ELSEVIER

Information Processing Letters 51 (1994) 85–92

Information Processing Letters

# Grammar coverings of a deterministic parser with action conflicts

Eun-Jung Lee *, Kwang-Moo Choe

*Department of Computer Science, Korea Advanced Institute of Science and Technology, 373-1, Kusung-dong, Yusung-ku, Taejon 305-701, South Korea*

Communicated by K. Ikeda; received 14 June 1993; revised 24 February 1994

**Abstract**

Extended LR parsing of ambiguous grammars is investigated in view of parser transformations. We define a cover grammar of a transformed parser in terms of grammar coverings, and we present a generation method of cover grammars, together with a proof. The method reflects a parser transformation into a cover grammar, which shows the effect of deleting actions in the parser. A general cover grammar for a transformed parser is introduced by the proposed method, and for a meaningful class of parsers, the generation of a reduced cover grammar is presented.

*Keywords:* Formal languages; Parser generating systems; Ambiguous grammars; Grammar coverings

## 1. Introduction

A useful method to parsing ambiguous grammars has been proposed by Aho and Johnson [1] to extend LR and LL parsing. This scheme allows small-sized, efficient parsers with natural grammar description, so it is accepted in almost all parser generating systems [4,5]. Although the wide acceptance in parser generating systems, deviated from the original way, there has been rather little theoretical research on this scheme. Ruzica [10] has shown that a transformation of a parser to resolving conflicts may change the language set of the original one, and it is indeed undecidable to know whether a language set is changed or not (deciding "disambiguatability"), because it is equivalent to the problem deciding whether the grammar is ambiguous or not. Moreover, he mentioned that there exists an equivalent grammar to the transformed parser. On the other hand, there are several approaches applying the scheme to reduce parsers [3,8].

In this paper, motivated to propose a theoretical model to the above scheme, we define *cover grammars of a parser* in terms of a grammar covering and a homomorphism from a parser to its

* Corresponding author. Email: {ejlee,choe}@plhae.kaist.ac.kr.

$$\begin{tikzcd}
G & \overline{G} \arrow[r, Leftrightarrow] \arrow[d, Rightarrow, "\text{deleting rules}"'] & M \arrow[d, Rightarrow, "\text{deleting actions}"] \\
& G' \arrow[r, Leftrightarrow] & M'
\end{tikzcd}$$

$\overline{G}$: a right-cover of $G$
$G'$ : subgrammar of $\overline{G}$
$M'$ : a transformed parser of $M$
$\Longleftrightarrow$: the proposed cover relation

Fig. 1. Cover of a transformed parser.

underlying grammar [11]. And the general method to construct a cover grammar is presented. Fig. 1 outlines our method. Given grammar $G$ and parsers $M$, $M'$ as in Fig. 1, our method begins with a cover grammar $\overline{G}$ which not only covers $G$ but also describes $M$, and finds a subgrammar of $\overline{G}$ which covers the transformed parser ($G'$ in the figure); deleting rules from $\overline{G}$ to $G'$ reflects deleting actions from $M$ to $M'$. We present a condition on $\overline{G}$ to decide whether there exists a subgrammar of $\overline{G}$ which covers a given transformed parser. This approach provides a useful model to help understanding the effect of transformations.

In the next section, notation and definitions are visited for later uses. Section 3 defines cover grammars of a parser and presents the generation method together with the correctness proof. Section 4 introduces two types of cover grammars in terms of machine description grammars, which illustrate that our method is useful. Section 5 ends this paper with a summary and discussing applications of the model.

## 2. Notation and definitions

In this section, we review several notation and definitions that will be used in this paper. We tried to use notation and terminology consistent with [11].

A *context-free grammar* (CFG) is a quadruple $G = (N, \Sigma, P, S)$. For a sentence $x$ of $G$, $\pi$ is a parse of $x$ if there exists a derivation sequence $S \Rightarrow^{\pi} x$ in $G$. A grammar $G$ is said to be *ambiguous* if there exists a sentence $w$ such that $S \Rightarrow^{\pi_1} w$ and $S \Rightarrow^{\pi_2} w$ where $\pi_1 \neq \pi_2$. And for a subset of productions $R \subset P$, a subgrammar $G - R = (N, \Sigma, P - R, S)$ is a useful grammar of $G$ after deleting productions in $R$.

Let $M(G)$ be an LR-based parser of $G$, $M(G) = (C, \Gamma, Goto, q_s)$, where $C$ is a set of states, $\Gamma$ a set of actions, and $q_s$ the start state in $C$; these components are defined for each parsing method, SLR($k$), LALR($k$) and LR($k$) as described in [11]. We denote an action $t \in \Gamma$ as a triple of $(C \times \{shift, reduce\, r\} \times \Sigma^k)$, where a lookahead of length $k$ is assumed. We write $[\gamma]$ to denote the accessible state of $M(G)$ with the viable prefix $\gamma$ of $G$. A homomorphism $\tau$ from $\Gamma$ to $P$ [11] is defined as follows:

$$\tau(q, reduce\, A \to \alpha, a) = A \to \alpha, \qquad \tau(q, shift, a) = \varepsilon.$$

Then a parser move of $M(G)$ is defined as:

(1) $[\varepsilon] \cdots [\gamma] | ay \vdash^{t} [\varepsilon] \cdots [\gamma][\gamma a] | y$ if $t = ([\gamma], shift, a) \in \Gamma$;
(2) $[\varepsilon] \cdots [\gamma] \cdots [\gamma\alpha] | uy \vdash^{t} [\varepsilon] \cdots [\gamma][\gamma A] | uy$ if $t = ([\gamma\alpha], reduce\, r, u) \in \Gamma$.

The sentence $x$ is accepted by $M(G)$ if there exists an action sequence $\rho$ in $M(G)$ such that $[\varepsilon] | x \vdash^{\rho} [\varepsilon][S] | \varepsilon$, and we say that $\rho$ is a *valid action sequence*.

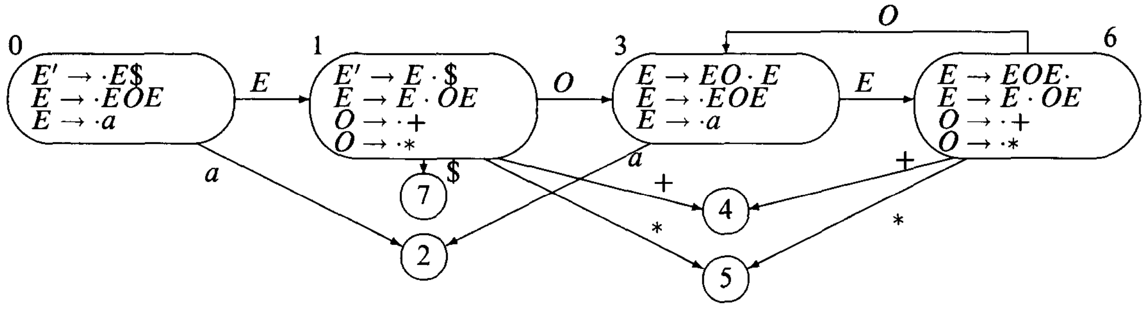

Fig. 2. LR(0) automaton of $G_{\text{exp}}$.

A parser is said to be *deterministic* if there is only one or zero element in the action set for any given state and symbol pair. A deterministic parser for an ambiguous grammar may be constructed by selecting a winner among several conflicting actions for a given state and next symbol [1], which is referred in this paper as a *parser transformation.* Let $\Delta$ be a subset of the action set $\Gamma$ in $M(G)$, then $M(G) - \Delta$ denotes a parser deleting actions in $\Delta$ from $M(G)$. Then *the transformed parser* by $\Delta$ is written as $M(G) - \Delta = (C, \Gamma - \Delta, Goto, q_s)$.

For example, consider an ambiguous grammar $G_{\text{exp}} : E \to EOE \mid a,\ O \to + \mid *$. The LR(0) automaton is shown in Fig. 2; there are shift–reduce conflicts on the symbols $+, *$ in state 6 for the LALR(1) parser. This grammar will be used as a running example in this paper. In the transformed parser, actions are inconsistent to the other features, *Goto* and states. For the example, when the action in $\Delta = \{(6, \mathit{shift}, +), (6, \mathit{shift}, *)\}$ are deleted, *Goto* from the state 6 by $O$ is invalidated.

Let $G_1 = (N_1, \Sigma_1, P_1, S_1)$, and $G_2 = (N_2, \Sigma_2, P_2, S_2)$ be two CFGs. Let $h$ be a homomorphism from $(N_1 \cup \Sigma_1)$ to $(N_2 \cup \Sigma_2)$ and extended to $P_1 \to P_2$ such that

$$h(A \to \alpha) = \begin{cases} h(A) \to h(\alpha) & \text{if } h(A) \neq h(\alpha), \\ \varepsilon & \text{otherwise.} \end{cases}$$

We say that there exists a *homomorphism* $h$ from $P_1$ to (onto) $P_2$ if $h(P_1) \subset P_2$ ($h(P_1) = P_2$). In this paper, $\Sigma_1 \supseteq \Sigma_2$ is assumed, since we are interested in two grammars with the same language set. And $G_1$ is *a right-cover* of $G_2$ with respect to $h$ if the following conditions are satisfied:

(1) if there is a derivation $S_1 \Rightarrow^{\pi_1} x$ in $G_1$, then there is a derivation $S_2 \Rightarrow^{h(\pi_1)} x$ in $G_2$;

(2) if there is a derivation $S_2 \Rightarrow^{\pi_2} x$ in $G_2$, then there is a derivation $S_1 \Rightarrow^{\pi_1} x$ in $G_1$ and $h(\pi_1) = \pi_2$.

## 3. Grammar coverings of a transformed parser

In this section, a cover grammar of a parser is defined in terms of a grammar covering, together with inspecting properties of cover grammars. We also present a formal method to generate a cover grammar of a transformed parser. In the literature, a cover grammar has been informally referred as an *equivalent grammar* of a parser [10]. First, we define a cover grammar of a parser. Note that a parser in what follows means an LR-based parser including a transformed one.

**Definition 1.** Let $M(G) - \Delta$ be a parser of $G$ deleting actions in $\Delta$ from $M(G)$, and $\tau$ a homomorphism from actions of $M(G)$ to $P$ as defined in Section 2. Then a grammar $\hat{G} = (\hat{N}, \hat{\Sigma}, \hat{P}, \hat{S})$ is a *cover* of $M(G) - \Delta$ with respect to $h$ and $\tau$ if the following conditions are satisfied:

(1) there exists a homomorphism $h$ from $\hat{P}$ to $P$;
(2) there exists a parse $\hat{\pi}$ in $\hat{G}$ such that $\hat{S} \Rightarrow^{\hat{\pi}} x$ if and only if there exists an action sequence $\rho$ in $M(G) - \Delta$ such that $[\varepsilon] \mid x \vdash^{\rho} [\varepsilon][S] \mid \varepsilon$ and $\tau(\rho) = h(\hat{\pi})^R$.

It is easy to show that a right-cover of $G$ covers $M(G)$, i.e., $\Delta = \emptyset$ in the above definition. Our goal is, for a given cover grammar $\overline{G}$ of $M(G)$, to find a set of rules $\overline{R} \subset \overline{P}$ such that a subgrammar $\overline{G} - \overline{R}$ covers $M(G) - \Delta$, reflecting the transformation from $M(G)$ to $M(G) - \Delta$. The set $\overline{R}$ corresponds to the action set $\Delta$. In order to find the correspondence, we define a *locating set* of an action as a set of rules in $\overline{G}$ which might locate at the position in derivations corresponding to the given action with respect to $h : \overline{G} \to G$ and $\tau$. For a cover grammar $\overline{G}$, following function $T : \Gamma \to 2^{\overline{P}}$ defines a locating set of an action:

$$T(q, reduce\, r, u) = \{\overline{A} \to \overline{\alpha} \mid \overline{S} \Rightarrow^* \overline{\mu A y} \Rightarrow \overline{\mu \alpha} y \text{ in } \overline{G},\ [h(\overline{\mu\alpha})] = q,$$
$$h(\overline{A} \to \overline{\alpha}) = \tau(q, reduce\, r, u),\ First_k(y) = u\},$$
$$T(q, shift, u) = \{\overline{X} \to \overline{\gamma} a \overline{\delta} \mid \overline{S} \Rightarrow^* \overline{\mu X} y \Rightarrow \overline{\mu\gamma} a \overline{\delta} y \text{ in } \overline{G},\ [h(\overline{\mu\gamma})] = q,$$
$$u = av,\ v \in First_{k-1}(\overline{\delta} y)\}.$$

The function $T$ is extended to the set of actions such that $T(\emptyset) = \emptyset$, $T(\Delta_1 \cup \Delta_2) = T(\Delta_1) \cup T(\Delta_2)$. Note that the function $T$ can be determined for any grammar $\overline{G}$ which right-covers $G$ on a homomorphism $h$. If two locating sets for $\Delta$ and $\Gamma - \Delta$ are disjoint to each other, then we can get the correspondence between action sequences in $M(G) - \Delta$ including any of $\Delta$ and useless derivation sequences in $\overline{G} - T(\Delta)$.

**Lemma 2.** *Let $\overline{G}$ be a cover grammar of $M(G)$ with respect to $h$ and $\tau$, $\overline{r}$ a rule in $\overline{G}$, and $\Delta$ a set of actions in $M(G)$. Let $\overline{\pi}$ and $\rho$ be a parse in $\overline{G}$ and a valid action sequence in $M(G)$ respectively, where $h(\overline{\pi}) = \tau(\rho)^R$. Moreover, assume that $T(\Delta) \cap T(\Gamma - \Delta) = \emptyset$. Then the following are satisfied:*
(1) *If there is an action $t \in \Delta$ such that $\rho = \rho_1 t \rho_2$, then there is a rule $\overline{r} \in T(\Delta)$ such that $\overline{\pi} = \overline{\pi}_1 \overline{r} \overline{\pi}_2$.*
(2) *If there is a rule $\overline{r} \in T(\Delta)$ such that $\overline{\pi} = \overline{\pi}_1 \overline{r} \overline{\pi}_2$, then there is an action $t \in \Delta$ and $\rho = \rho_1 t \rho_2$.*

**Proof.** The first statement comes from the definition of the locating set since if $\tau(\rho_1 t \rho_2) = h(\overline{\pi})^R$, then we can find a rule $\overline{r} \in T(t)$ such that $\overline{\pi} = \overline{\pi_1 r \pi_2}$. We now consider the second one. Let $\overline{r} = \overline{A} \to \overline{\alpha}$, $h(\overline{r}) = A \to \alpha$. Then there is a derivation sequence $\overline{S} \Rightarrow^{\overline{\pi}_1} \overline{\mu A} y \Rightarrow^{\overline{r}} \overline{\mu\alpha} y \Rightarrow^{\overline{\pi}_2} xy$ in $\overline{G}$ for some $\overline{\mu} \in (\overline{N} \cup \Sigma)^*$, and also in $G$:

$$S \Rightarrow^{h(\overline{\pi}_1)} h(\overline{\mu A})y \Rightarrow^{h(\overline{r})} h(\overline{\mu\alpha})y \Rightarrow^{h(\overline{\pi}_2)} xy$$

by the homomorphism $h$ from $\overline{G}$ to $G$. Moreover, the corresponding parser move in $M(G)$ with respect to $\tau$ is

$$[\varepsilon] \mid xy \vdash^{\rho_1} [h(\overline{\mu})] \cdots [h(\overline{\mu\alpha})] \mid y \vdash^{t} [h(\overline{\mu})][h(\overline{\mu A})] \mid y \vdash^{\rho_2} [\varepsilon][h(\overline{S})] \mid \varepsilon$$

where $\rho = \rho_1 t \rho_2$, $\tau(\rho) = h(\overline{\pi}_1 \overline{r} \overline{\pi}_2)^R$. So, $\overline{r} \in T(\Delta)$ and $T(\Delta) \cap T(\Gamma - \Delta) = \emptyset$ implies the action $t$ is in $\Delta$. □

This result enables us to find the set of rules in $\overline{G}$ corresponding to the deleted actions of $M(G) - \Delta$. We end this section with the theorem, a main result of this paper, which shows that $\overline{G} - T(\Delta)$ covers $M(G) - \Delta$ using preliminaries prepared up to now.

**Theorem 3.** *Let $\overline{G}$ be a cover grammar of $M(G)$ with respect to $h$ and $\tau$, and $M(G) - \Delta$ be a transformed parser of $M(G)$ by deleting actions in $\Delta$. Let $T$ be a function on actions of $M(G)$ to the locating sets of $\overline{G}$ as defined above. If $T(\Delta) \cap T(\Gamma - \Delta) = \emptyset$, then $\overline{G} - T(\Delta)$ covers $M(G) - \Delta$.*

**Proof.** Since $\overline{G}$ is a cover grammar of $M(G)$, it is enough to consider the deleted action sequences of $M(G) - \Delta$ and the derivation sequences of $\overline{G}$ which become useless after deleting rules in $T(\Delta)$. So, we have to show that a parse $\overline{\pi}$ in $\overline{G}$ contains a rule in $T(\Delta)$ if and only if there exists an action $t \in \Delta$ such that $\tau(\rho_1 t \rho_2) = h(\overline{\pi})^R$ for some $\rho_1, \rho_2$ which is the result of Lemma 2. Therefore, we can conclude that the grammar $\overline{G} - T(\Delta)$ covers $M(G) - \Delta$. □

We say that a cover grammar of $G$ *describes* $M(G) - \Delta$ if we can find a subgrammar which covers $M(G) - \Delta$ using the locating set. It enables us to find an adjusted cover grammar of a given parser as the choice of $\overline{G}$.

## 4. Several types of cover grammars

This section illustrates our method by using it to generate several cover grammars in terms of well-known grammars of the literature, which are often called machine description grammars (*mdg* in short). An mdg is a grammar annotated with parser information, such as first, follow symbols and state; this model has been frequently used in the literature [3,6,7]. We consider two types of cover grammars $\overline{G}_0$ and $\overline{G}_1$, *an action mdg* and *a head-free mdg*, respectively; the former can describe general parser transformations, whereas the later raises a useful class on parser transformations. We present several observations on these cover grammars. The action mdg has the form $\langle u, q, X, v\rangle$. Ruzica [10] stated the existence of a cover grammar using such an mdg. We will show that the method of Section 3 generates the same result.

**Definition 4.** Let $M(G) = (C_0, \Gamma, Goto, q_s)$ be a LALR(1) parser of a grammar $G$. Then *the action mdg* of $M(G)$ is $\overline{G}_0 = (\overline{N}_0, \Sigma, \overline{P}_0, \overline{S}_0)$, where $\Sigma$ is the set of terminal symbols of $G$, and $\overline{N}_0, \overline{P}_0$ are as follows:

(1) $\overline{N}_0 = \{\overline{S}_0\} \cup \{\langle u, q, X, v\rangle \mid q \in C_0,\ u \in First_k(Xv),\ v \in Follow(q, X),\ X \in N \cup \Sigma\}$,

(2) $$\begin{aligned}\overline{P}_0 = {} & \{\overline{S}_0 \to \langle u, q_s, S, \$\rangle \mid u \in First_k(S\$)\}\\ & \cup \{\langle u, q, a, v\rangle \to a \mid u = First_k(av),\ q \in C_0\}\\ & \cup \{\langle u, p_0, A, v\rangle \to \langle u_0, p_0, X_1, v_0\rangle\langle u_1, p_1, X_2, v_1\rangle \cdots \langle u_{n-1}, p_{n-1}, X_n, v_{n-1}\rangle \mid\\ & \quad u = u_0,\ v = v_{n-1},\ v_{i-1} = u_i,\ p_i = Goto(p_{i-1}, X_i),\\ & \quad 1 \leqslant i \leqslant n,\ A \to X_1 \cdots X_n \in P\}.\end{aligned}$$

In the symbol $\langle u, p, X, v\rangle$, the first component $u$ is called *the head part* and contained in the set $First_k(Xv)$. We consider several properties of $\overline{G}_0$ for later uses.

**Fact 5.** *There exists a homomorphism $h_0$ from $\overline{G}_0$ onto $G$ where $h_0(\langle a,\ q, X, b\rangle) = X$, and $\overline{G}_0$ right-covers $G$ on $h_0$.*

**Fact 6.** *If $\overline{S}_0 \Rightarrow^* \overline{\mu}\langle u, q, A, v\rangle y$ in $\overline{G}_0$, then $[h_0(\overline{\mu})] = q$ and $First_k(y) = v$.*

Computing the locating sets of $\overline{G}_0$ are straightforward from the above two properties such that

$$\begin{aligned}
&T_0(q, \mathit{shift}, a) = \{\langle u,q,a,v\rangle \to a \in \overline{P}_0\},\\
&T_0(q, \mathit{reduce}\,A \to X_1 \cdots X_n, v)\\
&\quad = \{\langle u_0,p_0,A,v_{n-1}\rangle \to \langle u_0,p_0,X_1,v_0\rangle \cdots \langle u_{n-1},p_{n-1},X_n,v_{n-1}\rangle) \in \overline{P}_0 \mid\\
&\qquad q = \mathit{Goto}(p_{n-1},X_n),\ v_{n-1} = v\}.
\end{aligned}$$

As the function $T_0$ shows, each rule in $\overline{G}_0$ corresponds only one action, and represents all information of a corresponding reduce action. Moreover, $T_0(\Delta_1) \cap T_0(\Delta_2) = \emptyset$ if and only if $\Delta_1 \cap \Delta_2 = \emptyset$. By Theorem 3, we can conclude that there exists a subgrammar of $\overline{G}_0$ which covers $M(G) - \Delta$ for any $\Delta$, since $T_0(\Delta) \cap T_0(\Gamma - \Delta) = \emptyset$.

Consider the example of $G_{\mathrm{exp}}$ when $\Delta = \{(6, \mathit{shift}, +), (6, \mathit{reduce}\,E \to EOE, *)\}$. The action mdg of $M(G_{\mathrm{exp}})$ has the following set of rules:

$$\begin{array}{lllll}
\overline{G}_{\mathrm{exp0}}: & a0E\$ \to a0E{+}\ {+}1Oa\ a3E\$ & a0E\$ \to a0E{*}\ {*}1Oa\ a3E\$ & a0E\$ \to a0a\$ & \\
& a0E{+} \to a0E{+}\ {+}1Oa\ a3E{+} & a0E{+} \to a0E{*}\ {*}1Oa\ a3E{+} & a0E{+} \to a0a{+} & \\
& \underline{a0E{*} \to a0E{+}\ {+}1Oa\ a3E{*}} & \underline{a0E{*} \to a0E{*}\ {*}1Oa\ a3E{*}} & a0E{*} \to a0a{*} & \\
& a3E\$ \to a3E{+}\ {+}6Oa\ a3E\$ & a3E\$ \to a3E{*}\ {*}6Oa\ a3E\$ & a3E\$ \to a3a\$ & \\
& a3E{+} \to a3E{+}\ {+}6Oa\ a3E{+} & a3E{+} \to a3E{*}\ {*}6Oa\ a3E{+} & a3E{+} \to a3a{+} & \\
& \underline{a3E{*} \to a3E{+}\ {+}6Oa\ a3E{*}} & \underline{a3E{*} \to a3E{*}\ {*}6Oa\ a3E{*}} & a3E{*} \to a3a{*} & \\
& {+}1Oa \to {+}1{+}a & {*}1Oa \to {*}1{*}a & {+}6Oa \to {+}6{+}a & {*}6Oa \to {*}6{*}a \\
& {+}1{+}a \to {+} & {*}1{*}a \to {*} & \underline{{+}6{+}a \to {+}} & {*}6{*}a \to {*} \\
& a0a\$ \to a & a0a{+} \to a & a0a{*} \to a & \\
& a3a\$ \to a & a3a{+} \to a & a3a{*} \to a &
\end{array}$$

where the underlined rules are contained in $T_0(\Delta)$. Hence, $\overline{G}_{\mathrm{exp0}} - T_0(\Delta)$ covers $M(G) - \Delta$.

Although $\overline{G}_0$ describes all parser transformations, we will introduce a new mdg, namely *head-free*, which is simpler than $\overline{G}_0$ and preserves the structure of the original grammar. The head-free mdg omits the head part of the symbol in $\overline{G}_0$.

**Definition 7.** Let $G, M(G)$ be the same as Definition 4. The *head-free mdg* is $\overline{G}_1 = (\overline{N}_1, \Sigma, \overline{P}_1, \overline{S}_1)$, where $\Sigma$ is the set of terminal symbols of $G$, and $\overline{N}_1$ and $\overline{P}_1$ are defined as follows:

(1) $\overline{N}_1 = \{S_1\} \cup \{\langle q,X,u\rangle \mid q \in Q,\ u \in \mathit{Follow}(q,X)\}$

(2) $$\begin{aligned}\overline{P}_1 = {} & \{\overline{S}_1 \to \langle q_s, S, \$\rangle\}\\
& \cup \{\langle q,a,u\rangle \to a \mid \langle q,a,u\rangle \in \overline{N}_1\}\\
& \cup \{\langle p_0,A,v_{n-1}\rangle \to \langle p_0,X_1,u_0\rangle \cdots \langle p_{n-1},X_n,u_{n-1}\rangle \mid\\
& \qquad p_i = \mathit{Goto}(p_{i-1},X_i),\ 1 \leqslant i \leqslant n,\ A \to X_1 \cdots X_n \in P\}.\end{aligned}$$

There exists a homomorphism $h_1$ from $\overline{G}_1$ onto $G$ such that $h_1(\langle q,X,u\rangle) = X$. And the locating set of $\overline{G}_1$ is defined in the same manner as $T_0$ as follows:

$$\begin{aligned}
&T_1(q, \mathit{shift}, a) = \{\langle q,a,u\rangle \to a \in \overline{P}_1 \mid v \in \mathit{First}_{k-1}(u)\},\\
&T_1(q, \mathit{reduce}\,A \to X_1 \cdots X_n, v) = \{\langle p_0,A,u\rangle \to \langle p_0,X_1,u_0\rangle \cdots \langle p_{n-1},X_n,u_{n-1}\rangle \in \overline{P}_1 \mid\\
&\qquad\qquad q \in \mathit{Goto}(p_{n-1},X_n),\ v \in \mathit{Follow}_{k,\overline{G}_1}(\langle p_0,A,u\rangle)\}.
\end{aligned}$$

**Theorem 8.** *Let $\overline{G}_1$ be a head-free mdg of $M(G)$, and $\Delta$ a set of actions deleted from $M(G)$. Then $\overline{G}_1$ describes $M(G)-\Delta$ if there are no two lookaheads $u,v \in \Sigma^k$ such that $(q, \mathit{reduce}\, A \to \alpha, u) \in \Delta$, $(q, \mathit{reduce}\, A \to \alpha, v) \notin \Delta$ where $q = \mathit{Goto}(p,\alpha)$ and $v \in \mathit{Follow}_{k,\overline{G}_1}(\langle p,A,u\rangle)$.*

**Proof.** We have to show that there is no rule $\overline{r} \in \overline{G}_1$ such that $\overline{r} \in T(\Delta) \cap T(\Gamma - \Delta)$ with the above assumptions. If $\overline{r} \in T_1(t_1) \cap T_1(t_2)$, $t_1 \neq t_2$, then $t_1, t_2$ are reduce actions. Assume that $\overline{r} = \langle p,A,u\rangle \to \overline{\alpha}$, and $(q, \mathit{reduce}\, A \to \alpha, u) \in \Delta$ without loss of generality. It is enough to show that there is no action $t' \in \Gamma - \Delta$ such that $\overline{r} \in T_1(t')$. If $\overline{r} \in T_1(t')$, then $t' = (q, \mathit{reduce}\, A \to \alpha, w)$ for some $w \neq u$ and $w \in \mathit{Follow}(\langle p,A,u\rangle)$, a contradiction to the assumption. □

For an operator grammar, i.e., there is no two adjacent nonterminals in the right-hand sides of rules, $\mathit{Follow}_{k,\overline{G}_1}(\langle p,A,u\rangle) = \{u\}$ and consequently, any transformed parser of $G$ can be described by the head-free mdg. All of practical parsers for programming languages, excepting C++ are described by head-free mdgs. Moreover, it is also remarkable that the cover grammar obtained by a head-free mdg allows a natural reduction as the following lemma shows.

**Lemma 9.** *Let $\hat{G}_1$ be a cover of $M(G)-\Delta$ which is obtained by the above method with head-free mdg, $\overline{G}_1$, and let $\hat{h}_1$ be a homomorphism on $\overline{G}_1$ as follows:*

$$\hat{h}_1(\langle p,X,a\rangle) = \begin{cases} \langle p,X,a\rangle & \text{if there is a rule } \overline{r} \in T_1(\Delta) \text{ such that } \langle p,X,a\rangle \Rightarrow^{\pi_1 r \pi_2} x \text{ in } \overline{G}_1, \\ X & \text{otherwise.} \end{cases}$$

*Then $\hat{h}_1(\hat{G}_1)$ covers $M(G)-\Delta$.*

**Proof.** Let $G' = \hat{h}_1(\hat{G}_1)$. Because $G'$ is a subgrammar of $\hat{G}_1$, there exists a homomorphism $h_1$ from $\hat{G}_1$ to $G$. Then, we can get a homomorphism $h'$ from $G'$ to $G$ by means of two homomorphisms $h_1$ and $\hat{h}_1$ such that $h'(X) = X, h'(\langle p,X,u\rangle) = X$. Moreover, the second condition of Definition 1 is satisfied since $\hat{G}_1$ right-covers $G'$ with respect to $\hat{h}_1$ and $\hat{G}_1$ covers $M(G)-\Delta$. □

With the example grammar of $G_{\text{exp}}$, consider the case when $\Delta = \{(6, \mathit{reduce}\, E \to EOE, +), (6, \mathit{reduce}\, E \to EOE, *)\}$. Then the condition of Theorem 8 is satisfied, and the head-free mdg $\overline{G}_{\text{exp1}}$ has the rules as follows:

$$\begin{array}{lll}
\overline{G}_{\text{exp1}}: & 0E\$ \to 0E{+}\ 1Oa\ 3E\$ \mid 0E{*}\ 1Oa\ 3E\$ \mid 0a\$ & 3E\$ \to 3E{+}\ 6Oa\ 3E\$ \mid 3E{*}\ 6Oa\ 3E\$ \mid 3a\$ \\
& 0E{+} \to \underline{0E{+}\ 1Oa\ 3E{+} \mid 0E{*}\ 1Oa\ 3E{+}} \mid 0a{+} & 3E{+} \to \underline{3E{+}\ 6Oa\ 3E{+} \mid 3E{*}\ 6Oa\ 3E{+}} \mid 3a{+} \\
& 0E{*} \to \underline{0E{+}\ 1Oa\ 3E{*} \mid 0E{*}\ 1Oa\ 3E{*}} \mid 0a{*} & 3E{*} \to \underline{3E{+}\ 6Oa\ 3E{*} \mid 3E{*}\ 6Oa\ 3E{*}} \mid 3a{*} \\
& 1Oa \to 1{+}a \mid 1{*}a & 6Oa \to 6{+}a \mid 6{*}a \\
& 0a\$ \to a \quad 0a{+} \to a \quad 0a{*} \to a \quad 3a\$ \to a & 3a{+} \to a \quad 3a{*} \to a \\
& 1{+}a \to + \quad 1{*}a \to a \quad 6{+}a \to + \quad 6{*}a \to a &
\end{array}$$

where $0E\$$ is the start symbol of $\overline{G}_{\text{exp1}}$ and the underlined rules are contained in $T(\Delta)$. In this example, since both of two lookaheads $+$ and $*$ originated from the same rule $E \to EOE$ are deleted, the head part of symbol $O$ need not be specified. The reduced cover grammar by Lemma 9 is

$$\begin{array}{lllll}
E \to 0E{+}\ O\ E \mid a & E \to 0E{*}\ O\ E \mid a & 0E{+} \to a & 0E{*} \to a & \\
E \to 2E{+}\ O\ E \mid a & E \to 2E{*}\ O\ E \mid a & 2E{+} \to a & 2E{*} \to a & O \to + \mid *
\end{array}$$

We are currently interested in further reductions in order to find a minimal cover. In this example, $0E+$, $0E*$, $2E+$, and $2E*$ can be intuitively united to $E2$ such that

$$E \rightarrow E2OE \mid a \quad E2 \rightarrow a \quad O \rightarrow + \mid *.$$

## 5. Conclusion

Motivated to approach formally to an extended parsing of an ambiguous grammar, we have defined a cover grammar of a transformed parser in terms of grammar coverings. We have presented a generation method of cover grammars from the cover of the underlying grammar. In addition, we visited two types of cover grammars for a transformed parser in order to illustrate the proposed method. And we visited a reduction on the resulting grammar.

Our approach allows further classifications on parser transformations depending on the describing grammars. On the other hand, recent report on writing a C++ grammar [9] states that the resolution scheme is useful to write such a complex syntax. But there is no helpful systems to guide resolving conflicts and deleting actions. Our approach can be applied to provide a model to show the effect of deleted actions or other information to help the user to debug grammars.

**Acknowledgement**

The authors wish to thank the referees for their helpful comments.

## References

[1] A.V. Aho, S.C. Johnson and J.D. Ullman, Deterministic parsing of ambiguous grammars, *Comm. ACM* **18** (1975) 441–452.
[2] F.L. DeRemer and T.J. Pennello, Efficient computation of LALR(1) lookahead sets, *ACM Trans. Programming Language Systems* **4** (1982) 415–649.
[3] A.J. Demers, Skeletal LR parsing, in: *Proc. 15th Ann. IEEE Symp. on Switching and Automata Theory* (1974) 185–198.
[4] S.C. Johnson, YACC: Yet another compiler-compiler, Computer Science Tech. Rept. Nr. 32, Bell Laboratories, Murray Hill, NJ, 1975.
[5] K. Koskimies, O. Nurmi, J. Paakki and S. Sippu, The design of a language processor generator, *Software – Practice and Experience* **18** (1988) 107–135.
[6] M.-J. Lee and K.-M. Choe, A SLR($k$) covering for LR($k$) grammars, *Inform. Process. Lett.* **37** (1991) 337–347.
[7] M.D. Mickunas, On the complete covering problem for LR($k$) grammars, *J. ACM* **23** (1976) 17–30.
[8] D.J. Rosenkrantz and H.B. Hunt, Efficient algorithms for automatic construction and compactification of parsing grammars, *ACM Trans. Programming Language Systems* **9** (1987) 543–566.
[9] J.A. Roskind, A YACC-able C++ 2.0 grammar, Release 1.1, 1990.
[10] D. Ruzica, *Local Disambiguatable Transformations*, Lecture Notes on Computer Science **32** (Springer, Berlin, 1975).
[11] S. Sippu and E. Soisalen-Soininen, *Parsing Theory* (Springer, Berlin, 1990).