The Parsing Algorithm of Translation Corresponding Tree (TCT) Grammar

Fai Wong*

Faculty of Science and Technology, University of Macau, Av. Pardre Tomás Pereira Taipa, Macau SAR, China
e-mail: derekfw@umac.mo

Abstract In machine translation (MT), parsing acts as a kernel step to analyze and acquire the syntactic information of an input sentence for the purpose to reproduce the corresponding translation in target language according to the syntactic relationships between the source and target sentences. The parsing process is guided by a set of language formalism, and the design of such algorithm is highly depending on how the language information it represents, especially in the development of example-based machine translation (EBMT) system, where the foundmental language information are the set of translation examples. In this paper, a parsing algorithm is designed to parse the Translation Corresponding Tree (TCT) structure based on the augmented GLR algorithm. The TCT, as the examples representation schema, has been used to the annotation of bilingual text in EBMT system. In order to achieve in parsing the tree language based on GLR parsing algorithm, a TCT equivalent synchronous formalism based on the notation of context free grammar (CFG) is proposed. Where the feature properties of a TCT structure can be fully expressed in term of the proposed formalism, and which, as a result, can be parsed by any CFG parsers. In this paper, GLR algorithm is extended and adapted to this parsing task in the development of an EBMT system for Portuguese to Chinese machine translation.

Key words: Machine Translation, Portuguese-Chinese MT, Translation Corresponding Tree (TCT), synchronous formalism, parsing algorithm

INTRODUCTION

This paper presents a schema to extend known recognition (parsing) algorithm for context free grammar (CFG) in order to obtain recognition algorithm for a type of grammaritcal formalism that models bilingual languages in the sence of synchronous. In particular, we use this schema to give recognition algorithm for Translation Corresponding Tree (TCT), the tree language that has been proposed by Wong [1] for bilingual text annotation. In our previous work [2] in the development of example-based machine translation (EBMT) system, TCT has been used as the representation structure to describe the translation examples that forms the foundamental knownledge of the example database. In EBMT, translation generally involves two operations: *recognizing* the constituent structure utterances of an input sentence against the knowledge database to extract suitable examples and *transferring* (or *recombining*) the fragments of examples in an analogical manner to determine the correct translation [3]. To be specific, the transfer operation is actually the process of deciding which fragment in target language sentence corresponds to the fragment in source language sentence. From the design point of view, the *recognizing* and *transferring* algorithms are highly depending on the representation format of linguisitic data. For example, in [4], an Earley algorithm is adapted for parsing the Tree Adjoining Grammars (TAG) [5]; and more works of adapting known parsing algorithms for recognizing different language formalisms (or language resources) can be found in [6]-[7]. However, to recognize the TCT language gives a different challenge in our current work, as it models the translation examples in a parallel case. Both the source and tareget (language) sentences are described with a single tree structure of TCT. Where the syntactic relationships between the source and target sentences are modeled, as well as their corresponding sentence utterances and constituent fragements are also captured by the inter levels of the structure tree. In addition, the concept of *parsing as translation* by integrating the operations of *recognizing* and *transferring* into the

parsing algorithm has been the central idea behind our current research work. This motivates us to develop an equivalent language formalism of CFG like production rules for the TCT language, such that it can be parsed by known CFG parsing algorithms.

This paper is organized as follows. Section 2 describes the translation corresponding tree (TCT) structures. Section 3 describes the proposed synchronous formalism which will be the intermediate parsable formalism for the TCT (tree) language, and section 4 discusses the transformation method from TCT languages into the synchronous formalism linked with lexical constraints. The augmented recognizing algorithm based on GLR for the formalism will be introduced in section 5, and section 6 presents the application of parsing algorithm in machine translation system, followed by a conculsion to end the paper.

KNOWLEDGE REPRESENTATION SCHEMA

Translation Corresponding Tree (TCT) [2] structure, as a translation example representation schema, has been used to the construction of bilingual knowledge base in EBMT system development, where each translation example is being described by a TCT structure. The main advantages in application to machine translation include: 1) it models bilingual text by using a single syntactic tree structure, it requires only one language parser to acquire the syntactic structure instead of two for the case of synchronous representation; 2) the use of sequence of mapping functions makes it suitable for describing translation examples that are not *parallel* translations nor *close* syntactic structures [8]. That is, the source sentence and target sentence do not have explicit corresponding constituents. The second property is quite important since in practical application, most of the translation examples are of *free translations*. This gives the language annotator enough flexibility to describe the linguistic relationship between the pair of sentences.

Following the definition of [1], the TCT structure is a general structure. It can flexibly associate a sentence string not only to its syntactic structure in the source language, but also explicitly to its translation in the target language. Therefore, it can describe the linguistic correspondences between different languages.

Definition 1 A TCT can be described as a 4-tuple ($\boldsymbol{T}_s$, ξ_s, ξ_t, σ), where $\boldsymbol{T}_s$ is any syntactic tree that describes the internal structure of a sentence in the source language, ξ_s and ξ_t are the sentence-pairs in the source and target languages, and σ is the correspondence between the syntactic tree $\boldsymbol{T}_s$ and the sentences ξ_s and ξ_t.

Definition 2 The correspondence σ between the syntactic tree $\boldsymbol{T}_s$ and the sentences ξ_s and ξ_t can be further defined as a triple (SNODE**,** STREE**,** TTREE), where SNODE and STREE are the substring intervals in the source sentence, and STC is the substring intervals in the target sentence.

The TCT structure uses triple sequence intervals [SNODE(n)/STREE(n)/TTREE(n)$\in\sigma$] encoded for each node in the tree to represent the corresponding relationships between the structure of the source sentence and the substrings from both the source and target sentences. Each set of corresponding information is made up of the following three interrelated correspondences: 1) between the node and the substring of the source sentence encoded by the interval SNODE(n), which denotes the interval containing the substring corresponding to the node; 2) between the subtree and the substring of the source sentence represented by the interval STREE(n), which indicates the substring interval dominated by the subtree with the node as root; 3) between the subtree and the substring of the target sentence represented by the interval TTREE (n), which indicates the interval containing the target sentence substring corresponding to the source sentence subtree. The associated substrings may be discontinuous in all cases. It preserves the ability to describe non-standard and non-projective linguistic phenomena for a language [9]. The schema also allows the annotator to flexibly define the corresponding translation from the target sentence to the source sentence structure when necessary. In practical use, the structure is extended to include extra linguistic constraints beside the syntactic part of speech that are useful to the specific purpose. For example, in the EBMT application, syntactic relationship between the source and target languages is included in the inter nodes of the structure for use to constraint the generation order for the translation in target language. In the construction of the example database in EBMT system, collection of constructed TCTs form the

elementary knowledge to facilitate the sentence translation. Fig. 1 shows the portions of translation examples represented in terms of the TCT structures.

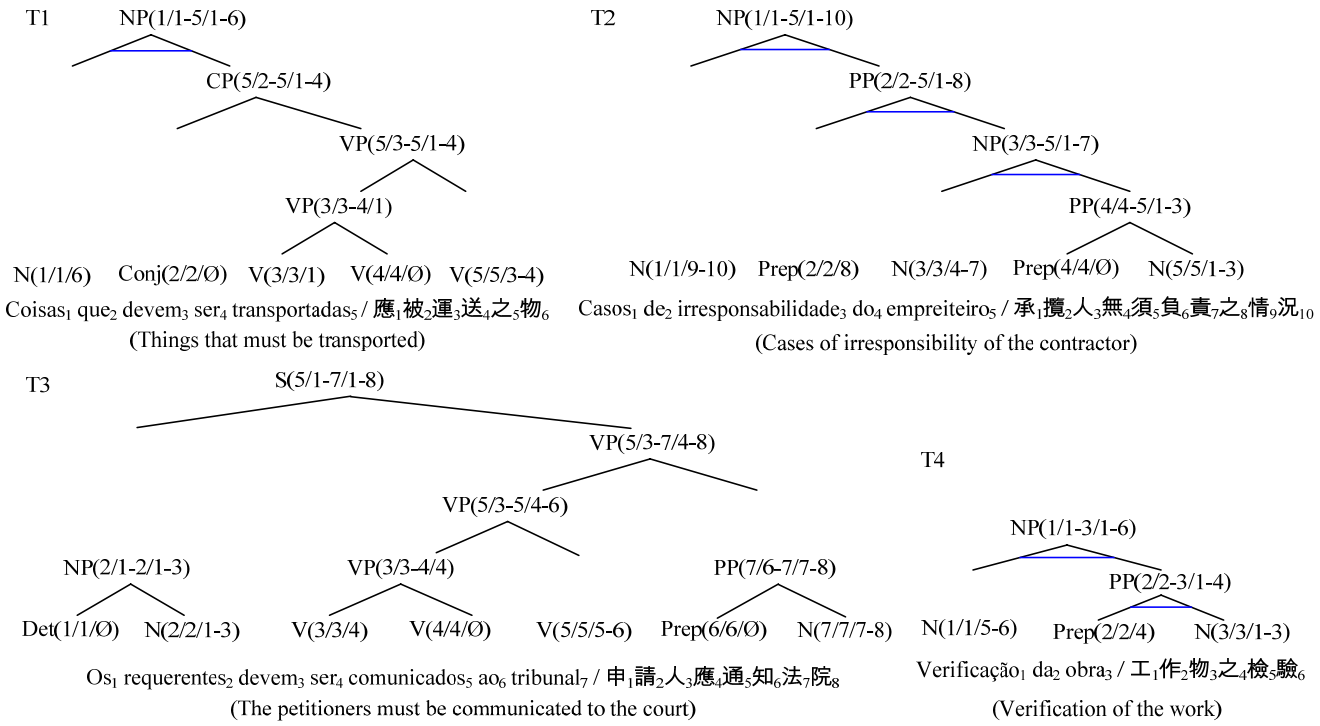

Fig. 1 Collection of translation examples annotated with TCT structures as the basic elements in the translation knowledge databse

SYNCHRONOUS FORMALISM

Before the TCT language can be parsable, the structure must be first transformed into some formalism that can be recognized by any known CFG parsing algorithm. The choice of the formalism is essential for the representation and the understanding of linguistic phenomena. It is also important to consider its applicability for MT applications. In our work, we propose to use the formalism of Constraint Synchronous Grammar (CSG) [10] as the equivalent grammar for the TCT. CSG is a variation of synchronous grammars [11] that is based on the formalism of CFG. In CSG formalism, it consists of a set of production rules that describes the sentential patterns of the source text and target translation patterns. Every production rule of CSG is in the form of:

S → *source sentential pattern* { [*target sentential pattern*; control conditions] ,
[*target sentential pattern*; control conditions] ,
...
}

In the left hand side, *S* is the reduced syntactic symbol. In the right hand side of the production, it is divided into two components: the sentential pattern of the source language, and the translation pattern of the target language. Furthermore, in each sentential pattern of the source language, it may consist of one or more translation patterns associated with control conditions based on the features of non-terminal symbols of the source rule for describing the possible generation correspondences in target translation. These conditions are not only used for inferring the structure of source input in the parsing module, but also for the structure of the target output pattern in the generation module. Formally, the formalism is defined as:

Definition 3 A *Constraint-Based Synchronous Grammar* (*CSG*) is 5-tuple $G = (V_N, V_T, P, C_T, S)$ which satisfies the following conditions:

- V_N is a finite set of *non-terminal* symbols;
- V_T is a finite set of *terminal* symbols which is disjoint with V_N;
- C_T is a finite set of *target components*;
- P is a finite set of *productions* of the form $A \rightarrow \alpha\ \beta$, where $\alpha \in (I(V_N) \cup V_T)^*$ and, $\beta \in C_T$, the non-terminal symbols that occur from both the source and target rules are *linked* under the index given by $I(V_N)$.
- $S \in V_N$ is the initial symbol.

Where *target component*, C_T, can be defined as a ordered vector of *target rules* in γ (pair of *rule* and *constraint*, $[r \in R^*, c \in C^*]$, and $\gamma = R^* \times C^*$ in form of $[r, c]$) having the form $\sigma = \{\gamma_1, \ldots, \gamma_q\}$, where $1 \le i \le q$ to denote the i-th tuple of σ. The *target rules* are being arranged in the order of $\gamma_1 \prec \gamma_2 \prec \ldots \prec \gamma_q$ determined by the degree of generalization rule according to the associated constraint.

Based on this synchronous formalism, pair of languages can be modeled and analyzied simultaneously. For example, the following CSG productions can generate both of the parallel texts ["*Ele deu um livro ao José.* (He gave a book to José)", "*他給了若澤一本書*"] and ["*Ele comprou um livro ao José*. (He bought a book from José)", "*他向若澤買了一本書*"]:

$$\begin{aligned} S \rightarrow NP_1\ VP^*\ NP_2\ PP\ NP_3\ \{&[NP_1\ VP^1\ NP_3\ VP^2\ NP_2; VP_{category}=vb1,\\ &\quad VP_{sense\ of\ subject} = NP_{1sense},\\ &\quad P_{sense\ of\ indirect\ object}=NP_{2sense},\\ &\quad VP_{sense\ of\ object}=NP_{3sense}],\\ &[NP_1\ VP\ NP_3\ NP_2\ ; \quad VP = vb0,\\ &\quad VP_{sense\ of\ subject} = NP_{1sense},\\ &\quad VP_{sense\ of\ indirect\ object} = NP_{2sense}]\\ \}& \end{aligned} \tag{1}$$

$$VP \rightarrow v\ \{[v\ ;\ \varnothing]\} \tag{2}$$

$$NP \rightarrow det\ NP^*\ \{[NP\ ;\ \varnothing]\} \tag{3}$$

$$NP \rightarrow num\ NP^*\ \{[num\ 本\ NP;\ NP_{sense}=sense\ of\ book]\} \tag{4}$$

$$NP \rightarrow n\ \{[n\ ;\ \varnothing]\} \tag{5}$$

$$NP \rightarrow pro\ \{[pro\ ;\ \varnothing]\} \tag{6}$$

$$PP \rightarrow p\ \{[p\ ;\ \varnothing]\} \tag{7}$$

$$n \rightarrow José\ \{[若澤\ ;\ \varnothing]\} \mid livro\ \{[書\ ;\ \varnothing]\} \tag{8}$$

$$pro \rightarrow ele\ \{[他\ ;\ \varnothing]\} \tag{9}$$

$$v \rightarrow deu\{[給了\ ;\ \varnothing]\} \mid comprou\ \{[向, 買了\ ;\varnothing]\} \tag{10}$$

$$num \rightarrow um\ \{[一\ ;\ \varnothing]\} \tag{11}$$

$$p \rightarrow a\ \{\varnothing\} \tag{12}$$

$$det \rightarrow o\ \{\varnothing\} \tag{13}$$

Production (1), as a typical rule representation, has two generative rules associated with the sentential pattern of the source $NP_1\ VP^*\ NP_2\ PP\ NP_3$. The determination of the suitable generative rule is based on the control conditions defined by rule. The one satisfying all the conditions determines the relationship between the source and target sentential pattern. For example, if the category of the verb is vb1, and the sense of the subject, indirect, and direct objects governed by the verb, VP, corresponds to the first, second, and the third nouns (NP), then the source pattern $NP_1\ VP^*\ NP_2\ PP\ NP_3$ is associated with the target pattern $NP_1\ VP^1\ NP_3\ VP^2\ NP_2$. Their relationship is established by their given *subscripts* and the sequence is based on the target sentential pattern. In other words, in the production $S \rightarrow NP_1\ VP^*\ NP_2\ PP\ NP_3$ $[NP_1\ VP\ NP_3\ NP_2]$, although the first NP and the verb corresponds to each other in the same sequence, the sequence for the second and third NP in the source are changed in the target sentential pattern. The asterisk "*" indicates

the head element, and its usage is to propagate all the related features/linguistic information of the head symbol to the reduced non-terminal symbol in the left hand side. The use of the "*" is to achieve the property of features inheritance in CSG formalism. The *superscripts* of the syntactic symbols represents the *fan-out* relationship for distinuous constitiuents of sentences, due to the structure deviations of two different languages, in particular for languages from different families such as Portuguese and Chinese [12]. This allows the source and target production components rewritten indepently, and is flexibly enough for the description of typical linguistic phenomena.

CONVERSION OF TCT STRUCTURE

The objective to convert the collection of TCT structures into the proposed synchronous formalism, as discussed in the first section, is obvious. The transformation involves the extraction of generalized syntactic and lexicalized grammar rules from the set of elementary structures. The lexicalization of a syntactic grammar consisits of the association of a set of lexicons to the syntactic constituents. Which merged the lexicon(s) and grammar in a single entity. Lexicalization provides at least two advantages: First, the ability to describe syntactically each specific lexical entry allows us to choose the required complexity of the syntactic structures with flexibility. Too much generalization in syntactic descriptions generally results in unexpected border effects. Secondly the lexicalization allows parsing heuristics according to the lexical constraints which can greatly reduce the search space as well as the number of analytical structures due to the ambiguaties of language [13]. During the conversion process, the possible syntactic constituents (inter levels) of trees are first extracted and rewritten into corresponding grammar rules together with the associated lexical items starting from the leaves, in the manner of bottom up, and level by level towards the root node. The syntactic symbol of each structure node which has different lexical items in its dominated leaves is distinct by appending with unqiue *subscripts*. Following shows the rewritten (lexicalized) synchronous grammars for, T3, one of the TCT structures in Fig. 1:

$$S_1 \rightarrow NP_{os\_req}\ VP_{VP_{deser\_com}\_PP_{aotri}}\ \{NP_{os\_req}\ VP_{VP_{deser\_com}\_PP_{aotri}}\} \quad (14)$$

$$VP_{VP_{deser\_com}\_PP_{aotri}} \rightarrow VP_{devser\_com}\ PP_{aotri}\ \{VP_{devser\_com}\ PP_{aotri}\} \quad (15)$$

$$VP_{devser\_com} \rightarrow VP_{dev\_ser}\ v_{com}\ \{VP_{dev\_ser}\ v_{com}\} \quad (16)$$

$$VP_{dev\_ser} \rightarrow v_{dev}\ v_{ser}\ \{v_{dev}\ v_{ser}\} \quad (17)$$

$$NP_{os\_req} \rightarrow det_{os}\ n_{req}\ \{det_{os}\ n_{req}\} \quad (18)$$

$$PP_{ao\_itri} \rightarrow p_{ao}\ n_{tri}\ \{p_{ao}\ n_{tri}\} \quad (19)$$

$$det_{os} \rightarrow os\ \{\emptyset\} \quad (20)$$

$$p_{ao} \rightarrow ao\ \{\emptyset\} \quad (21)$$

$$v_{dev} \rightarrow deven\ \{應\} \quad (22)$$

$$v_{ser} \rightarrow ser\ \{\emptyset\} \quad (23)$$

$$v_{com} \rightarrow comunicados\ \{通知\} \quad (24)$$

$$n_{req} \rightarrow requerentes\ \{申請人\} \quad (25)$$

$$n_{tri} \rightarrow tribunal\ \{法院\} \quad (26)$$

In order to make this example grammar more readable, each syntactic symbol is distinguished with the *subscripts* by using the prefix of the associated lexicon. In practice, we use the *sequence subscripts* for labeling the syntactic (reduced) symbols of production rules for the easy of manipulation and processing.

Following up the transformation, in order to resolve the effects of sparse data when working with production rules, we would like to generate generalized productions, which include nonterminal symbols that can be filled with other constituents. Therefore, after extracting the initial lexicalized rules from the TCT structures, we recursively generalize each existing rule. However, we abstract only if rules contains common lexical elements both in the source and target components, by replacing with corresponding nonterminal syntactic symbol. After the abstraction, both the used lexicalized rules and the generalized rules will be retained as the final grammar for parsing.

PARSING ALGORITHM

The transformed synchronous formalism can be parsed by any known CFG recognition algorithm including the Earley [14] and GLR [15] algorithms. In our work, the generalized LR algorithm is adapted to recognize our formalism augmented by taking into account the features constraints and the inference of target structure, hence to realize the recognition algorithm for synchronous grammar. The extension of GLR algorithm involves two parts: the *parse table* and the *recognition* mechanism. Since GLR algorithm uses a parse table to achieve a considerable efficiency over the Earley's non-compiled method which has to compute a set of LR items at each stage of parsing [15]. The parse table is further extended by engaging with the features constraints and the target rules into the actions table. Our strategy is thus to parse the source rules of the productions through the normal shift actions proposed by the parsing table, while at the time reduce action to be fired, the associated conditions are checked to determine if the active reduction is a valid action or not depending on if the working symbols of patterns fulfill the constraints on features.

Ste	ACTIONs/GOTOs																		Reduced Rules Constraints/Target Rules
	pro	*num*	*n*	*v*	*det*	*p*	*NP*	*VP*	*PP*	*S*	⊥	*o*	*a*	*um*	*ele*	*José*	*livro*	*deu*	*comprou*	
0	s8	s9	s10		s11		s7			s6		s5		s2	s1	s4	s3			
1															r1					(1) pro → *ele* {[*他*; ∅]}
2														r1						(1) num → *um*
3																	r1			(1) n → *livro* {[*書*; ∅]}
4																r1				(1) n → *José* {[*若澤*; ∅]}
5												r1								(1) det → *o*
6											acc									
7				s14				s15										s12	s13	
8	r1																			(1) NP → pro
9	s8	s9	s10		s11		s16					s5		s2	s1	s4	s3			
10			r1																	(1) NP → n
11	s8	s9	s10		s11		s17					s5		s2	s1	s4	s3			
12																		r1		(1) v → *deu* {[*給了*; ∅]}
13																			r1	(1) v → *comprou* {[*向, 買了*;∅]}
14				r1																(1) VP → v
15	s8	s9	s10		s11		s18					s5		s2	s1	s4	s3			
16							r1													(1) NP → *num* NP* {[num *本* NP; NP_{sem}=$SEM_{\_book}$]}
17							r1													(1) NP → det NP* {[NP ; ∅]}
18						s21			s20				s19							
19													r1							(1) p → *a*
20	s8	s9	s10		s11		s22					s5		s2	s1	s4	s3			
21						r1														(1) PP → p
22										r1										(1) S → NP_1 VP* NP_2 PP NP_3 {[...]}

Fig. 2 Extended LR(1) parse table

Extended Parse Table

Fig. 2 shows an extended LR(1) parsing table for Productions (1)-(13)1 as constructed using the LR table construction method described in [16] extended to consider the rule components of productions by associating the corresponding target rules with constraints, which are explicitly expressed in table. The

[1] For simplicity, the productions used for building the parse table are deterministic, so no conflict actions such as shift/reduce and reduce/reduce appear in the parse table in Fig. 2

parsing table consists of two parts: a compact ACTION-GOTO table[2] and CSONTRAINT-RULE table. The ACTION-GOTO table *s* indexed by a state symbol *s* (row) and a symbols $x \in V_N \cup V_T$, including the end marker "⊥". The entry ACTION[*s*, *x*] can be one of the following: *s n*, *r m*, *acc* or *blank*. *s n* denotes a shift action representing GOTO[*s*, *x*]=*n*, defining the next state the parser should go to; *r m* means a reduction by the m^{th} production located in the entry of CONSTRAINT-RULE in state *s*, and *acc* denotes the accept action and *blank* indicates a parsing error. The CONSTRAINT-RULE table is indexed by state symbol *s* (row) and the number of productions *m* that may be applied for reduction in state *s*. The entry CONSTRAINT-RULE[*s*, *m*] consists of a set of involved productions together with the target rules and features constraints that are used for validating if the active parsing node can be reduced or not, then try to identify the corresponding target generative rule for reduced production.

Modified Recognition Algorithm

In the parsing process, the algorithm operates by maintaining a number of parsing processes in parallel, each of which represents an individual parsed result, hence to handle the case of non-deterministic. In general, there are two major components in the process, *shift*(*i*) and *reduce*(*i*), which are called at each position *i*=0, 1, …, *n* in an input string $I = x_1x_2\ldots x_n$. The *shift*(*i*) process with top of stack vertex *v* shifts on x_i from its current state *s* to some successor state *s*' by creating a new leaf *v*'; establishing edge from *v*' to the top of stack *v*; and making *v*' as the new top of stack vertex. The *reduce*(*i*) executes a reduce action on a production *p* by following the chain of parent links down from the top of stack vertex *v* to the ancestor vertex from which the process began scanning for *p* earlier, then popping intervening vertices off the stack. Now, for every reduction action in *reduce*(*i*), there exists a set *C* of ordered constraints, $c_1 \prec \ldots \prec c_m$, with the production, each of which is associated with a target rule that may be the probable corresponding target structure for the production, depending on whether the paired constraint gets satisfied or not according to the features of the parsed string *p*. Before reduction takes place, the constraints c_j ($1 \le j \le m$) are tested in order started from the most specific one, the evaluation process stops once a positive result is obtained from evaluation. The corresponding target rule for the parsed string is determined and attached to the reduced syntactic symbol, which will be used for rewriting the target translation in phase of generation. At the mean while, the features information will be inherited from the designated head element of production. The parsing algorithm for the formalism is given in Fig. 3.

```
PARSE(grammar, x1 … xn)
  xn+1 ⇐ ⊥
  Ui ⇐ ∅ (0 ≤ i ≤ n)
  U0 ⇐ v0
  for each terminal symbol xi (1 ≤ i ≤ n)
    P ⇐ ∅
    for each node v ∈ Ui-1
      P ⇐ P ∪ v
      if ACTION[STATE(v), xi] = "shift s'", SHIFT(v, s')
      for each "reduce p" ∈ ACTION[STATE(v), xi], REDUCE(v, p)
      if "acc" ∈ ACTION[STATE(v), xi], accept
    if Ui=∅, reject

SHIFT(v, s)
  if v' ∈ Ui s.t. STATE(v')=s and ANCESTOR(v', 1)=v and state transition δ(v, x)=v'
    do nothing
  else
    create a new node v'
    s.t. STATE(v')=s and ANCESTOR(v', 1)=v and state transition δ(v, x)=v'
    Ui ⇐ Ui ∪ v'

REDUCE(v, p)
  for each possible reduced parent v1' ∈ ANCESTOR(v, RHS(p))
    if UNIFY(v, p)="success"
      s" ⇐ GOTO(v1', LHS(p))
```

[2] Original version introduced in [15] maintains two tables, ACTION and GOTO.

```
    if node v″∈U_{i-1} s.t. STATE(v″)=s″
      if δ(v_1′, LHS(p))=v″
        do nothing
      else
      if node v_2′∈ANCESTOR(v″,1)
        let v_c″ s.t. ANCESTOR(v_c″,1)=v_1′ and STATE(v_c″)=s″
        for each "reduce p″ ∈ ACTION[STATE(v_c″),x_i]
          REDUCE(v_c″,p)
      else
        if v″∈P
          let v_c″ st. ANCESTOR(v_c″,1)=v_1′ and STATE(v_c″)=s″
          for each "reduce p″ ∈ ACTION[STATE(v_c″),x_i]
            REDUCE(v_c″,p)
        else
          create a new node v_n
          s.t. STATE(v_n)=s″ and ANCESTOR(v_n,1)=v_1′ and
          state transition δ(v_n,x)=v_1′
          U_{i-1}⇐U_{i-1}∪v_n
  else current reduction failed

UNIFY(v,p)
  for "constraint c_j″ ∈ CONSTRAINT(STATE(v)) (1 ≤ j ≤ m, c_1≺…≺c_m)
    if ξ(c_j,p)="true"            (ξ(∅,p)="true")
      TARGET(v)⇐j
      return "success"
```

Fig. 3 Modified generalized LR Parsing algorithm

The parser takes two arguments PARSE(*grammar*, $x_1 \ldots x_n$), where the grammar is provided in form of parsing table. It calls upon the functions SHIFT(v, s) and REDUCE(v, p) to process the shifting and rule reduction as described. The UNIFY(v, p) function is called for every possible reduction in REDUCE(v, p) to verify the legal reduction and select the target rule for the source structure for synchronization. The function TARGET(v) after unification passed is to dedicate the j^{th} target rule as correspondence.

PARSING AS TRANSLATION

In our current research work, TCT language has been used to construct the translation knowledge base in the development of Portuguese to Chinese machine translation system based on example-based translation approach. In addition, the TCT structures are further rewritten into the synchronous formalism as the final representation for these knowledge. Where both the syntactic constituents and corresponding lexical items between the source and target sentences are being modeled and described through the pair of CFG like production rules as discussed in this paper. The translation of sentence based on the proposed formalism and the corresponding recognition algorithm is totally achieved by the parser component. Formally, the analysis of sentence can be described as follows:

Definition 4 A set P of productions is said to *accept* an input string s iff there is a derivation sequence Q for s using source rules of P, and any of the constraint associated with every *target component* in Q is satisfied[3]. Similarly, P is said to *translate* s iff there is a synchronized derivation sequence Q for s such that P accepts s, and the link constraints of associated *target rules* in Q is satisfied. The derivation Q then produces a translation t as the resulting sequence of terminal symbols included in the determined target rules in Q.

Hence, to the translation of an input text, it essentially consists of three steps. First, for an input sentence s, the structure of string is analyzed by using the rules of source components from the syncrhonous productions; by using the augmented generalized LR parsing algorithm as described. Secondly, the link constraints that are determined during the rule reduction process are propagated to the corresponding target rules R (as selection of target rules) to construct a target derivation sequence Q. And finally, based

[3] If there is no any constraint associated to a target rule, during the parsing phase, the reduction of the source rule is assumed to be valid all the time.

on the derivation sequence *Q*, translation of the input sentence *s* is generated by referencing the set of generative rules *R* that attached to the corresponding constituent nodes in the parsed tree, hence to realize the translation in target language.

In the preliminary experiement, 500 TCT annotated translation example of Portuguese-Chinese sentences are used to construct the synchronous formalism by extracting the syntactic and lexicalized rules from the fragments of structures. The average sentence length is 12.5 words. To test the system, another 50 sentences excluded from the training set are used for translation. According to the evaluation creation, the translations of 31 sentences are classified as *good* (74%), 6 of them belong to *acceptable* (12%), while 7 of them are *failure* (14%) or cannot be translated by the system due to the lack of unknown words and the syntactic rules. The overall translation accuracy of the system reaches 86%. This illustrates the feasibility of our proposed algorithm to integrate the process of *transferring* into the recognition algorithm. That is the parsing of source sentence and obtaining the translation in target language are carried out in parallel at the same phase.

CONCLUSION

Translation system based on example-based paradigm involves the processes of parsing the source sentence and transferring the recognized sentence fragments into the corresponding translation sentence. The operations concern with the type of linguistic data it uses, and TCT language has been the knowledge representation schema applied in our existing translation system. Which is a synchronous based annotation schema for describing the pair of translation sentences. Parsing of it requires us to propose an equivanlent intermediate synchronous formalism based on CFG productions such that it can be parsed by any known CFG parsing algorithm. In this research, we extended the generalized LR algorithm by introducing the feature constraints and the inference of target sentence syntax into the mechanism, to obtain a recognition algorithm for the TCT tree language. The preliminary empirical result shows that the proposed algorithm is feasible, and has been applied to the machine translation system for Portuguese to Chinese translation.

Acknowledgements The research work reported in this paper was partially supported by "*Fundo para o Desenvolvimento das Ciências e da Tecnologia*" (Science and Development Fund) under grant 041/2005/A and *Center of Scientific and Technological Research* (University of Macau) under Cativo: 5571.

REFERENCES

[1] F. Wong, M. C. Dong, and D. C. Hu, *Machine Translation Based on Translation Corresponding Tree Structure,* Tsinghua Science and Technology, 11, (2006), 25-31,.

[2] F. Wong, D. C. Hu, Y. H. Mao, and M. C. Dong, *A Flexible Example Annotation Schema: Translation Corresponding Tree Representation,* in Proceedings of *The 20th International Conference on Computational Linguistics*, Switzerland, Geneva (2004), pp. 1079-1085.

[3] S. Sato and M. Nagao, *Toward Memory-Based Translation*, in Proceedings of *The 13th International Conference on Computational Linguistics*, Finland, Helsinki (1990), pp. 247-252.

[4] Y. Schabes and A. K. Joshi, *An Earley-Type Parsing Algorithm for Tree Adjoining Grammars*, in Proceedings of *The 26th Annual Meeting on Association for Computational Linguistics*, Morristown, NJ, USA (1988), pp. 258-269.

[5] A. K. Joshi, *An Introduction to Tree Adjoining Grammars*, Alexis Manaster Ramer ed. John Benjamins Publishing Co., Amsterdam/Philadephia (1987).

[6] K. V. Shanker and D. J. Weir, *Parsing Some Constrained Grammar Formalisms*, Computational Linguistics, 19, (1994), pp. 591-636.

[7] M. A. Alonso, D. Cabrero, E. d. l. Clergerie, and M. Vilares, *Tabular Algorithms for TAG Parsing*, in Proceedings of *EACL'99, Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Bergen, Norway (1999), pp. 150-157.

[8] R. Grishman, *Iterative Alignment of Syntactic Structures for a Bilingual Corpus*, in Proceedings of *Second Annual Workshop on Very Large Corpora (WVLC2)*, Kyoto, Japan (1994), pp. 57-68.

[9] C. Boitet and Y. Zaharin, *Representation trees and string-tree correspondences*, in Proceedings of *COLING-88*, Budapest (1988), pp. 59-64.

[10] F. Wong, D. C. Hu, Y. H. Mao, M. C. Dong, and Y. P. Li, *Machine Translation Based on Constraint-Based Synchronous Grammar*, in Proceedings of *The Second International Joint Conference on Natural Language (IJCNLP-05)*, Jeju Island, Republic of Korea (2005), pp. 612-623.

[11] P. M. Lewis and R. E. Stearns, *Syntax-directed transduction*, Association for Computing Machinery, 15, (1968), pp. 456-488.

[12] F. Wong and Y. H. Mao, *Framework of Electronic Dictionary System for Chinese and Romance Languages*, Automatique des Langues (TAL), 44, (2003), pp. 225-245.

[13] A. Abeillé, B. Daille, and A. Husson, *FTAG: An Implemented Tree Adjoining Grammar for Parsing French Sentences*, in Proceedings of *TAG+3*, Paris (1994).

[14] J. Earley, *An Efficient Context-Free Parsing Algorithm*, CACM, 13, (1970), pp. 94-102.

[15] M. Tomita, *An Efficiency Augmented Context-Free Parsing Algorithm*, Computational Linguistics, 13, (1987), pp. 31-46.

[16] A. V. Aho, R. Sethi, and J. D. Ullman, *Compiler: Principles, Techniques and Tools* Addison-Wesley (1986).