1 3F-1 Bookkeeping

- **0 pts** Correct

3f-2

$$\frac{\vdash e_1 \text{ matches } S_1 \text{ leaving } S_2 \quad \vdash e_2 \text{ matches } S_2 \text{ leaving } S_3}{\vdash e_1 e_2 \text{ matches } S_1 \text{ leaving } S_3}$$

$$\frac{\vdash e_1 \text{ matches } S_1 \text{ leaving } S_2}{\vdash e_1 | e_2 \text{ matches } S_1 \text{ leaving } S_2}$$

$$\frac{\vdash e_2 \text{ matches } S_1 \text{ leaving } S_2}{\vdash e_1 | e_2 \text{ matches } S_1 \text{ leaving } S_2}$$

$$\frac{}{\vdash e_1 * \text{ matches } S \text{ leaving } S}$$

$$\frac{\vdash e \text{ matches } S_1 \text{ leaving } S_2 \quad \vdash e* \text{ matches } S_2 \text{ leaving } S_3}{\vdash e_1 * \text{ matches } S_1 \text{ leaving } S_3}$$

2 3F-2 Regular Expressions, Large Step

- **0 pts** Correct

I select to argue.

Consider the following cases

$$\frac{\vdash e_1 \text{ matches } s \text{ leaving } S \qquad \vdash e_2 \text{ matches } s_1 \text{ leaving } s_1 \;\cdots\; \vdash e_2 \text{ matches } s_i \text{ leaving } s_i}{\vdash e_1 e_2 \text{ matches } s \text{ leaving } \bigcup_{s_i \in S} s_i}$$

Now, suppose for different s, the number of element in S is different.

That means need to let e_2 match with all $s_i \in S$.

Now. it is clear that it is unlikely to write a rule to do that.

$$\frac{\vdash e \text{ matches } s \text{ leaving } S \qquad \vdash e^* \text{ matches } s_i \text{ leaving } \square}{\vdash e^* \text{ matches } s \text{ leaving } \square}$$

no idea what it should be

no idea what it should be

$$\frac{\vdash e \text{ matches } s \text{ leaving } S \qquad \vdash e \text{ matches } s_1 \text{ leaving } s_1 \;\cdots\; \vdash e \text{ matches } s_i \text{ leaving } s_i}{\vdash e^* \text{ matches } s \text{ leaving } \square}$$

no idea what it should be

For both base case and inductive case,

it is impossible.

3 F-3

3 3F-3 Regular Expressions and Sets

- 0 pts Correct

3F-4

① For e_1, e_2, generate 2 NFAs.

② transfer 2 NFAs to DFAs.

③ test the equivalence of the 2 DFAs.

idea: $e_1 \sim e_2$ is decidable if we can reduce a regular expression to a DFA.

Here: NFA: Non-deterministic Finite Automaton

DFA Deterministic Finite Automaton.

3F-5

The reason those tables long time is that DPLL is back-tracking-based searching algorithm, and more inequality in those test cases, the more running time it will take, since it needs more time to do the "back track".

I think arith module could be optimize, since it uses "linear search" or iterate all the candidates in the range (lower bound to upper bound).

so if using some other search algorithms, like binary search, it will be optimized and improved.

4 3F-4 Equivalence

- **0 pts** Correct

3F-4

① For e_1, e_2, generate 2 NFAs.

② transfer 2 NFAs to DFAs.

③ test the equivalence of the 2 DFAs.

idea: $e_1 \sim e_2$ is decidable if we can reduce a regular expression to a DFA.

Here: NFA: Non-deterministic Finite Automaton

DFA Deterministic Finite Automaton.

3F-5

The reason those tables long time is that DPLL is back-tracking-based searching algorithm, and more inequality in those test cases, the more running time it will take, since it needs more time to do the "back track".

I think arith module could be optimize, since it uses "linear search" or iterate all the candidates in the range (lower bound to upper bound).

so if using some other search algorithms, like binary search, it will be optimized and improved.

5 3F-5 SAT Solving

- **0 pts** Correct