리덕션 골의 예상: 결정적인 접근 방법
(Prediction of reduction goals : deterministic approach)

이 경 옥 †
(Gyung-Ok Lee)

요 약 LR 파싱 시에 리덕션 골을 리덕션 시점 이전에 찾는 기법은 우문맥 계산 등의 다양한 응용을 갖는다. 기존 연구로서 미리 결정될 수 있는 리덕션 골의 집합을 생성해주는 방식이 제안되었다. 한편 이와 같은 집합 형태의 접근은 비결정적이어서 응용에 따라서는 부적절한 경우가 있다. 이에 본 논문에서는 집합의 형태가 아닌 유일한 예상 가능한 리덕션 골을 제시하는 결정적인 방법을 제안한다.
키워드 : 컴파일러, LR 파싱, 리덕션 골

Abstract The technique of reduction goal prediction in LR parsing has several applications such as the computation of right context. An LR parser generating the set of pre-determined reduction goals was previously suggested. The set approach is nondeterministic, and so it is inappropriate in some applications. This paper suggests a deterministic technique to give a uniquely predictable reduction symbol.
Key words : Compiler, LR parsing, reduction goal

1. 서 론

[1, 2]에서 파서의 생성 시에 미리 결정이 가능한 리덕션(reduction) 골(goal)에 관한 예상 정보를 분석하여 저장함으로 파싱 시간에 이 정보를 이용하는 기법이 제안되었다. 기존의 제안된 방법은 예상 가능한 모든 리덕션 골을 집합 형태로 알려주거나 예상 가능한 골 중의 일부를 파서 생성자가 임의로 결정하게 하였다. 이 방식은 응용에 따라 불필요한 정보 저장을 위한 공간이 낭비될 수 있고, 실제로 필요한 정보를 얻기 위한 작업이 별도로 요구된다. 파서 생성자의 자유로운 선택에 의존하는 경우엔 대상 문법과 생성된 파서에 관한 자세한 이해를 요구한다. 또한 파서 생성을 자동적으로 하는 시스템에서는 비결정적으로 파서 생성자의 결정이 들어감을 구현하기 어렵다.

본 논문에서는 임의의 파싱 시점에서 예상 가능한 골을 유일하게 제시하는 결정적인 방법을 제안한다. 이 방법은 예상 가능한 골에 대한 정보를 놓치지 않는다. 근본 아이디어는 현 파싱 시점으로부터 가장 가까운 골을 선택하는 것이다. 가령 현 시점에서 A와 B에 대한 예상이 가능할 경우에 A에 대한 리덕션이 먼저 일어남을 알 수 있는 경우에 A를 예상 골로 생성한 후 실제로 A에 대한 예상이 마친 후에 다시 B를 예상하는 골로 생성한다. 이 경우에 A와 B에 대한 전체적인 예상 정보를 놓치지 않게 된다. 한편 리덕션 골들간의 이런 순서화의 가능성 여부는 명백하지는 않다. 본 논문에서는 예상 가능한 리덕션 골을 찾기 위한 정형식에 대한 성질을 조사하여 골들이 순서대로 배열될 수 있음을 보인다. 이 순서화에 기반해 가장 가까운 순서로 예상되는 리덕션 골을 생성하는 파서를 제안한다. 이 경우 항상 가까운 순서로 골들이 정렬되기에 예상 가능한 정보들이 빠짐없이 이용될 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 배경 지식을 언급하고, 3장에서는 관련된 정형식의 순서화 가능성을 보인다. 4장에서는 파서 생성 알고리즘을 제안하고, 5장에서는 결론을 맺는다.

· 이 논문은 2003년도 한신대학교 특별연구비 지원에 의하여 연구되었음.
† 종신회원 : 한신대학교 정보통신학과 교수
golee@hanshin.ac.kr
논문접수 : 2002년 5월 2일
심사완료 : 2003년 2월 7일

2. 배경 지식

본 논문에서는 기본적으로 [1, 2, 3, 4]에서의 표기법과

정의를 동일하게 사용한다. 문법 G는 $G=(N, \Sigma, P, S)$로 정의하며 임의의 고정된 자유 문맥 문법을 나타낸다. 그리고 $V=N \cup \Sigma$이다. 또한 G는 특정 규칙 $S' \rightarrow S$(S'는 다른 규칙에는 나타나지 않는 심볼이다)을 포함시킨 확장된 문법임을 가정하며 LR(1) 문법임을 가정한다. $M(G)=(Q, V, GOTO, q_0, \varnothing)$는 G의 LR 기계이다. 여기서 Q는 상태들의 집합이고 $GOTO$는 전이 함수이다. q_0는 시작 상태이고 최종 상태들의 집합은 $\varnothing$이다.

사용자의 편의를 위해서 [1, 2]에서의 d 관계와 Π 관계를 다음과 같이 나타낸다. $A \in N, X \in V, \alpha \in V^*$라고 하자. $Ad^{\alpha}X$가 성립하기 위한 필요 충분 조건은 $A \rightarrow \alpha X \beta \in P$가 성립하는 것이다. d 관계의 재귀적 전이 클로져(reflexive transitive closure)는 d^*에 의해 표현된다. d 관계에 근거한 방향 그래프에서의 경로는 $A_0 d^{\alpha_1} A_1 d^{\alpha_2} A_2 \cdots A_{n-1} d^{\alpha_n} A_n$으로 표현된다. 표기법 $\langle A, \alpha, X \rangle$은 경로들의 집합 $\{h \mid h=A_0 d^{\alpha_1} A_1 d^{\alpha_2} A_2, \cdots A_{n-1} d^{\alpha_n} A_n,\ A_0=A,\ \alpha=\alpha_1 \alpha_2 \cdots \alpha_n,\ A_n=X\}$를 나타낸다.

정의 1.

[1, 2] $A, B \in N$, $\alpha=\beta\gamma$, $Ad^{*^{\alpha}}a, a \in \Sigma$라고 하자.

$(A, \alpha)\Pi_a(B, \gamma)$이기 위한 필요충분조건은 다음과 같다.

즉, 각 경로 $A_0 d^{\alpha_1} A_1 d^{\alpha_2} A_2 \cdots A_{n-1} d^{\alpha_n} A_n$ $\in \langle A, \alpha, a \rangle$, $A_0=A$, $\alpha_1 \alpha_2 \cdots \alpha_n=\alpha$, $A_n=a$에 대해서 $A_l=B, \alpha_{l+1} \cdots \alpha_n=\gamma$인 $l(1 \leq l \leq n)$이 존재한다. □

다음의 성질은 [1, 성질 3]로 부터 얻어진다.

성질 1.

$(A, \alpha)\Pi_a(B, \gamma)$이고 $\alpha=\beta\gamma$라고 하자.

$M(G)$상의 이동

$[\varepsilon] \| x\$ \Rightarrow^* [\varepsilon] \cdots [\eta a] \| y\$ \Rightarrow^* [\varepsilon] \cdots [\eta A] \| z\$ \Rightarrow^* [\varepsilon]\ [S] \| \$$,

$1: y\$=a$는항상 $[\varepsilon] \| x\$ \Rightarrow^* [\varepsilon] \cdots [\eta a] \| y\$ \Rightarrow^* [\varepsilon] \cdots [\eta \beta B] \| u\$ \Rightarrow^* [\varepsilon] \cdots [\eta A] \| z\$ \Rightarrow^* [\varepsilon]\ [S] \| \$$ 형태이다. □

3. Π 관계의 순서화

이 절에서는 Π 관계의 몇가지 성질을 조사하여 Π 관계의 원소들 간의 순서화가 가능함을 보인다. 그리고 가장 가까운 순서로 레벨링을 얻을 수 있는 함수를 정의한다.

보조정리 1.

(1) $(A, \alpha)\Pi_a(A, \alpha)$는 성립하지 않는다.

(2) $(A, \beta\delta\zeta)\Pi_a(B, \delta\zeta), (B, \delta\zeta)\Pi_a(C, \zeta)$라고 하자.
이때 $(A, \beta\delta\zeta)\Pi_a(C, \zeta)$가 성립한다.

(3) $(A, \beta\delta\zeta)\Pi_a(B, \delta\zeta), (A, \beta\delta\zeta)\Pi_a(C, \zeta) \neq (C, \zeta))$ $((B, \delta\zeta)$ 라고 하자.
이때 $\delta \neq \varepsilon$이면 $(B, \delta\zeta)\Pi_a(C, \zeta)$이고, $\delta=\varepsilon$이면 $(B, \zeta)\Pi_a(C, \zeta)$ 또는 $(C, \zeta)\Pi_a(B, \zeta)$가 성립한다.

(4) $(A, \alpha)\Pi_a(B, \gamma), (B, \gamma)\Pi_a(A, \alpha)$ $((A, \alpha) \neq (B, \gamma))$인 A, α, a, B, γ는 존재하지 않는다.

(증명)

(1) $(A, \alpha)\Pi_a(A, \alpha)$가 성립한다고 하자. 그러면 $\langle A, \alpha, a \rangle$에 속하는 $h_0=A_0\ d^{\alpha_1}, A_1 \cdots A_{n-1} d^{\alpha_n} A_n$에 대해서 $A_{l_0}=A, \alpha_{l_0+1} \cdots \alpha_n=\alpha$인 $l_0(1 \leq l_0 \leq n)$가 존재한다.

경로 $h_1=A_{l_0} d^{\alpha_{l_0+1}} A_{l_0+1} \cdots A_{n-1} d^{\alpha_n} A_n$은 $\langle A, \alpha, a \rangle$에 속하기에 $A_{l_1}=A, \alpha_{l_1+1} \cdots \alpha_n=\alpha$인 $l_1(l_0+1 \leq l_1 \leq n)$이 존재한다. 유사하게 $A_{l_2}=\cdots=A_{l_m}=A$, $\alpha_{l_2+1} \cdots \alpha_n= \cdots =\alpha_{l_m+1} \cdots \alpha_n=\alpha$인 $l_2, \cdots, l_m$을 찾을 수 있으며, 일반성을 잃지 않으면서 l_m은 n을 넘지 않는 가장 큰 정수라고 가정할 수 있다. 이때 $h_m=A_{l_m} d^{\alpha_{l_m+1}} A_{l_m+1} \cdots A_{n-1} d^{\alpha_n} A_n$은 $\langle A, \alpha, a \rangle$에 속하고, 따라서 $A_{l_{m+1}}=A, \alpha_{l_{m+1}+1} \cdots \alpha_n=\alpha$인 $l_{m+1}(l_m+1 \leq l_{m+1} \leq n)$이 존재한다.
이것은 l_m의 가정에 대한 모순이다.
따라서 $(A, \alpha)\Pi_a(A, \alpha)$는 성립할 수 없다.

(2) $\langle A, \beta\delta\zeta, a \rangle$에 속하는 임의의 경로 $h=A_0 d^{\alpha_1} A_1 d^{\alpha_2} A_2 \cdots A_{n-1} d^{\alpha_n} A_n$을 생각해보자.
$(A, \beta\delta\zeta)\Pi_a(B, \delta\zeta)$이기에 $A_l=B, \alpha_{l+1} \cdots \alpha_n=\delta\zeta$인 $l(1 \leq l \leq n)$이 존재한다.
또한 $(B, \delta\zeta)\Pi_a(C, \zeta)$이기에 h의 후위 경로 $A_l d^{\alpha_{l+1}} A_{l+1} \cdots A_{n-1} d^{\alpha_n} A_n$에 대해서 $A_i=C, \alpha_{i+1} \cdots \alpha_n=\zeta$인 $i(l+1 \leq i \leq n)$가 존재한다.
따라서 $(A, \beta\delta\zeta)\Pi_a(C, \zeta)$가 성립한다.

(3) $(A, \beta\delta\zeta)\Pi_a(B, \delta\zeta)$이기에 $Ad^{*^{\beta}}B$가 성립한다.
$\delta \neq \varepsilon$라고 하자. 이때,
$\langle B, \delta\zeta, a \rangle$에 속하는 경로 $h=B_0 d^{\delta_1} B_1 \cdots B_{m-1} d^{\delta_m} B_m$을 생각해보자. 그러면

$Ad^{*\beta}B$ 이기에 $\alpha_1 \cdots \alpha_{n-m} = \beta$, $A_{n-m+i} = B_i$, $i = 0, 1, \cdots, m$이고 $\alpha_{n-m+i} = \delta_i$, $i = 1, \cdots, m$인 경로 $h_1 = A_0 d^{\alpha_1} A_1 \cdots A_{n-1} d^{\alpha_{n-1}} A_n$가 $\langle A, \beta\delta\zeta, a\rangle$에 존재한다. 한편,

$(A, \beta\delta\zeta)\Pi_a(C, \zeta)$ 이기에 경로 h_1에 대해 $A_l = C$, $\alpha_{l+1} \cdots \alpha_n = \zeta$인 $l(n-m+1 \le l \le n)$이 존재한다. 따라서 h에 대해서,

$B_{l-(n-m)} = C$, $\delta_{l-(n-m)+1} \cdots \delta_n = \zeta$이다. 그러므로, $(B, \delta\zeta)\Pi_a(C, \zeta)$가 성립한다.

$\delta = \varepsilon$인 경우에 대해서도 유사하게 증명이 가능하다.

(4) $(A, \alpha)\Pi_a(B, \gamma), (B, \gamma)\Pi_a(A, \alpha)((A, \alpha) \neq (B, \gamma))$가 성립한다면 (2)에 의해서 $(A, \alpha)\Pi_a$ (A, α)가 성립함을 의미한다. 한편 이 식은 (1)에 의해서 성립할 수 없다. □

$(A, \alpha)\Pi_a(B, \gamma)$, $(A, \alpha)\Pi_a(C, \delta)$ $((B, \gamma) \neq (C, \delta))$ 라고 하자. 보조정리 1 (3)에 의해서 $(B, \gamma)\Pi_a(C, \delta)$를 가정할 수 있다.

이때 보조정리 1 (4)에 의해서 $(C, \delta)\Pi_a(B, \gamma)$가 성립하지 않는다. 이 성질을 일반화시켜서 다음의 정리를 얻을 수 있다.

정리 1.

$(A, \alpha)\Pi_a(B_1, \gamma_1), (A, \alpha)\Pi_a(B_2, \gamma_2), \ldots, (A, \alpha)\Pi_a(B_n, \gamma_n)$

이 서로 다른 관계들이라고 하자.

각 $(B_1, \gamma_1), (B_2, \gamma_2), \ldots, (B_n, \gamma_n)$에 대해서 $(B_{m_i}, \gamma_{m_i})\Pi_a(B_{m_{i+1}}, \gamma_{m_{i+1}}), i = 1, \ldots, n-1$을 만족하는 $(B_{m_1}, \gamma_{m_1}), (B_{m_2}, \gamma_{m_2}), \ldots, (B_{m_n}, \gamma_{m_n})$의 배열이 유일하게 존재한다. □

다음의 Φ 함수는 정리 1의 유일성에 근거해 정의될 수 있다.

정의 2.

A, α, a가 주어졌다고 하자.

$(A, \alpha)\Pi_a(B_i, \gamma_i)$, $i = 1, \cdots, n$인 (B_1, γ_1), (B_2, γ_2), $\cdots$, (B_n, γ_n)에 대해 다음과 같은 성질에 의해서 Φ 함수가 정의된다.

(i) $(B_{m_i}, \gamma_{m_i})\Pi_a(B_{m_{i+1}}, \gamma_{m_{i+1}}), i = 1, 2, \ldots, n-1$를 만족하는배열 $(B_{m_1}, \gamma_{m_1}), (B_{m_2}, \gamma_{m_2})$, $\cdots$, (B_{m_n}, γ_{m_n})이 존재한다면

$\Phi(A, \alpha, a) = (B_{m_n}, \gamma_{m_n}) \cdots (B_{m_2}, \gamma_{m_2})(B_{m_1}, \gamma_{m_1})$이다.

(ii) 그 밖의 경우엔 $\Phi(A, \alpha, a)$는 정의되지 않는다. □

예제 1.

$G_1 = (\{S, A, C, B, X, Y\}, \{a, b, c\}, P_1, S)$이고

$P_1 = \{S \to A, S \to C, A \to BX, A \to BY, C \to Ba, B \to b, X \to BA, Y \to BC, X \to c\}$라고 하자. 이때,

$(A, BBB)\Pi_b(A, B), (A, BBB)\Pi_b(B, \varepsilon)$이 성립한다. 한편 $(A, B)\Pi_b(B, \varepsilon)$이 성립함을 알 수 있다. 따라서, $\Phi(A, BBB, b) = (B, \varepsilon)(A, B)$이다. □

함수의 정의로부터 다음의 성질들을 얻을 수 있다.

성질 2.

$\Phi(A, \alpha, a) = (B_n, \gamma_n)(B_{n-1}, \gamma_{n-1}) \cdots (B_1, \gamma_1)$라고 하자. 이때 각 $i = 1, \cdots, n$에 대해서 $\gamma_{i-1} = \alpha_i \gamma_i$이다. □

성질 2에 의해서 $\alpha = \alpha_0 \alpha_1 \cdots \alpha_{n-1} \alpha_n$이라고 하면

$\Phi(A, \alpha, a) = (B_n, \alpha_n)(B_{n-1}, \alpha_{n-1}\alpha_n) \cdots (B_1, \alpha_1 \cdots \alpha_{n-1}\alpha_n)$으로 쓸 수 있다.

성질 3.

$\Phi(A, \alpha, a) = (B_n, \alpha_n)(B_{n-1}, \alpha_{n-1}\alpha_n) \cdots (B_1, \alpha_1 \cdots \alpha_{n-1}\alpha_n)$이고 $\alpha = \alpha_0 \alpha_1 \cdots \alpha_{n-1} \alpha_n$라고 하자. 이때 $M(G)$상의 이동

$[\varepsilon]|x\$ \Rightarrow^* [\varepsilon] \cdots [\eta\alpha]|y\$ \Rightarrow^* [\varepsilon] \cdots [\eta A]|z\$ \Rightarrow^* [\varepsilon][S]|\$$, $1{:}y\$ = a$는

$[\varepsilon]|x\$ \Rightarrow^* [\varepsilon] \cdots [\eta\alpha]|y\$ \Rightarrow^* [\varepsilon] \cdots [\eta\alpha_0\alpha_1 \cdots \alpha_{n-1}B_n]|w_n\$$ $\Rightarrow^* [\varepsilon] \cdots [\eta\alpha_0\alpha_1 \cdots \alpha_{n-2}B_{n-1}]|w_{n-1}\$ \Rightarrow^* [\varepsilon] \cdots [\eta\alpha_0 B_1]$ $|w_1\$ \Rightarrow^* [\varepsilon] \cdots [\eta A]|z\$ \Rightarrow^* [\varepsilon][S]|\$$의 형태이다.

(증명)

Φ 함수의 정의에 의해서

$(A, \alpha)\Pi_a(B_i, \alpha_i \cdots \alpha_n), i = 1, \cdots, n$와

$(B_i, \alpha_i \cdots \alpha_n)$ Π_a $(B_{i+1}, \alpha_{i+1} \cdots \alpha_n)$, $i = 1, \cdots, n-1$이 성립한다. 따라서 성질 1을 적용시켜서 성질 3을 얻을 수 있다. □

4. 파서의 생성

이 절에서는 미리 결정될 수 있는 리덕션 골을 가진 LR 파서를 생성하는 알고리즘을 제시한다. 먼저 LR 기계의 상태를 예상 가능한 리덕션 골로 레벨링하는 알고리즘을 제시하며, 기존의 방법에서의 Π 관계를 대신하여 Φ 함수를 사용한다. 한편 [1]에서는 레벨링의 무한과정을 막기 위해서 순환적 경로 집합의 개념이 도입되었고 Π 관계의 부분 관계인 $\Pi^{non-cyclic}$가 비순환적 경로 집합에 대해 정의되었다. 같은 원리로 $\Phi^{non-cyclic}$은 $\langle A, \alpha, a\rangle$이 비순환적 경로 집합인 경우에만 정의되는 Φ의 부분 함수이다.

알고리즘 1.

입력: $M(G)$

출력: $LABEL$ 테이블

방법:

1. $LABEL(q_0)=\{(q_0, S, \varepsilon)\}$

2. **for** 시작 상태가 아닌 모든 LR 상태 q **do** $LABEL(q)=\emptyset$

3. **repeat**

for 각 상태 $q \in Q$ **do**

(a) **for** $(\hat{q}, A, \alpha) \in LABEL(q)$과 $a \in FIRST(q)$ **do**

\- **if** $\Phi^{non-cyclic}(A, \alpha, a)$가 정의된다

then

$$\Phi^{non-cyclic}(A, \alpha, a) = (B_n, \alpha_n)(B_{n-1}, \alpha_{n-1}\alpha_n)\cdots(B_1, \alpha_1 \cdots \alpha_{n-1}\alpha_n),$$

$\widehat{p_{i+1}} = GOTO(\widehat{p_i}, \alpha_i)$, $\widehat{p_0} = q$, $i = 0, 1, \cdots, n-1$라 하자.

(i) **for** 각 $r_i = GOTO(\widehat{p_i}, B_i), i=1, \cdots, n$ **do**

$LABEL(r_i) = LABEL(r_i) \cup \{(\widehat{p_{i-1}}, B_{i-1}, \alpha_{i-1}B_i)\}$

(ii) **if** $[C \to \delta.a\xi, w] \in q$가 존재한다 **then**

$p = GOTO(q, a)$일 때 $LABEL(p) = LABEL(p) \cup \{(\widehat{p_n}, B_n, \alpha_n a)\}$

(iii) **if** $[C \to \delta., a] \in q$가 존재한다 **then**

$p = GOTO(\widehat{p_n}, \rho C), \alpha_n = \rho\delta$일 때

$LABEL(p) = LABEL(p) \cup \{(\widehat{p_n}, B_n, \rho C)\}$

\- **else**

(iv) **if** $[C \to \delta.a\xi, w] \in q$가 존재한다 **then**

$p = GOTO(q, a)$일 때 $\mathrm{LABEL(p)=LABEL(p)} \cup \{(\hat{q}, A, \alpha a)\}$

(v) **if** $[C \to \delta., a] \in q$가 존재한다 **then**

$p = GOTO(\hat{q}, \beta C), \alpha = \beta\delta$일 때

$\mathrm{LABEL(p)=LABEL(p)} \cup \{(\hat{q}, A, \beta C) \mid \beta C \neq A\}$

(b) **if** $\langle A, \alpha', a \rangle$가 순환적 경로 집합이고 $\alpha = \alpha' a$일 때 $(\hat{q}, A, \alpha)$가 $LABEL(q)$에 새로이 더해진다

then $(\hat{q}, A, \alpha)$를 $LABEL(q)$로부터 제거한다.

until $LABEL$ 테이블이 변하지 않는다 □

보조정리 2.

$LABEL(q)$에 $(\hat{q}, A, \alpha)$가 속한다고 하자. 모든 타당한(valid) LR 컨피큐레이션 $[\varepsilon]\cdots[\eta]\cdots[\eta\alpha] \| x\$$에 대해서 $M(G)$상에 $[\varepsilon]\cdots[\eta]\cdots[\eta\alpha] \| x\$ \Rightarrow^{*} [\varepsilon]\cdots[\eta][\eta A] \| y\$$의 움직임이 존재한다.

(증명)

알고리즘 1의 블록 3 (a)의 수행 횟수에 대한 인덕션에 의해서 증명이 가능하며 자세한 과정은 생략한다. □

알고리즘 2는 파싱 시간에 미리 결정이 가능한 리덕션 골을 생성하는 방법이다.

기존의 [1, 2]의 방법에서는 CURRENT_LABEL이 레벨들의 집합으로 주어진데 반해서 알고리즘 2에서는 하나의 레벨이 유일하게 제시된다.

정리 2.

주어진 스택 스트링 $[\varepsilon]\cdots[\theta]$에 대해서 CURRENT_LABEL은 유일하게 결정된다.

(증명)

알고리즘 2

입력: 스택스트링 $[\epsilon]\cdots[\theta]$

출력: $[\epsilon]\cdots[\theta]$에 대한 CURRENT_LABEL

방법:

1. **If** $\theta=\eta\alpha$이고 $(VALID(\eta), A, \alpha)\in LABEL(VALID(\theta))$이다 **then** CURRENT_LABEL은 (A, α)이다.
2. **else**
   (a) CURRENT_LABEL이 정의된 $[\epsilon]\cdots[\theta]$의 가장 긴 전위 스트링(prefix) $[\epsilon]\cdots[\eta]$을 찾아라
   (b) $[\epsilon]\cdots[\eta]$에 대한 CURRENT_LABEL이 (B, β)이고 $\theta=\eta\gamma$인 경우에 CURRENT_LABEL은 $(B, \beta\gamma)$이다. □

시작 컨피규레이션을 $[\epsilon]\|z\$$이라 하고 현재의 컨피규레이션을 $[\epsilon]\cdots[\theta]\|y\$$이라 하자. 이때 본 정리는 유도 과정 $[\epsilon]\|z\$ \Rightarrow^{*} [\epsilon]\cdots[\theta]\|y\$$의 길이에 대한 인덕션으로 증명될 수 있으며 자세한 과정은 생략한다. □

5. 맺는말

본 논문에서는 미리 결정될 수 있는 리덕션 골들의 생성에 관한 결정적인 접근 방법을 제시하였다. 근본 아이디어로는 기존의 비결정적으로 제시된 집합내의 예상 가능 리덕션 골들을 가장 가까운 순서로 배열시키는 것이다. 그 결과 비록 예상 가능 골이 결정적인 방식으로 유일하게 제시되더라도 전체의 예상 정보를 잃지 않는다는 장점을 가진다.

끝으로 제시된 방법에 대한 응용과 그 효과에 대해서 논하고자 한다. [1, 2]에서 미리 결정된 리덕션 골의 응용으로 우문맥 (right context) [5] 계산의 적용과 오류 보정 시에 활용되는 가능한 (feasible) 리덕션 골 [4, 6]로의 적용이 논의된 바 있다. 이들 응용에 본 논문에서 제시한 가장 가까운 리덕션 골의 선택을 적용 시에 각 응용의 취지에 적합한 효율적인 적용이 가능하다. 먼저 우문맥 계산 시에는 가장 가까운 리덕션 심볼을 기준으로 한 인수 분해는 공통 요소에 대한 최대한의 인수 분해를 수행하기에 그만큼의 효율적인 계산을 수행할 수 있다. 또 다른 응용의 예인 가능한 리덕션 골로서 가장 가까운 리덕션 골을 선택한 경우에도 [4]에서의 가장 짧은 오류 컨피규레이션의 선택의 취지와 일치하여 효율적인 구(phrase) 단위 오류 복구를 가능하게 한다.

참 고 문 헌

[1] 이경옥, 최광무, "미리 결정된 리덕션 심볼들을 가진 LR 파서", 정보과학회 논문지, 제26권, 7호, pp. 931-937, 1999.

[2] Lee, Gyung-Ok and Choe, Kwang-Moo, "An LR parser with pre-determined reduction goals", Information Processing Letters, Vol 72, pp. 189-196, 1999.

[3] Aho, A. V. and Ullman, J. D., The Theory of Parsing, Translation and Compiling, vols.1 2. p. 1002, Englewood Cliffs, NJ:Prentice-Hall 1972, 1973.

[4] Sippu, S. and Soisalon-Soininen, E., Parsing Theory, vol I, II, p. 228, p.426, Springer-Verlag Berlin Heidelberg, 1990.

[5] Jung, Min-Soo, Choe, Kwang-Moo, and Han, Taisook, "An Efficient Computation of Right Context for LR-based Error Repair", Information Processing Letters, Vol 49, No 2, pp. 63-72, 1994.

[6] Sippu, S. and Soisalon-Soininen, E., "A Syntax Error-Handing Technique and Its Experimental Analysis", ACM Trans. on Progra. Lang. and Sys., Vol 5, No 4, pp. 656-679, 1983.

이 경 옥
1990년 2월 서강대학교 전자계산학과 졸업(학사). 1992년 8월 한국과학기술원 전산학과 졸업(석사). 2000년 2월 한국과학기술원 전산학과 졸업(박사). 현재 한신대학교 정보통신학과 조교수. 관심분야는 프로그래밍 언어론과 컴파일러 구성론 등