오토마타 과제1 – 정규식 NFA 변환 및 문자열 수락 판정

2022-18758 이재현

1. 프로그램 개요

정규 표현식을 NFA로 변환한 후, NFA를 이용하여 특정 입력 문자열이 정규식에 수락되는지 여부를 판별하는 프로그램이다.

Main1.py – 정규식 NFA 변환

Main2.py – NFA 문자열 수락 여부 판별

2. 프로그램 구조

Main1.py는 먼저 정규 표현식을 입력으로 받는다. 이후 정규 표현식을 후위 표기법으로 변환하고, 후위 표기법을 이용해 NFA를 생성한 다음 NFA의 상태와 전이 정보를 출력한다.

Main2.py는 먼저 NFA 정보와, 검사할 문자열을 입력받는다. 이후 NFA를 시뮬레이션하여 수용되면 yes, 아니면 no를 출력한다.

3. 상세 설명

Main1.py의 주요 구현을 상세 설명한다.

Class NFA – NFA 클래스는 시작 상태, 종료 상태, 전이를 인자로 가진다.

Def regex_to_postfix – shunting yard algorithm으로 정규식을 후위표기법으로 변환한다. 스택을 활용하여 연산자 우선순위를 고려해 변환하며, *는 단항 연산자이므로 즉시 출력한다.

Def build_nfa_from_postfix – 후위표기법 식 연산과 동일하게 스택에 연산자(NFA)를 넣어가며 연산(NFA 구성) 한다. NFA를 구성하는 방식, e전이 그리는 방식은 수업에서 배운 것과 동일하게 하였다.

Main2.py의 주요 구현을 상세 설명한다.

Def epsilon_closure - e전이를 통해 도달할 수 있는 상태 계산 함수로, BFS를 사용하여 탐색하며 상태 집합을 리턴한다.

Def simulate_nfa – 초기 상태에 epsilon_closure를 호출하여 e전이를 먼저 계산한다. 그리고 문자열의 각 문자에 대해 반복문을 돌면서 NFA를 시뮬레이션한다. 현재 상태 집합에 대해 가능한 전이를 찾아서 다음 상태 집합에 추가한다. 다음 문자로 넘어가기 전에 e전이를 다시 계산한다. 최종 상태에서 종료 상태가 존재한다면 true를 리턴한다.

다른 부분의 구현들에 대해서는 코드에 있는 주석으로 대체한다.