AI HW3 보고서

전공: 컴퓨터공학과 학년: 4 학년 학번: 20202106 이름: OSHIMA ASUKA

1. 알고리즘 구현 방법

①def standardize_variables(nonstandard_rules):

World 의 rules 가 입력으로 주어지며 해당 룰 내에 모든 변수를 고유한 변수명으로 바꾸는 처리를 한다. 먼저 원래 world 의 변경 없이 작업을 하기위해 인자로 주어진 nonstandard_rules 의 copy 를 만들었다. 변경후의 변수명을 저장하는 리스트 variables 를 만든다. nonstandard_rules 의 각 ID 와 rule 에 대해 아래와 같은 반복처리를 한다. 각 rule 의 antecedents 리스트에 대해 something 또는 someone 를 포함하는 item 인 경우 고유한 변수명을 생성한다. Consequent 에 대해서도 동일하게 something 또는 someone 를 포함하는 item 인 경우 고유한 변수명을 생성한다.

②def unify(query, datum, variables):

하나의 query 와 datum, 그리고 variables(list)가 입력으로 주어진다. query 와 datum 을 unify 하여 각 변수를 고유한 이름으로 치환하며, 치환한 변수를 별도의 dictionary 에 저장한다. query 와 datum 의 길이가 다른 경우 unify 는 불가능하기 때문에 none 를 리턴한다. query 와 datum 의 길이가 같을 때

1. query 와 datum 의 요소를 비교하면서 query 와 datum 가 변수리스트에 존재하는지를 확인한다.
2. query 와 datum 의 양쪽이 변수 인 경우, 같은 assign 이 있는 지 없는 지를 확인하여 같은 assign 이 없는 경우 새로운 대입을 생성하여 기록한다.
3. Unify 가 성공하면 unify 된 리스트와 변수의 assign 을 표현하는 subs 를 리턴한다.

③def apply(rule, goals, variables):

주어진 룰에 대하여 goal 내의 goal 과 unification 이 가능한지를 확인하고 가능한 경우 unify 한 결과를 리스트하여 그에 따른 새로운 newgoals 리스트를 리턴하는 함수다. 먼저 결과를 저장하는 applications 과 goalsets 라는 리스트를 초기화한다. Goals 리스트의 각 목표에 대하여 반복처리를 한다.

1. unify 함루를 이용해서 룰의 결과 (`rule['consequent']`) 과 현재 목표의 unify 를 시도한다.
2. Unify 가 성공하 경우 원래의 룰을 변경하지 않도록 룰을 copy 하여 , 룰의 consequence 과 antecedents 를 업데이트 하고 그것을 applications 리스트의 저장한다.
3. 원래의 목표리스트에서 unify 의 선공한 목표를 삭제하여 unify 가 적용한 룰의 antecedents 를 추가하야 새로운 goal 리스트(goalsets)를 생성한다.

④def backward_chain(query, rules, variables):

Query 의 정당성을 증명할 수 없는 경우 None 를 리턴한다. goal 리스트에 대해 루프 처리한다. goals 에서 목표를 pop 한다. 사용가능한 모든 룰에 대하여 앞서 pop 한 목표에 적용가능한지를 찾는다. 만약 룰이 적용가능한 경우 applications 리스트의

첫번째 룰을 proof 에 추가한다.

2. 결과 화면