } \bigcirc \text{ leaving } S'}{\vdash e^* \text{ matches } s \text{ leaving } S'}$$

Still, I'm not sure about what I can do to use $e^*$ to match a string set. But it's exactly what we need.
represent using

## 3 3F-3 Regular Expressions and Sets

**- 0 pts** Correct

3F.4.

I think $e_1 \cup e_2$ is decideable. We could transfer $e_1$ and $e_2$ to 2 NFA (non-deterministic finite automation). Then we transfer these to 2 NFA to 2 DFA (deterministic finite automation). Finally we check whether these 2 DFA is equal, and it's know that it's decideable to check whether 2 DFA are equal. by comparing whether their minimal-DFAs are the same.

4 3F-4 Equivalence

- **0 pts** Correct

**Exercise 3F-5. 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 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.

The last two test cases contain comparatively more inequality evaluations compared to other test cases. As a result, there is a much larger space for DPLL(T) to search through and figure out whether the input formula are satisfiable or not, which leads to a bad efficiency of DPLL(T). For this code, I would definitely rewrite the arith module first. For now, the code tried out all the candidates of integer values in the constraints on the given range. Instead of such naive linear search, I think we could use binary search to improve its performance. Additionally, there is several well-known integer programming algorithms that we could try for improving this module's performance.

**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