자동 문제 생성 시스템

Automatic Problem Generation System

2020.06.17

곽신우 (201411260)
문성찬 (201511260)
배율희 (201511266)
주재빈 (201511298)

작품 설명

- **제공된 데이터를 토대로 객관식 문제를 생성하는 시스템**
  - ✓ 사용자는 시스템에 문제를 생성하고자 하는 도메인에 대해 입력한다
  - ✓ 시스템은 사용자에게 생성된 문제를 제공한다.
- **기능**
  - ✓ 사용자에게 데이터를 제공 받음
  - ✓ 사용자는 데이터를 추가하거나 삭제할 수 있음
  - ✓ 시스템은 데이터를 토대로 문제를 생성하여 사용자에게 반환
  - ✓ 시스템은 사용자가 선택한 답변의 정답 여부를 기록할 수 있음

SW

- 구문 분석 알고리즘
- 문제 생성 모델
- 데이터 입력 인터페이스
- 문제 출력 인터페이스

COST SW

- Pytorch
- konlpy
- Django
- React
- React Native

HW

- Android 기반 스마트폰
- PC

최종 산출물 시안

컴파일러의 역사

최초의 컴파일러

- 최초의 이론적인 컴파일러는 코라도 보헴(Corrado Böhm)이 1951년에 쓴 박사 논문에서 제안되었다.
- 최초로 구현된 컴파일러는 1951년 그레이스 호퍼(Grace Hopper)의 A-0 시스템(Arithmetic Language ver 0)으로, 엄밀히는 로더나 링커에 가까웠다.
- 최초로 상용화된 컴파일러는 1957년 IBM에서 만든 FORTRAN의 컴파일러다.
- 초창기 컴파일러는 시스템 성능이 못받쳐줘서 어셈블러에 비해 덜 선호되었다.

Parsing Algorithm

LL Parser

- 왼쪽에서 오른쪽으로 읽고, 왼쪽부터 트리를 유도하여 이런 이름이 붙었다.
- LL(k) 문법(모호하지 않은 문법(Unambigiuous Grammar)의 일부)를 시간 복잡도 O(N)만에 파싱할 수 있다.
- 좌향재귀(Left Recursive) 문법은 파싱할 수 없다. (ex: E ::= E | T F)
- 1961년 A.A. Grau와 Edgar T.의 ALGOL 컴파일러 구현에 관한 논문에 등장하였고, 같은 해 리처드 웨이쇼프(Richard Waychoff)도 그들과 별개로 구현에 성공했다.
- 재귀 하향식 구현법(Recursive Descent)을 사용한다.
- 구현이 단순하지만, LR 파서에 비해 덜 강력하다.

다음 중 LR Parser에 대한 설명으로 옳은 것을 고르시오.

○LR Parser은/는 1965년 도날드 커누스(Donald Knuth)가 고안했다..
○LR Parser은/는 보통 입력으로 Extended Backus-Naur Form(EBNF) 문법을 받으며, 다양한 프로그래밍 언어로 파서를 출력할 수 있다..
○LR Parser은/는 최초로 상용화된 컴파일러는 1957년 IBM에서 만든 FORTRAN의 컴파일러다..
○LR Parser은/는 모호하지 않은 문법(Unambiguous Grammars)은 시간 복잡도 O(N^2)만에 파싱할 수 있다..

다음이 설명하는 것으로 알맞은 것을 고르시오.

* 1965년 도날드 커누스(Donald Knuth)가 고안했다.
* 왼쪽에서 오른쪽으로 읽고, 오른쪽부터 트리를 유도하여 이런 이름이 붙었다.
* 초창기에는 구현이 너무 복잡하여 실현되지 못했으나, 1969년 Frank DeRemer가 Simple LR(SLR) 기법과 Look-ahead LR(LALR) 기법을 고안하여 현실화하였다.
* 테이블을 사용하는 방법과 재귀 상향식 구현법(Recursive Ascent)이 있다.

○Earley Parser
○LR Parser
○Parser Generator
○LL Parser

Alternative Solutions

- 사람이 문제를 직접 만들어서 사용

- 기존에 존재하는 유사 서비스
  - ✓ 시험지 자동생성 (문제를 시험지에 배치하는 서비스)
    - https://patents.google.com/patent/KR100935913B1/ko
    - https://patents.google.com/patent/KR20100022916A/ko
  - ✓ 수학문제 자동생성 App (일정한 유형의 문제에서 숫자를 변경)
    - https://play.google.com/store/apps/details?hl=ro&id=carrotfarm.codemath.m3

Project Justification

- 소수의 사람이 만든 데이터를 많은 사람이 사용할 수 있다
- 문제를 매번 새롭게 생성하기 때문에 한정된 데이터에서 비교적 많은 결과를 생성할 수 있다
- 실제로 유의미하게 동작하는 문제 생성 시스템이 없다
- 다양한 도메인에 유동적으로 대응하는 시스템이 필요하다
- 다양한 서비스에 접목하여 사용할 수 있다.

Risk Analysis

위험요소	확률 [0,1]	중요도 [0,10]	점수
평문으로 기술된 지식의 구조화 실패	0.9	10	9
문제를 생성하기 위한 알고리즘 구성 실패	0.4	10	4
사용할 프레임워크에 대한 이해 부족	0.4	6	2.5
대면 협업을 하기 어려움	0.2	4	0.8

Risk Reduction Plan

1위	2위	3위
평문으로 기술된 지식의 구조화 실패	**문제를 생성하기 위한 알고리즘 구성 실패**	**사용할 프레임워크에 대한 이해 부족**
평문의 구조화에 실패할 경우, 사용자가 입력하는 방식의 자유도를 어느정도 제한하여 사용자 본인이 구조화된 정보를 입력하도록 유도	여러가지 방식을 동시에 시도하여 가능한 방식을 채택 여러 알고리즘을 사용자가 선택할 수 있게 구성	프로젝트 계획 단계에서 도구에 대한 학습 기간을 미리 확보하여 실제 개발에 들어가기 전에 개발 역량을 확보

Success Criteria

- 지식 데이터 입력
  - ✓ 사람이 이해하기 쉬운 형태의 지식 데이터를 입력할 수 있다
  - ✓ 입력과 편집의 과정이 간단하다
  - ✓ 사용자 자신이 만든 지식 데이터를 저장하고 열람할 수 있다
- 생성된 문제 반환
  - ✓ 지식 데이터를 입력한 사람은 자신의 데이터로 문제를 생성할 수 있다.
  - ✓ 지식 데이터를 입력한 사람은 자신의 데이터로 문제를 생성할 수 있는 사람을 지정할 수 있다. (열람 권한)
  - ✓ 지식 데이터를 입력한 사람은 지정된 사람에게 문제 생성 알림을 보낼 수 있다.

E.O.D

2020.06.17

곽신우 (201411260)
문성찬 (201511260)
배율희 (201511266)
주재빈 (201511298)