지역적 양방향 분석을 이용한 견고한 자연어 파싱 기법

박성완° 나동열
연세대학교 전산학과
{adonis, dyra}@magics.yonsei.ac.kr

A Robust Natural Language Parsing Method Using Local Bi-directional Analysis

Sung-Wan Park° Dong-Yul Ra
Dept. of Computer Science, Yonsei University

요 약

자연어 파싱에 많이 사용되는 Earley 파싱 알고리즘은 입력문장에 에러(error)가 있으면 즉시 종료되기 때문에 견고한(robust) 파싱을 구현하기 어렵다. 본 논문에서 우리는 Earley 파싱 알고리즘을 보다 견고한 파싱 기법으로 만드는 방법을 제안한다. Earley 파싱을 하다가 멈추면 파싱 모드를 지역적 양방향 분석으로 전환시킨다. 에러 위치 다음에 나타나는 단어를 아일랜드(island)로 정한다. 아일랜드를 지역적으로 양방향으로 확장시켜 에러 위치까지 도달하게 한 다음 에러의 종류를 파악하고 이를 복구하는 기법을 사용함으로써 견고성을 얻을 수 있다.

1. 서 론

사람의 언어 처리 메카니즘은 매우 견고하다. 입력 문장에 에러가 있더라도 이를 인식하고 복구하여 계속 분석을 진행한다. 컴파일러와 자연어 처리 시스템처럼 언어처리 소프트웨어에 이런 능력을 추가한다는 것은 중요하다 [3,6].

따라서 문맥자유문법을 파싱하는 동안 에러를 알아내고 복구하는 것에 관하여 컴파일러 분야에서 많은 연구가 있었다 [2]. 그중에서 중요한 것으로 최소 에러 분석(the least-error analysis)라 부르는 기법이 있다 [4]. 여기서의 기본 접근 방법은 입력의 각 단어에서 있을 수 있는 모든 가능한 에러를 고려한 것이다. 입력의 결과로 얻는 여러 파스 중에서 에러의 수가 가장 적은 것을 최종 파스로 채택한다. 입력문의 각 단어 (예로 i 번째 단어 a_i)에 대해 다음의 각 경우를 가정한 분석을 모두 생성하며 진행한다.

(1) a_i의 앞에 예상되는 단어가 삭제된 경우(deletion)
(2) 예상되는 단어 대신 a_i가 들어온 경우(mutation)
(3) a_i가 불필요한 단어로서 추가된 경우(insertion)
(4) a_i가 예상되는 단어 자체인 경우(no error)

최종적으로 얻는 파스 중에서 가정된 에러의 수를 가장 작게 가진 파스를 선택한다. 이 기법은 견고한 자연어파서 개발에 많이 이용되어 왔다 [7].

견고한 파서를 개발하는 데 있어 자연어 분야에서는 대체적으로 상향식 파싱을 이용해 왔다[5]. 상향식 파싱은 에러가 있더라도 계속 진행하여 나갈 수 있는 능력이 있다. 그러나 자연어분석에서 가장 많이 이용되는 알고리즘으로 Earley 파싱 알고리즘이 있다(이는 액티브 차트 파싱(active chart parsing)이라고도 불린다)[1]. 이 알고리즘은 하향식(top-down) 정보를 효과적으로 이용하기 때문에 상향식(bottom-up)접근 보다 매우 효율적이다. 그러나 Earley 파싱의 큰 단점은 견고한 성질을 얻기 어렵다는 것이다. 문장에서 에러가 있으면 그 위치에서 즉시 중지하고 만다.

이를 개선하기 위해서 본 논문에서는 Earley 파싱 알고리즘에게 견고성을 부여하는 기법을 소개한다. 이를 위하여 우리는 Earley 파싱에 "지역적 양방향 분석기법"을 추가한 파싱 기법을 제안한다.

입력문장에 대하여 주(main) 파싱 모드(mode)로 Earley 알고리즘이 수행 중이라고 가정하자. 에러로 인해 파싱이 데드엔드(dead end)가 되면 파싱은 임시적으로 지역적 양방향 분석 모드로 전환된다. 입력문에서 에러가 발생한 위치로부터 얼마간의 다음 단어들(local words)은 양방향 분석 기법으로 처리된다. 그 다음 에러의 종류가 파악되고 수정 복구된다. 그런 후에 파싱은 주 파싱 모드인 Earley 파싱으로 되돌아와 계속 진행한다. 에러가 나중에 또 발생하면 같은 방법으로 처리한다. 우리의 접근 방법은 입력 단어의 삭제, 변형, 추가를 알아내고 복구할 수 있다. 구가 삭제되거나 추가된 경우도 유사한 방법으로 처리 될 수 있다.

2. Earley 파싱 및 에러 탐지

문맥 자유문법 $G=(N, \Sigma, P, S)$라 하자. 입력 스트링 $w = a_1 \ldots a_n$ 이 주어진다고 가정하자.

a_i의 앞은 위치 i-1로 a_i의 뒤는 위치 i로 간주한다. Earley 파싱은 아이템을 만들면서 수행한다. $A\rightarrow\alpha\beta\in P$ 이고 r과 t는 w에서의 위치라하자. 아이템은 $[A\rightarrow\alpha\cdot\beta, r, t]$으로 표시된다. 이 아이템이 생성된다는 사실은 $S\Rightarrow^* a_1\ldots a_r A\delta\Rightarrow a_1\ldots a_r\beta\delta\Rightarrow^* a_1\ldots a_r a_{r+1}\ldots a_t\beta\delta$와 필요충분조건이다. 이 아이템은 r위치에서부터 t위치까지의 입력 부분을 분석했음을 의미한다. 아이템 리스트 L_t는 t까지 분석된 모든 아이템을 포함하는 연결 리스트이다.

Earley 알고리즘은 집합 L_t에 있는 각 아이템마다 가능한 다음 오퍼레이션을 더 이상 가능하지 않을 때까지 적용한다. 오퍼레이션이 적용되어 새로운 아이템이 생성되면 L_t리스트의 마지막에 추가한다.

- **Predict :** 각 $B\rightarrow\eta\in P$에 대하여 아이템 $[A\rightarrow\alpha\cdot B\beta, r, t]$이면, 그리고 집합 L_t에 존재하지 않는다면 아이템$[B\rightarrow\cdot\eta, t, t]$을 L_t에 추가한다.
- **Scan :** 아이템 $[A\rightarrow\alpha\cdot a\beta, r, t]$이 있고 $a=a_{t+1}$이면 아이템 $[A\rightarrow\alpha a\cdot\beta, r, t+1]$을 집합 L_{t+1}에 추가한다.
- **Complete :** $[B\rightarrow\eta\cdot, s, t]$ 과 아이템 $[A\rightarrow\alpha\cdot B\beta, r, s]$이 존재하면 $[A\rightarrow\alpha B\cdot\beta, r, t]$를 집합 L_t에 추가한다.

초기에 모든 $S\rightarrow\alpha\in P$에 대하여 아이템 $[S\rightarrow\cdot\alpha, 0, 0]$을 집합 L_0에 추가한다. 알고리즘은 L_0으로부터 분석을 시작한다. 일반적으로 아이템 집합 L_i ,$0\le i\le n$-1 안의 모든 아이템을 오퍼레이션을 적용하고 더 이상 리스트에 추가가 되지 않으면 끝이 난다. 그러면 그 다음 리스트인 L_{i+1}이 비어 있지 않으면 분석은 다음 리스트 L_{i+1}로 이동하여 분석한다.

만일 L_{i+1}이 비어 있다면, Earley 알고리즘은 멈추게 된다. 이것은 그 위치에서 구문적인 에러가 있다는 것을 의미한다. 특히 이것은 다음 단어 a_{i+1}를 사용하여 scan 오퍼레이션을 적용하려는 모든 시도가 실패했을 때 발생한다. L_n의 모든 아이템을 처리했을 때 입력 문장 w에 대한 파싱의 성공 여부가 결정 된다. L_n의 안에 아이템 $[S\rightarrow\alpha\cdot, 0, n]$이 있다면 파싱이 성공한 것이다.

Earley 알고리즘은 어떤 리스트 L_{i-1}, $1\le i\le n$, 을 분석한 후 중지하면 에러가 발생한 것으로 간주한다. 즉 파서는 단어 a_i를 scan 할 수 없었기 때문에 에러가 난 것이다.

3. 지역적 양방향 분석

단어 a_i를 스캔할 수 없기 때문에 에러가 발생한 상황이라 하자. 단어 a_i의 뒤에 나오는 (임의의) 단어를 초기 아일랜드(island)로서 선택한다. a_j, $i+1\le j\le n$를 아일랜드로 선택하였다 하자. 그다음 이 단어 a_j에서 시작하여 아일랜드를 양방향으로 키워 나아간다(이를 지역적 양방향 분석이라 부르자.) 이 분석에서는 아이템을 $[A\rightarrow\alpha\cdot\gamma\cdot\beta, r, s]$의 형식으로 표시한다. 이 아이템이 생성된 것은 $\gamma\Rightarrow^{\cdot} a_r\ldots a_s$와 필요충분조건이다. 그리고 초기 아일랜드 단어는 a_{r+1}와 a_s사이의 단어이다. 아이템은 생성되면서 어젠다(agenda)에 넣어진다.

분석은 다음과 같이 동작한다:

(1) 아일랜드 단어 a_j를 이용하여 프로젝트 오퍼레이션을 적용하여 생성된 아이템들을 어젠다에 추가한다.
(2) 어젠다에서 맨 앞의 아이템을 꺼낸다.
(3) 꺼낸 아이템에 대해서 모든 가능한 오퍼레이션을 적용하여 아이템을 생성하여 어젠다에 넣는다.
(4) 그다음 어젠다가 비어 있지 않으면 (2)로 가고 그렇지 않으면 지역적 양방향 분석을 종료한다.

어젠다는 큐처럼 운용된다. 오퍼레이션들은 다음과 같다:

(아래에서 I 는 어젠다에서 꺼낸 아이템을 의미한다.)

- **Project (by word) :** 아일랜드 a 에 대해 $A\rightarrow\alpha a\beta\in P$마다 어젠다의 끝에 $[A\rightarrow\alpha\cdot a\cdot\beta, j\text{-}1, j]$를 추가한다.
- **Project (by phrase) :** I= $[B\rightarrow\cdot\eta\cdot, r, s]$에 대해 $A\rightarrow\alpha B\beta\in P$마다 어젠다의 끝에 $[A\rightarrow\alpha\cdot B\cdot\beta, r, s]$를 추가한다.
- **Predict (left) :** I= $[A\rightarrow\alpha B\cdot\gamma\cdot\beta, r, s]$이라면 각 $B\rightarrow\eta\in P$에 대하여 아이템 $[B\rightarrow\eta\cdot\cdot, r, r]$를 추가한다.
- **Predict (right) :** I= $[A\rightarrow\alpha\cdot\gamma\cdot B\beta, r, s]$이라면 각 $B\rightarrow\eta\in P$에 대하여 아이템 $[B\rightarrow\cdot\cdot\eta, s, s]$를 추가한다.
- **Scan (left) :** I= $[A\rightarrow\alpha a\cdot\gamma\cdot\beta, r, s]$이고 $a=a_r$이라면 어젠다에 아이템 $[A\rightarrow\alpha\cdot a\gamma\cdot\beta, r\text{-}1, s]$를 추가한다.
- **Scan (right) :** I= $[A\rightarrow\alpha\cdot\gamma\cdot a\beta, r, s]$이고 $a=a_{s+1}$이라면 어젠다에 아이템 $[A\rightarrow\alpha\cdot\gamma a\cdot\beta, r, s+1]$를 추가한다.
- **Complete (left) :** I= $[B\rightarrow\cdot\eta\cdot, r, s]$이고 아이템 $[A\rightarrow\alpha B\cdot\gamma\cdot\beta, s, t]$가 있으면 어젠다에 아이템 $[A\rightarrow\alpha\cdot B\gamma\cdot\beta, r, t]$를 추가한다.
- **Complete (right) :** I= $[B\rightarrow\cdot\eta\cdot, s, t]$이고 아이템 $[A\rightarrow\alpha\cdot\gamma\cdot B\beta, r, s]$가 있으면 어젠다에 아이템 $[A\rightarrow\alpha\cdot\gamma B\cdot\beta, r, t]$를 추가한다.

꺼내어져 처리된 아이템들은 챠트라 불리는 테이블에 저장된다. 지역적 양방향 분석에서 에러를 유발한 단어였던 a_i는 left scan 작업이 사용할 수 있는 마지막 단어이다. 인덱스가 i 보다 작은 단어들은 Earley 분석으로 처리된 영역의 것이다. (a_1부터 a_{i-1}까지 단어들은 Earley 알고리즘에 의하여 분석된 것들이다.)

$[H_1\rightarrow\alpha_1\cdot a\beta, r_1, i\text{-}1]$와 같은 모든 아이템에 대하여 $a_i\neq a$이므로 Earley 파싱이 멈추게 되었고 그로부터 지역적 양방향 분석이 유발된 사실과 관련하여 다음 사항을 관찰할 수 있다:

"a_i가 a이지 않은 변형 에러가 있다고 가정하자(삭제 및 삽입의 경우도 유사함). a_i를 a로 대치하여 ω 로부터 ω_1을 얻는다 하자. ω_1에 대한 파스트리 T를 생각하자. 리프노드 a에서부터 루트 S까지 경로가 있 다. 이 경로 $<H_0, H_1, \ldots, H_m>$은 에러경로라 하자. $H_0=a$이고 $H_m=S$이다."

파스 트리에서 리프들을 제외한 각 노드들은 그노드와 대응하는 프로덕션을 갖는다. 각 에러 경로에 있는 노드 H_i, $1\le i\le m$, 는 대응 프로덕션으로서 $H_i\rightarrow\alpha_i H_{i-1}\beta_i$를 갖는다고 하자. 이때 Earley 분석이 $[H_i\rightarrow\alpha_i\cdot H_{i-1}\beta_i, r_i, r_{i-1}]$ 한다. $\beta_i\Rightarrow^{\cdot} a_{s_i}\cdots a_{s_{i+1}}$ 이라 할 때 다음이 성립한다.

$r_m \le r_{m-1} \le \ldots \le r_1 \le r_0$ where $r_m = 0$, $r_0 = i-1$
$i-1 \le s_1 \le s_2 \le \ldots \le s_m$ where $s_m = n$

• 에러의 종류의 파악

(1) Deletion error : 아이템 $[H_1 \rightarrow \alpha_1 \cdot H_0\beta_1, r_1, i-1]$의 H_0 즉 a 가 삭제된 경우이다. 즉 α_i의 앞에서 단어 a가 삭제된 경우이다. 그러면 a_i는 β_1을 구성하는데 사용될 것이다. 이 경우라면, 적어도 다음 두 아이템 한 쌍이 존재하여야 한다.

$I_{1,1} = [H_1 \rightarrow \alpha_1 \cdot H_0\beta_1, r_1, i-1]$
$I_{1,2} = [H_1 \rightarrow \alpha_1 H_0 \cdot \beta_1 \cdot, i-1, s_2]$

이 아이템 쌍은 삭제 에러인 경우에서 에러를 찾아내는 최소의 에러의 상황이다. $l > 1$에 대하여 β_l에 대한 스트링 안에 있는 단어가 아일랜드로 선택된다면 지역적 양방향 분석의 결과로 다음 아이템들이 존재하여야 한다.

$I_{2,2} = [H_2 \rightarrow \alpha_2 H_1 \cdot \beta_2 \cdot, s_2, s_3]$
...
$I_{l,2} = [H_l \rightarrow \alpha_l H_{l-1} \cdot \beta_l \cdot, s_l, s_{l+1}]$

이 들 아이템에 대응하는 Earley 분석으로 생성된 아이템은 다음과 같다.

$I_{2,1} = [H_2 \rightarrow \alpha_2 \cdot H_1\beta_2, r_2, r_3]$
...
$I_{l,1} = [H_l \rightarrow \alpha_l \cdot H_{l-1}\beta_l, r_l, r_{l-1}]$

$0 \le r_l \le r_{l-1} \le \ldots \le r_1 \le r_0 = i-1 \le s_1 \le s_2 \le \ldots \le s_l \le n$ 이다.

$I_{1,1}$과 $I_{1,2}$의 존재는 삭제 에러의 존재의 최소의 조건이다. $I_{2,2}, \ldots, I_{l,2}$와 $I_{2,1}, \ldots, I_{l,1}$의 존재는 지역적 양방향 분석의 창을 보다 크게 만들어서 얻어진 추가적인 조건이다. 이 아이템들을 에러의 추가적인 조건(additional context of error)이라 부르자. l을 크게 함에 따라 에러의 추가적인 조건이 많아지게 된다. 에러의 추가적인 조건을 보다 많이 갖고 그것을 조사함에 따라 에러의 타입에 대하여 보다 정확한 결정을 낼 수 있다.

(2) Mutation error : 아이템 $[H_1 \rightarrow \alpha_1 \cdot a\beta_1, r_1, i-1]$가 기대하는 단어 a가 a_i에서 변형된 것이다. 그래서 β_1에 대한 스트링은 a_{i+1}에서 시작하여야 한다. 따라서 최소의 아이템 쌍은 다음과 같다.

$I_{1,1} = [H_1 \rightarrow \alpha_1 \cdot a\beta_1, r_1, i-1]$
$I_{1,2} = [H_1 \rightarrow \alpha_1 a\beta_1 \cdot, i, s_2]$

삭제 에러의 경우처럼 α_i로부터 먼 아일랜드를 선택하여 에러의 추가적인 조건을 얻을 수 있다.

(3) Insertion error : 단어 a_i가 불필요하게 더 들어간 단어인 경우이다. 아이템 $[H_1 \rightarrow \alpha_1 \cdot H_0\beta_1, r_1, i-1]$의 단어 $H_0 = a$가 a_{i+1}와 매치시킨다. 그래서 β_1에 대한 스트링은 a_{i+2}에서 시작하여야 한다. 최소의 아이템 한 쌍은 다음과 같다(단 $H_0 = a = a_{i+1}$).

$I_{1,1} = [H_1 \rightarrow \alpha_1 \cdot H_0\beta_1, r_1, i-1]$
$I_{1,2} = [H_1 \rightarrow \alpha_1 \cdot H_0\beta_1 \cdot, i, s_2]$

위 두 경우와 마찬가지로 추가적인 에러조건은 양방향분석 지역을 크게 할수록 더 많이 얻을 수 있다. 에러의 종류에 대한 판별은 위 (1)-(3) 중의 어느 경우인지를 조사하므로써 가능하다.

4. 에러 복구

전 섹션에서 설명한 에러 조건들로 에러들을 확인한 후 주 파싱 모드, Earley 알고리즘으로 분석으로 되어지도록 변경하기 위해서 에러를 복구할 필요가 있다. β_l에 대한 스트링의 마지막 단어를 a_q라 하자. Earley 분석은 a_{i-1}까지 수행되었다. 그리고 지역적 양방향 분석은 a_i서부터 a_q까지 수행되었다. 그러므로 두 분석 모드를 연결할 필요가 있다. 다음과 같이 새 아이템을 추가한다.

(1) 아이템 $I_{1,2}=[H_1 \rightarrow \alpha_1 H_0 \cdot \beta_1 \cdot, s_1, s_2]$을 이용하여 아이템 $I_{1,1}=[H_1 \rightarrow \alpha_1 \cdot H_0\beta_1, r_1, i-1]$을 확장시킨 아이템 $I_{1,\cdot}=[H_1 \rightarrow \alpha_1 H_0\beta_1 \cdot, r_1, s_2]$을 추가한다.

. . .

(l) $I_{l,\cdot}$ 와 $l_{l,2}=[H_l \rightarrow \alpha_l H_{l-1} \cdot \beta_l \cdot, s_l, s_{l+1}]$을 이용하여 $I_{l,1}=[H_l \rightarrow \alpha_l \cdot H_{l-1}\beta_l, r_l, r_{l-1}]$을 확장시킨 아이템 $I_{l,\cdot}=[H_l \rightarrow \alpha_l H_{l-1}\beta_l \cdot, r_l, s_{l+1}]$을 추가한다.

아이템 $I_{l,\cdot}$는 에러 앞 부분은 Earley 알고리즘에 의해 분석으로 에러 뒤부터 a_q까지는 지역적 양방향 분석이 된 것을 연결한 결과를 나타낸다. 위치 q에 대한 아이템 리스트 L_q 에 대한 처리부터 원래의 주 파싱 모드인 Earley 파싱이 시작된다.

5. 결 론

Earley 파싱 알고리즘은 매우 유용하나 견고성이 전혀 없는 단점이 있다. 견고한 파싱이 요구되는 경우 사람들은 Earley 알고리즘을 이용하는 것을 포기하고 비효율적인 상향식 파싱 알고리즘을 사용하게 된다. 본 논문에서는 이러한 점을 개선하기 위하여 Earley 파싱에 지역적 양방향 파싱을 보태어 견고한 파싱이 가능토록 하였다.

[참고문헌]

[1] A.V. Aho and J.D. Ulman, *The Theory of Parsing, Translation, and compiling, Vol. 1: Parsing,* (Prentice Hanll, Englood Cliffs, NJ, 1972).

[2] S.O. Anderson and R.C. Backhouse, 'Locally least cost error recovery in Earley's algorithm,' *ACM Transactions on Programming Languages and Systems*, Vol. 3, No. 3, (1981), 318-347.

[3] J.G. Carbonell and P.J. Hayes, 'Recovery strategies for parsing extragrammatical language,' *American Journal of Computational Linguistics*, Vol. 9, No. 3-4, (1983), 123-146.

[4] G. Lyon, 'Syntax-directed least errors analysis for context-free languages: a practical approach,' *Commu. ACM*, 17(1), 1974, 3-14.

[5] C.S. Mellish, 'Some chart-based techniques for parsing ill-formed input, Proceedings of ACL,' 1989, pp.102-109.

[6] R.M. Weischedel and N.K. Sondheimer, 'Meta-rules as a basis for processing ill-formed input,' *American Journal of Computational Linguistics*, Vol. 9, No. 3-4, (1983), 161-177.

[7] 이공주, 권철중, 김길창, '문법 범위를 벗어나는 문장 분석을 위한 구문 정보 기반의 견고한 구문 분석기,' 정보과학회 논문지, 23권4호, pp.402-409, 1996.