# Schema Method: A Framework for Correcting Grammatically Ill-formed Input

Ikuo KUDO1) , Hideya KOSHINO2) , Moonkyung CHUNG2) and Tsuyosi MORIMOTO1)

1)*ATR Interpreting Telephony Research Laboratories*
*Twin 21 Building MID Tower*
*2-1-61 Shiromi, Higashi-ku*
*Osaka 540, Japan*

2)*CSK Research Institute*
*3-22-17 Higashi-Ikebukuro, Toshima-ku*
*Tokyo 170, Japan*

## Abstract

The schema method is a framework for correcting grammatically ill-formed input. In a natural language processing system ill-formed input cannot be overlooked. A computer assisted instruction (CAI) system, in particular, needs to show the user's errors. This framework diagnoses ill-formed input, corrects it and explains the error, if an input is ill-formed. The framework recognizes a sentence at two steps: first parses weak grammar, and then strongly filters the parsed sentence. When it is known what sentences are passed by the filter, it can be used even if it is imperfect. As the strong filter, a new method is used: an interpretation schema and an interpretation rule. An interpretation schema collects input information schemata and then an interpretation rule judges whether the collected schemata are correct or incorrect. This approach overcomes the problem of relaxation control, the major drawback of the previous syntactically-oriented methods, and is also more efficient.

## 1. Introduction

Ill-formed input cannot be ignored when a natural language processing system such as a computer assisted instruction (CAI) system or a machine translation system is built. Particularly in a CAI System, students often make mistakes, such as mispunctuation, lack of agreement, misplaced/improperly-used words, etc. In these cases, a CAI system needs to point out input errors, and show why the input is wrong. In order to do so, the system needs to diagnose and correct ill-formed input to explain the errors. The schema method as a framework for correcting grammatically ill-formed input is suggested and the diagnosis and correction of errors is discussed.

There have been many studies for processing ill-formed input for English. The point of those studies is the diagnosis: how does the system find an error? The approaches are classified into two groups: the syntactically-oriented group and the frame-based group.

The syntactically-oriented group includes robust parsers based on Augmented Transition Networks (ATN) which use the relaxation technique /Kwansny 1981/ or the meta-rule /Weischedel 1980, 82, 87/, and the EPISTLE system which addresses the problems of the checking grammar and style of texts, such as letters, reports and manuals, written in ordinary English /Heidorn 1982/, /Jensen 1983/.

The frame-based group attempts to deal with ungrammatical input through extensions to pattern matching parsing /Hayes 1981/, through conceptual case frame instantiation /Schank 1980/ and through approaches involving multiple cooperating parsing strategies /Carbonell 1983/. The target of that study is dialogue phenomena in communication with limited-domain systems, such as data-base systems, electronic mail systems, etc.

The aim of this study is error-correction of non-native speakers written English text. This approach is syntactically oriented.

The syntactically-oriented approaches /Kwansny 1981/ /Weischedel 1980,82,87/, /Heidorn 1982/, /Jensen 1983/ are very similar. Their basic idea is relaxation. They first attempt to parse the input, using fully grammatical rules. If the sentence is not parsed, some of the conditions are relaxed. However these approaches have two major drawback.

(1)Relaxation control strategies: when inputs are ill-formed, some means of ranking alternatives is appropriate. The number of relaxed configurations may be large.

> *One of the most critical problems is control. The need to relax the very rules that constrain the search for an interpretation is like opening Pandora's box. /Weischedel 1987 (PP.117)/*

(2)Computational inefficiency: the relaxation approach cannot recognize ill-formed input before the analysis with well-formed grammar is finished. Furthermore, fully well-formed grammar is needed. To make fully well-formed grammar, subcategorization of parts of speech is needed and other conditions are added. As a result, there are too many rules.

In comparison to previous approaches, this approach does not use the relaxation technique. The difference between previous approaches and this one is the method of recognizing an ill-formed sentence. Previous approaches first use a strong filter, then relax the conditions. This approach, however, first uses weak grammars, and then strongly filters the passed sentence. This approach recognizes a sentence at two steps.

An attempt is made to expand lexical-functional grammar (LFG) /Kaplan 1982/ to deal with ill-formed input. LFG has two drawbacks: (1) LFG can't deal with errors of omission and (2) LFG has no framework for error correction. If an input sentence is well-formed, this framework obtains an LFG f-structure. If not, the sentence is corrected.

Examples of error correction are given in the next section. In the section following the basic idea is described and the problem of a unification mechanism for processing ill-formed input is discussed. This framework is shown in section 4.

## 2. Non-native speaker's ill-formed phenomena

In this section, treated examples of non-native speaker's ill-formed phenomena are given. The application is a CAI system for Japanese junior high school students in a primary English course. Their errors are different from a native speaker's. Typical errors are shown in Table 1.

English is very different from Japanese in parts of speech, word-order, tense, etc. For a Japanese, there is no concept of (1) countable and uncountable nouns ① ② ③ in Table 1, (2) singular and plural forms ③ (3) articles ④ ⑤ (4) agree-ment between subject and verb ⑥ (5) adverb word-order ⑦.

Japanese interfered with the students' acquision of English. The following errors are often made by Japanese adults as well. (4)verb style ⑧ (5) category mistakes, word misuse ⑨. Furthermore, junior high school students are reading and hearing a foreign language (English) for the first time, and thus have no concept of foreign language whatsoever. (6) Logical error ⑩: the student who made the mistake explained that "are+not → aren't", "is+not → isn't" so " am+ not→ amn't". (7) Primary students are not familiar with English grammar and can't distinguish between "Who" or "Where" ⑪ ⑫. (8)Surface error: letter or punctuation problems ⑬

Table 1. Examples of errors by junior high school students

| | |
|---|---|
| ①*He plays piano.<br>He plays the piano. | ②*He play the baseball.<br>He plays baseball. |
| ③*some good advices<br>some good advice | ④*I am student.<br>I am a student. |
| ⑤*A moon is smaller than an erath.<br>The moon is smaller than the earth. | |
| ⑥*He is one of those men who is difficult to please.<br>He is one of those men who are difficult to please. | |
| ⑦*I have finished my homework already.<br>I have already finished my homework . | |
| ⑧*He is listening music on the radio now.<br>He is listening to music on the radio now. | |
| ⑨*We cannot play baseball in here.<br>We cannot play baseball here. | |
| ⑩*Yes, I amn't.<br>Yes, I am not. Yes, I'm not. | |
| ⑪*Who does cook breakfast?<br>Who cooks breakfast? | ⑫*Where they live?<br>Where do they live? |
| ⑬*Does mr. brown have a book<br>Does Mr. Brown have a book? | |
| ⑭*We must stop to complain.<br>We must stop complaining. | |

Grammatical errors ①~ ⑬ are treated, but not semantic errors ⑭ and absolutely ill-formed sentences which are not comprehensible. The aim is to diagnose grammatical errors and show a reason for the error. For example:

Input sentence; Mr Brown has a pen,
correction; Mr. Brown has a pen.
the reason; A period is needed after " Mr".
The comma after "pen" should be a period.

## 3. Basic idea

In this section, the basic idea of the framework and the problem of the LFG unification mechanism in dealing with ill-formed input is described.

### 3.1 Two -level filter

The framework uses two-level filters for input sentence classification: a well-formed sentence, a relatively ill-formed sentence or an absolutely ill-formed sentence as shown in Figure 1.

(1)First an attempt to parse the input, using normal context-free grammar (Filter I ) is made. Both a well-formed sentence and the relatively ill-formed sentence which includes feature errors are passed through the filter (Filter I ).

(2)Secondly, these inputs are checked with a strong filter (Filter II ). A well-formed sentence passes, but a relatively ill-formed sentence does not.

(3)An input which is not passed through the first filter (Filter I ), includes word-order or omitted-word errors, or unnecessary words ⑪ ⑫. The input is classified by a filter (III), called Improper Grammar, as relatively ill-formed or absolutely ill-formed .

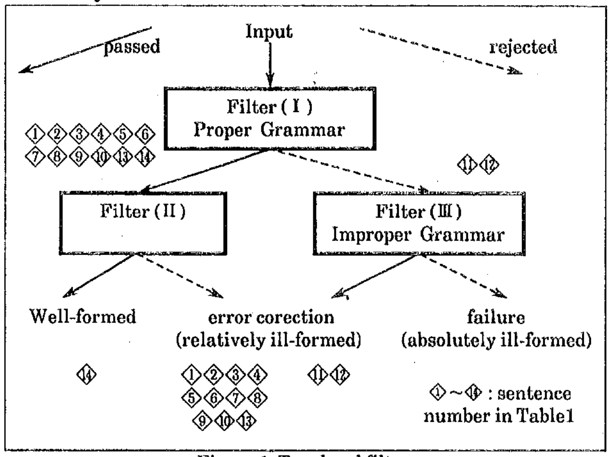


Figure 1 Two-level filter

### 3.2 Filter test

Filter ( I ) is a context-free grammar. This filter is a weak filter. Therefore some relatively ill-formed inputs are passed. Consider how many sentences are derived from the grammar rules in Figure 2. 25 (5×1×5) sentences are generated by the grammar rules and dictionary entries. Of course, not only well-formed sentences as in (1) below, but also ill-formed sentences as in (2), (3), (4) below, are included.

| Grammar rules | | Dictionary | |
|---|---|---|---|
| S→NP VP | : Verbal Phrase (VP) | pronoun | →this |
| VP→verb NP | | verb | →is |
| NP→pronoun | : Noun Phrase (NP) | det | →an |
| NP→det noun | | noun | →apple |
| NP→noun | | noun | →apples |

The generated sentences
(1)This is an apple. (2)This is apple.
(3)This is an apples. (4)This is apples.

Figure 2 The generated sentences

### 3.3 The problem of the LFG unification mechanism for ill-formed input

Relatively ill-formed sentences, as well as feature errors, pass through Filter( I ). Filter(II) must work as a strong grammatical filter. LFG contains such a strong filter, called the unification mechanism, "from F-Descriptions to F-Structures /Kaplan 1982 (pp.203)/". For example,

"This is *a apple*"

In LFG a disagreement, "*a apple*", is rejected because the following equations are not unified.

$$\begin{cases} (\uparrow \text{SPEC}) = \text{a} & \text{from a} \\ (\uparrow \text{SPEC}) = \text{an} & \text{from apple} \end{cases}$$

However, for diagnosis and error-correction there are some drawbacks in LFG framework :

(1)LFG can't check an error of omission as in the noun phrase '*∅ apple*' in the sentence

"This is *apple*".

As the sentence *lacks the article "an"*, there is no determiner equation and the unification mechanism does not work. Thus the sentence is recognized as a well-formed sentence.

$$\begin{cases} \varnothing & \text{from } \varnothing \quad \text{:lack of article} \\ (\uparrow \text{SPEC}) = \text{an} & \text{from apple} \end{cases}$$

(2)LFG has no error-correction framework. It only rejects the ill-formed input. Addition of an error-correction mechanism is thus necessary.

## 3.4 Improper Grammar [Filter (III)]

In this application, users are non-native speakers unfamiliar with English grammar. Thus, a user often makes word-order errors, includes unnecessary words, or leaves out words ⑦ ⑧. A teacher could show why "does" is not necessary in the sentence ⑦ "*Who does cook breakfast?", or why "do" is needed in ⑧ "*Where they live?". If a system diagnoses such sentences, it needs to provide the grammar rules for analysis. The type of error shown in Figure 3 is called improper grammar.

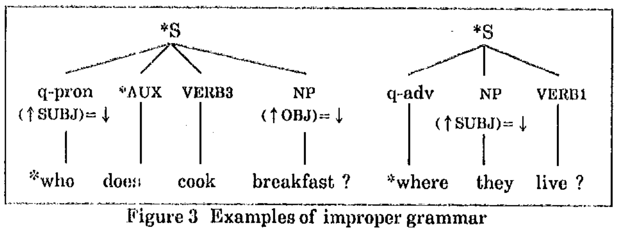


Figure 3 Examples of improper grammar

## 4. The framework

In this section an overview of the framework is explained. Unification approach has some drawbacks for diagnosis as we described in 3.3. A new method is used as a filter (II). The idea is to compare input style with proper surface styles which are synthesized from lexical and grammatical conditions. An interpretation schema collects the conditions (surface schema and LFG schema) and an interpretation rule synthesizes proper styles and judges whether the sentence is ill- or well-formed as shown in Figure 4. In this section, at first, new schemata are notated: surface schema (4.1), surface constraint (4.2), interpretation schema, interpretation schema with condition, conditional schema and kill schema (4.3). And then the instantiation mechanism and interpretation of

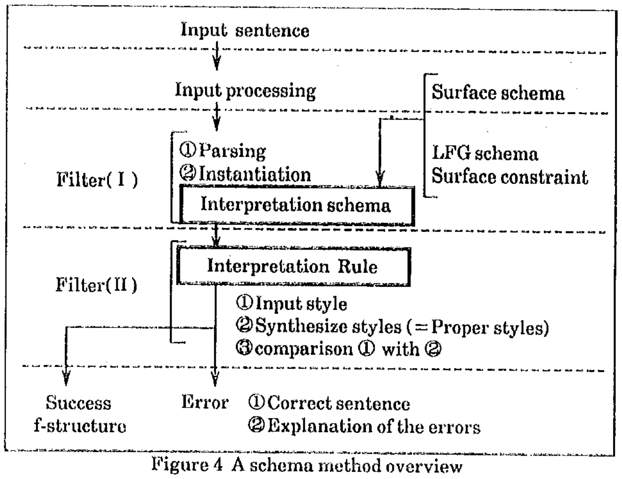


Figure 4 A schema method overview

new schemata are described (4.4) (4.5). Finally error-correction is illustrated (4.6).

## 4.1 Input processing
## [Surface schema]

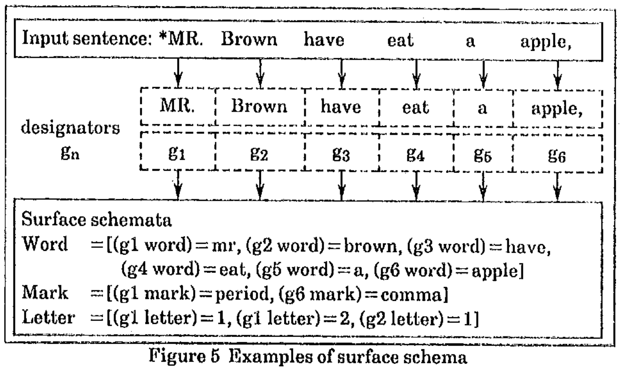
A capital letter and a punctuation indicate surface of an input sentence. In this framework such information is represented as a schema, called a surface schema. In the input processing, the input sentence is converted into surface schemata. The schema is notated as follows.

$(g_n \text{ f-name}) = \text{value}$

"$g_n$" is the designator which shows the word-order "$_n$". "f-name" is a function name of schema, like word, letter or mark, etc. "value" is its schema 's value.

For example, the ill-formed input, "MR.Brown have eat a apple," is represented as surface schemata in Figure 5. "MR." is represented as four-surface schemata:

"(g1 word) = mr"; the word is "mr".
"(g1 mark) = period"; the mark after the word is a period.
"(g1 letter) = 1"; the first letter of the word is a capital ("M").
"(g1 letter) = 2"; the second of the word letter is a capital ("R").



Input sentence: *MR. Brown have eat a apple,
designators
gn
MR.
Brown
have
eat
a
apple,
g1
g2
g3
g4
g5
g6
Surface schemata
Word =[(g1 word)=mr, (g2 word)=brown, (g3 word)=have,
(g4 word)=eat, (g5 word)=a, (g6 word)=apple]
Mark =[(g1 mark)=period, (g6 mark)=comma]
Letter =[(g1 letter)=1, (g1 letter)=2, (g2 letter)=1]

Figure 5 Examples of surface schema

## 4.2 Lexicon
## [Lexical surface constraint]

In the lexicon, lexical features and constraints are involved as schemata. A constraint for a surface schema is called a surface constraint. A surface constraint is notated as follows:

$(\text{IT f-name}) =_c \text{value}.$

"IT" means meta-variable. "It" is substituted for "$g_n$", when the surface constraint is instantiated.

There are two kinds of surface constraints: lexical and grammatical. The capital letter "M" in "Mr." is a lexical constraint, because it is capitalized regardless of sentence position. A lexical surface constraint is assigned to the dictionary (Figure 6).

(IT word) $=_c$mr; the word must be "mr".

(IT mark) $=_c$period; the mark after the word must be a period.

(IT letter) $=_c$1; the first letter of the word "mr" must be a capital.

| Lexicon | | Lexical surface constraints | LFG schemata |
|---|---|---|---|
| noun3 | Mr. | (IT word) $=_c$mr<br>(IT mark) $=_c$period<br>(IT letter) $=_c$1 | (↑ PRED-1) = mr<br>(↑ GENDER) = male<br>(↑ CATEGORY) = noun3 |
| noun1 | Brown | (IT word) $=_c$brown<br>(IT letter) $=_c$1 | (↑ PRED) = Brown<br>(↑ PERSON) = 3<br>(↑ NUM) = SG<br>(↑ CATEGORY) = noun1 |

Figure 6 Lexicon

## 4.3 Grammar

### [Grammatical surface constraint]

The first letter in a sentence is always a capital letter and the last punctuation in a sentence is noted as a mark ( a period, a question mark or an exclamation point, etc.). These are regarded as grammatical constraints. In our framework these grammatical constraints are represented as grammatical surface constraints. They are assigned to grammar rule as shown in Figure 7 .

($\mathrm{IT_F}$ letter) $=_c$1; This means the first letter in the sentence must be a capital letter. $\mathrm{IT_F}$ shows first order in the sentence.

($\mathrm{IT_L}$ mark) $=_c$period; This means the last mark in the sentence must be a period. $\mathrm{IT_L}$ shows last order in the sentence.

Grammar rule

| S | → | NP | VP |
|---|---|---|---|
| | | (↑ SUBJ) = ↓<br>($\mathrm{IT_F}$ letter) $=_c$1 | ↑ = ↓<br>($\mathrm{IT_L}$ mark) $=_c$period |

Figure 7 Grammar rule with surface constraints

### [Interpretation schema]

In order to diagnose and correct errors, our framework has three steps; (1)collecting information on the input sentence, (2)synthesis of interpretation and (3) comparison of (1) and (2).

The interpretation schema collects LFG schemata and surface schemata. It is assigned to lexicon or grammar rules. In the parsing process, it is instantiated and collects schemata. The schemata corrected by interpretation schema are conveyed to the interpretation rule. This schema is notated as follows.

(↑ f-name) $=_i$\{values\}

↑ is a meta-variable as well as LFG notation and "f- name" is a functional name of the interpretation schema. Its values are sets of schemata.

For example an interpretation schema for agreement between determiner and noun is notated as follows.

① (↑ DET-NOUN) $=_i$\{[DET],[NOUN]\}

[DET] means set of schemata from determiner, and [NOUN] means from noun.

(Example 1) For the correctly-formed noun phrase "an apple", the interpretation schema, DET-NOUN, is attached to grammar rule (1) as shown in Figure 8. In instantiation, the interpretation schema collects LFG schemata in lexicon and surface schemata as its values below.

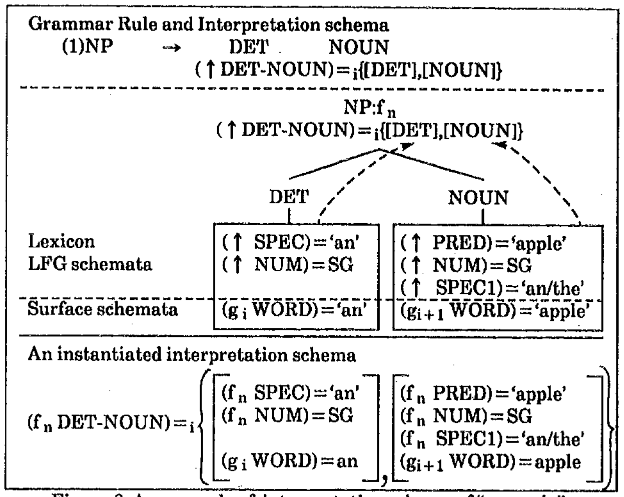


Figure 8 An example of interpretation schema of "an apple"

(Example 2) In another case, the ill-formed noun phrase "∅ apple", lacks an article. As above, an interpretation schema collects schemata in Figure 9.

Other examples of interpretation schemata and their attached grammar are shown in Figure 10.

Grammar Rule and Interpretation schema

(2)NP → NOUN

(↑ DET-NOUN) $=_i$\{[∅],[NOUN]\}

An instantiated interpretation schema

$$(f_n\ \text{DET-NOUN}) =_i \left\{ \begin{bmatrix} \varnothing \end{bmatrix}, \begin{bmatrix} (f_n\ \text{PRED}) = \text{'apple'} \\ (f_n\ \text{NUM}) = \text{SG} \\ (f_n\ \text{SPEC1}) = \text{'an/the'} \\ (g_j\ \text{WORD}) = \text{apple} \end{bmatrix} \right\}$$

Figure 9 An example of interpretation schema of "∅ apple"

### [Interpretation schema with a condition and conditional schema]

An interpretation schema with a condition, and its conditional schema are a pair and act as an interpretation schema. An interpretation schema with condition can act when there is a conditional schema. These schemata are notated as (a) an interpretation schema with a condition:

(↑ f-name) $=_{i-\mathrm{CON}}$\{values\}

and (b) a conditional schema:

(↑ f-name) $=_{\mathrm{CON}}$\{values\}.

For example, this schema ⑦ means that if a noun phrase [NP:$f_2$] is a pronoun [PRONOUN], it checks whether the case of pronoun is subjective [subj]. If the noun phrase is not a pronoun, such as "an apple", there is no need to check.

⑦ ($f_2$ CASE) $=_{i-\mathrm{CON}}$\{[NP:$f_2$],[subj]\}.

The following schema ⑧ is its conditional schema. It is attached to grammar rule (5) and means the noun phrase is a pronoun.

⑧ ($f_2$ CASE) $=_{\mathrm{CON}}$\{[PRONOUN]\}

### [Kill schema]

A kill schema is the instantiation inhibition mechanism. It works to kill the interpretation schemata and is notated as follows:

(↑ f-name) $=_k$\{(↑ f-name-1), (↑ f-name-2),.......\}.

(3)NP → DET ADJ NOUN
$(\uparrow \text{DET-ADJ-NOUN}) =_{i} \{[\text{DET}],[\text{ADJ}],[\text{NOUN}]\}$

(4)NP → ADJ NOUN
$(\uparrow \text{DET-ADJ-NOUN}) =_{i} \{[\emptyset],[\text{ADJ}],[\text{NOUN}]\}$

(5)NP → PRONOUN
$(\uparrow \text{CASE}) =_{\text{CON}} \{[\text{PRONOUN}]\}$

(6)$\text{S:f}_1$ → $\text{NP:f}_2$ $\text{VERB3:f}_1$ $\text{NP:f}_3$
$(\text{f}_1\ \text{SUBJ}) = \text{f}_2$ $(\text{f}_1\ \text{OBJ}) = \text{f}_3$
$(\text{IT}_F\ \text{letter}) =_{c} 1$ $(\text{IT}_L\ \text{mark}) =_{c} \text{period}$
$(\text{f}_1\ \text{SUBJ\&V-FORM}) =_{i} \{[\text{NP:f}_2],[\text{VERB3}]\}$
$(\text{f}_2\ \text{CASE}) =_{i\text{-CON}} \{[\text{NP:f}_2],[\text{subj}]\}$
$(\text{f}_3\ \text{CASE}) =_{i\text{-CON}} \{[\text{NP:f}_3],[\text{obj/poss}]\}$

(7)$\text{S:f}_1$ → NP:f2 $\text{AUX:f}_1$ $\text{VERB3:f}_1$ $\text{NP:f}_3$
$(\text{f}_1\ \text{SUBJ}) = \text{f2}$ $(\text{f}_1\ \text{OBJ}) = \text{f}_3$
$(\text{IT}_F\ \text{letter}) =_{c} 1$ $(\text{IT}_L\ \text{mark}) =_{c} \text{period}$
$(\text{f}_1\ \text{SUBJ\&A-FORM}) =_{i} \{[\text{NP:f2}],[\text{AUX}]\}$
$(\text{f}_1\ \text{AUX\&V-FORM}) =_{i} \{[\text{AUX}],[\text{VERB3}]\}$
$(\text{f}_2\ \text{CASE}) =_{i\text{-CON}} \{[\text{NP:f}_2],[\text{subj}]\}$
$(\text{f}_3\ \text{CASE}) =_{i\text{-CON}} \{[\text{NP:f}_3],[\text{obj/poss}]\}$

(8)$\text{S:f}_1$ → $\text{NP:f}_2$ $\text{VERB-be:f}_1$ $\text{NP:f}_3$
$(\text{f}_1\ \text{SUBJ}) = \text{f}_2$ $(\text{f}_1\ \text{COMP}) = \text{f}_3$
$(\text{IT}_F\ \text{letter}) =_{c} 1$ $(\text{IT}_L\ \text{mark}) =_{c} \text{period}$
$(\text{f}_1\ \text{SUBJ\&V-FORM\&COMP}) =_{i} \{[\text{NP:f}_2],[\text{VERB-be}],[\text{NP:f}_3]\}$
$(\text{f}_1\ \text{SUBJ\&V-FORM\&COMP}) =_{k} \{(\text{f}_2\ \text{DET-NOUN}), (\text{f}_2\ \text{DET-ADJ-NOUN})\}$

| Interpretation Schemata | Grammar rule |
|---|---|
| ① $(\uparrow \text{DET-NOUN}) =_{i} \{[\text{DET}],[\text{NOUN}]\}$ | Rule(1)(2) |
| ② $(\uparrow \text{DET-ADJ-NOUN}) =_{i} \{[\text{DET}],[\text{ADJ}],[\text{NOUN}]\}$ | Rule(3)(4) |
| ③ $(\text{f}_1\ \text{SUBJ\&V-FORM}) =_{i} \{[\text{NP:f}_2],[\text{VERB3}]\}$ | Rule(6)(8) |
| ④ $(\text{f}_1\ \text{SUBJ\&A-FORM}) =_{i} \{[\text{NP:f2}],[\text{AUX}]\}$ | Rule(7) |
| ⑤ $(\text{f}_1\ \text{AUX\&V-FORM}) =_{i} \{[\text{AUX}],[\text{VERB3}]\}$ | Rule(7) |
| ⑥ $(\text{f}_1\ \text{SUBJ\&V-FORM\&COMP}) =_{i} \{[\text{NP:f}_2],[\text{VERB-be}],[\text{NP:f}_3]\}$ | Rule(8) |
| **Interpretation Schemata with condition** | **Grammar rule** |
| ⑦ $(\text{f}_2\ \text{CASE}) =_{i\text{-CON}} \{[\text{NP:f}_2],[\text{subj}]\}$ | Rule(6)(7)(8) |
| **Conditional schema** | **Grammar rule** |
| ⑧ $(\uparrow \text{CASE}) =_{\text{CON}} \{[\text{PRONOUN}]\}$ | Rule(5) |
| **Kill schema** | **Grammar rule** |
| ⑨ $(\text{f}_2\ \text{SUBJ\&V-FORM\&COMP}) =_{k} \{(\text{f}_2\ \text{DET-NOUN}), (\text{f}_2\ \text{DET-ADJ-NOUN})\}$ | Rule(8) |

②This schema checks agreement between determiner, adjective and noun such as 'the same name', '*some good advices', '*a good jobs', and '*a interesting book'.

③This schema checks whether verb form (V-FORM) is a proper form for subject style (SUBJ). [NP:f2] is subject. For example "Tom gives...", "*He laugh ...", "You made ...." and "*Mr.and Mrs. Brown laughs ...".

④This schema checks whether auxiliary verb form (A-FORM) is a proper form for subject style (SUBJ). [NP:f2] is subject. For example "*Tom have given..." and "He can laugh ...".

⑤ This schema checks whether verb form (V-FORM) is a proper form for auxiliary verb. For example "Tom has given...", "*Tom has give...", "*You can laughed ..." and "He is speaking ..."

⑥This schema checks agreement between subjective noun phrase, verb "be" and compliment. [NP:f2] is subjective noun phrase and [NP:f3] is compliment. For example "*These is apples." , "*He is students." and "*They are a student."

Figure 10 Examples of grammar and interpretation schema

↑ is a meta-variable and "f-name" is a kill-schema's name. Its value in {.......} is the killed schamata's name.

There are hierarchy and priority between interpretation schemata. A kill schema is used to keep interpretation schemata independent. The schema attached to noun phrase can collect schemata only within the noun phrase, while the schema attached to sentence level can collect schemata in the sentence. Thus, the former is local and the latter is global. For example,

"* This is a apples. "

The noun phrase, " a apples ", is wrong and should be "an apple". But the local interpretation schema ① (Figure 10) can't determine which is correct, "an apple" or "apples", while the global interpretation schema ⑥ can judge that "an apple" is correct. The global interpretation schema ⑥ checks for agreements within [NP:f3] instead of the local interpretation schemata ① or ②. Therefore, the local interpretation schemata ① and ②, are not necessary. Thus, the kill schema ⑨, which corresponds to the global interpretation schema ⑥, kills local interpretation schemata ① and ②.

⑨ $(\text{f}_2\text{SUBJ\&V-FORM\&COMP}) =_{k} \{(\text{f}_2\ \text{DET-NOUN}), (\text{f}_2\ \text{DET-ADJ-NOUN})\}$

① $(\text{f}_2\ \text{DET-NOUN}) =_{i} \{[\text{DET}],[\text{NOUN}]\}$

② $(\text{f}_2\ \text{DET-ADJ-NOUN}) =_{i} \{[\text{DET}],[\text{ADJ}],[\text{NOUN}]\}$

## 4.4 Instantiation

How to instantiate schema is explained. Both ↑ and ↓ -meta-variables are assigned to actual variables ($\text{f}_1$, $\text{f}_2$....) as well as LFG.

A surface schema, a surface constraint and an interpretation schema include "IT" meta-variables. "IT" meta-variables are assigned as follows.

(1)In input processing, the designator "$\text{g}_n$" which shows the word-order in the input sentence is assigned to surface schema.

(2)When the dictionary is looked up, surface constraints in the lexicon are instantiated. "IT" meta-variable in a surface constraint is bound to the designator "$\text{g}_n$" in surface schema.

(3)When a grammar rule is fitted, surface constraints in the

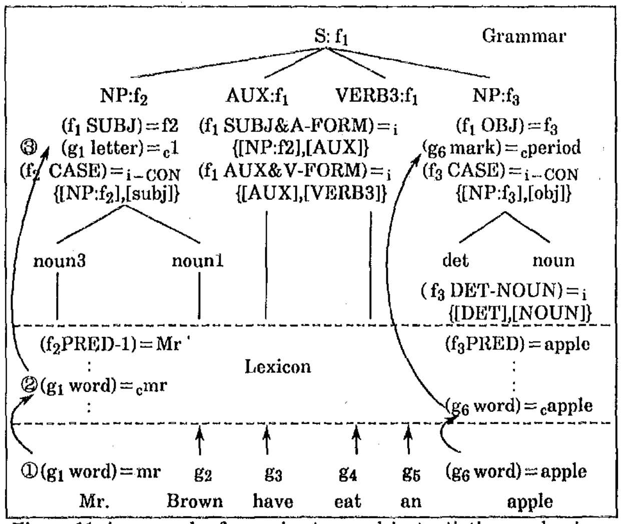


Figure 11 An example of a parsing tree and instantiation mechanism

grammar rule are bound to the designator "$g_n$".

An example is shown in Figure 11.

## 4.5 Interpretation (Filter II)

After the parsing process, interpretation schemata, interpretation schemata with a condition, conditional schemata and kill schemata are instantiated. Interpretation schemata are interpreted by interpretation rule. Input is judged for consistency or inconsistency.

The interpretation schemata are independent, thus the interpreted order is free. The interpretation flow is as follows.

(1)check conditional schema: if it is an interpretation schema with condition, find the paired condition. If conditional schema are not paired, inhibit the instantiated interpretation schema with a condition.

(2)check kill schemata: if the kill schema includes interpretation schemata which should be killed, inhibit the instantiated interpretation schema.

(3)Interpretation rule: if it is not included, interpret it.

### [Interpretation rule]

An interpretation rule diagnoses the input sentence. The schemata collected by an interpretation schema are checked by an interpretation rule. An interpretation rule synthesizes the word by using collected schemata. The diagnosis process is as follows.

(1)Find input style from an interpretation schemata.

(2)Synthesize correct style by using an interpretation rule.

(3)Compare input style with synthesized style. If consistent, the input style is right. If not, correct the input style to the synthesized correct style.

An interpretation rule synthesizes the result with conditions from interpretation schema. For example, the DET-NOUN rule is shown in Table 2. This rule determines if the noun is corrected and synthesizes the specification (SPEC) form as adapted for the noun.

(Example 1) In the case of correctly-formed noun phrase "an apple", the interpretation rule is shown in Figure 8.

(1)input style: ($g_i$ word) = an, ($g_{i+1}$ word) = apple from surface schemata in Figure 8.

(2)synthesized style: conditions are (↑NUM) = SG, (↑SPEC1) = 'an/the' from noun and (↑SPEC) = 'an' from determinant in Figure 8, the result is (↑SPEC) = an from Rule 8 in Table 2.

(3)Compare '($g_n$ word) = an' with '(↑SPEC) = an'. The value is the same. Thus this noun phrase is correctly-formed.

(Example 2) In the case of the ill-formed noun phrase "∅ apple" which lacks an article, the interpretation rules are shown in Figure 9.

(1)input style: ∅, ($g_j$ word) = apple from surface schemata.

(2)synthesized style: conditions are (↑NUM) = SG, (↑SPEC1) = 'an/the' from noun, the result is (↑SPEC) = an from rule 10 in Table 2.

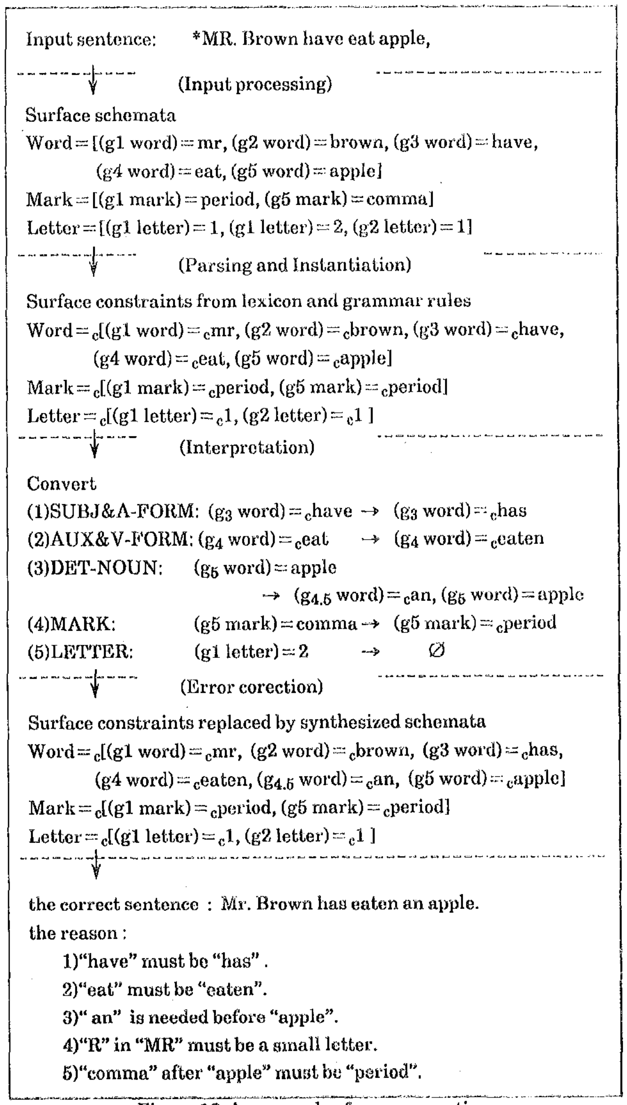
(3)Comparison ∅ with (↑SPEC) = an, as a result it lacks the article "an". Add the surface constraint "($g_{n-0.5}$ word) $=_c$ an" before "($g_n$ word) $=_c$apple".

Table 2 Interpretation Rule for DET-NOUN

| Rule No. | Conditions | | | Result |
|---|---|---|---|---|
| | NUM From noun | SPEC1 From noun | SPEC From det | SPEC |
| 1 | PL | the | — | the |
| 2 | PL | a | — | ∅ |
| 3 | PL | an | — | ∅ |
| 4 | PL | ∅ | — | ∅ |
| 5 | SG | a/the | a | a |
| 6 | SG | a/the | the | the |
| 7 | SG | a/the | — | a |
| 8 | SG | an/the | an | an |
| 9 | SG | an/the | the | the |
| 10 | SG | an/the | — | an |
| 11 | SG | X | — | X |

## 4.6 Error correction

The error correction phase explains the error to the user. For example, "*MR. Brown have eat apple," the flow of error correction is shown in Figure 12. Input sentence is converted into surface schemata and parsed. Surface constraints and interpretation schemata are then obtained. These interpretation rules are diagnosed and three errors found; (1)SUBJ&A-FORM, (2)AUX&V-FORM and (3)DET-NOUN



Input sentence: *MR. Brown have eat apple,

(Input processing)

Surface schemata

Word = [(g1 word) = mr, (g2 word) = brown, (g3 word) = have,
(g4 word) = eat, (g5 word) = apple]

Mark = [(g1 mark) = period, (g5 mark) = comma]

Letter = [(g1 letter) = 1, (g1 letter) = 2, (g2 letter) = 1]

(Parsing and Instantiation)

Surface constraints from lexicon and grammar rules

Word $=_c$[(g1 word) $=_c$mr, (g2 word) $=_c$brown, (g3 word) $=_c$have,
(g4 word) $=_c$eat, (g5 word) $=_c$apple]

Mark $=_c$[(g1 mark) $=_c$period, (g5 mark) $=_c$period]

Letter $=_c$[(g1 letter) $=_c$1, (g2 letter) $=_c$1 ]

(Interpretation)

Convert

(1)SUBJ&A-FORM: ($g_3$ word) $=_c$have → ($g_3$ word) $=_c$has

(2)AUX&V-FORM: ($g_4$ word) $=_c$eat → ($g_4$ word) $=_c$eaten

(3)DET-NOUN: ($g_5$ word) = apple
→ ($g_{4.5}$ word) $=_c$an, ($g_5$ word) = apple

(4)MARK: (g5 mark) = comma → (g5 mark) $=_c$period

(5)LETTER: (g1 letter) = 2 → ∅

(Error corection)

Surface constraints replaced by synthesized schemata

Word $=_c$[(g1 word) $=_c$mr, (g2 word) $=_c$brown, (g3 word) $=_c$has,
(g4 word) $=_c$eaten, ($g_{4.5}$ word) $=_c$an, (g5 word) $=_c$apple]

Mark $=_c$[(g1 mark) $=_c$period, (g5 mark) $=_c$period]

Letter $=_c$[(g1 letter) $=_c$1, (g2 letter) $=_c$1 ]

the correct sentence : Mr. Brown has eaten an apple.

the reason :

1)"have" must be "has".

2)"eat" must be "eaten".

3)" an" is needed before "apple".

4)"R" in "MR" must be a small letter.

5)"comma" after "apple" must be "period".

Figure 12 An example of error correction

(Figure 10). Furthermore, surface errors, (4)MARK and (5)LETTER, are found by the difference between surface schemata and surface constraints. The surface constraints are replaced by synthesized schemata. The corrected sentence, "Mr. Brown has eaten an apple. ", is then synthesized from surface constraints. The explanations 1)~5) are generated by the result of interpretation rule.

## 5.An application (英語CAI)

This framework to a CAI system, called " 英 語(English) CAI", was applied and designed to teach English to junior high school students. This system has two main modules; (1)machine translation /Kudo 1986/ and (2)this framework. If students produce ill-formed English input, the system corrects the errors and shows why they are wrong. If there are no errors, the sentence is translated into Japanese.

This system was implemented in Prolog (about 120KB). Performance is real-time (answers within 5 seconds). Actually this system was used by junior high school students. We collected mistakes and then fed back to the system.

This system is one of applications of this framework in a limited domain. The framework is easy to apply to another domain. To construct a new system, only need be changed the grammar, dictionary and interpretation rules.

## 6. Limitation and future work

The framework for grammatically ill-formed input was described. The following problems remain unsolved:
(1)The problem of semantically ill-formed input: in this framework a semantically ill-formed sentence is passed. A semantic filter must be added after filter (II).
(2)The problem of interpretation: interpretation is often changed by context and situation. Human beings correct ill-formed sentences by recognizing context and situation. For example, *He is a boy?*
Which interpretation is right, dialogue situation, word-order error (*Is he a boy ?*) or mispunctuation (*He is a boy.*)? A system will need a context recognizer and a situation recognizer.

## Conclusions

This paper has suggested the schema method, a new framework for correcting ill-formed input. This framework recognizes input at two steps with weak and strong filters. When it is known what sentences are passed by the filter, it can be used even if imperfect. This method has the following advantages:
(1)the problem of control strategies for relaxation can be avoided because the relaxation technique is not used, and
(2)computational efficiency.

The LFG framework for correcting grammatically ill-formed input was extended; a surface schema and an interpretation schema have been proposed. This framework can correct errors without breaking LFG framework, because these schemata, as well as LFG schema, can be treated. Therefore to make an applied system is very easy. This framework was implemented in Prolog to devise a useful CAI system.

## Acknowledgment

We would like to thank Akira Kurematu, president of ATR Interpreting Telephony Research Laboratories (ATR) and Mr.Yada, president of CSK Research Institute (CRI) for their constant encouragement. And we would also like to acknowledge the helpful comments of Hitoshi Iida (ATR). Many thanks also to Hideo Kobayashi (Nishi-Sugamo Junior High School), Kenji Okamoto, Yoshio Ooyama, Kei Okabe and Syuuichi Tanaka (CRI).

## References

Carbonell, J.G.&Hayes, P.J. (1983) ' Recovery Strategies for Parsing Extragrammatical Language' American Journal of Computational Linguistics, Volume 9 , pp.123-146.

Hayes, P.J & Mouradian, G.V. (1981) 'Flexible Parsing' American Journal of Computational Linguistics, 7(4), pp.232-242.

Heidorn, G.E. (1982) 'Experience with an Easily Computed Metric for Ranking Alternative Parses' Proceeding of 20th Annual Meeting of the ACL. Totont, Canada, pp.82-84.

Heidorn, G.E., Jensen, K., Miller, L.A. Byrd, R.J. and Codoro, M.S. (1982) 'The EPISTLE Text-Critiquing System' IBM Systems Journal 21 (3), pp.305-326.

Jensen, K., Heidorn, G.E., Miller, L.A. and Ravin, Y. (1983) 'Parse Fitting and Prose Fixing: Getting a Hold on ill-formedness' American Journal of Computational Linguistics, Volume 9, Number 3-4, July-December, pp.147-160.

Kaplan, R.M.& Bresnan, J. (1982) 'Lexical-Functional Grammar: A Formal System for Grammatical Representation' In:Bresnan, J. (ed) 'The Mental Representation of Grammatical Relations', The MIT Press, Cambrige, Massachusetts, pp.173-281.

Kudo, I. & Nomura, H. (1986) 'Lexical-functional Transfer: A Transfer Framework in a Machine Translation System Based on LFG', Proceeding of 11th International Conference on Computational Linguistics, Bonn, August, pp.112-114.

Kwasny, S.C. & Sondheimer, N.K. (1981) 'Relaxation Techniques for Parsing Grammatically Ill-formed Input in Natural Language Understanding Systems' American Journal of Computational Linguistics, Vol. 7, Number 2, April-June, pp.99-108.

Matumoto, I.& Matumoto, Y. (1976) 'A Practical Handbook of Common Mistakes in English among Japanese Students and Businessmen', Hokuseido

Schank, R.C .& Leboeitz, M. & Birnbaum, L. (1980) 'An Integrated Understander' American Journal of Computational Linguistics, Volume 6, Number 1, January-March, pp.13-30.

Schuster, E.(1985) 'Grammar as user models' Proceedings of the Nineth International Joint Conference on Artificial Intelligence, August, Los Angeles, California, pp.20-22

Weischedel, R.M. & Black, J.E. (1980) 'Responding Intelligently to Unparsable Inputs' American Journal of Computational Linguistics, Volume 6, Number 2, pp.97-109.

Weischedel, R.M. & Sondheimer, N.K. (1982) 'An Improved Heuristic for Ellipsis Processing' Proceeding of 20th Annual Meeting of the ACL. Totont, Canada, pp.85-88.

Weischedel, R.M. & Sondheimer, N.K. (1987) 'Meta-rules as a Basic for Processing Ill-formed Input' In:R.G.Reilly (ed.) Communication Failure in Dialogue and Discourse, Elsevier Science Publishers B.V. (North-Holland), pp.99-120.