Hypothesis Scoring over Theta Grids Information in Parsing Chinese Sentences with Serial Verb Constructions

Koong H. C. Lin and Von-Wun Soo
Department of Computer Science, National Tsing-Hua University HsinChu,
Hsinchu, 30043, Taiwan, R.O.C.
E-Mail:soo@cs.nthu.edu.tw

Abstract

Serial verb constructions (SVCs) in Chinese are popular structural ambiguities which make parsing difficult. In this paper, we propose a quantitative model, 'S-model', based on theta grids information, that can systematically resolve ambiguities of SVCs to arbitrate competence between verbs in parsing SVCs sentences. S-model has three major characteristics: (1) it can resolve SVCs without relying on specific types of SVCs classified by linguists; (2) it can handle long SVCs, i.e., SVCs with more than two verbs; (3) it can simultaneously determine whether a verb candidate is really acts as a verb in the sentence.

1 Introduction

In Mandarin Chinese, it is common that there are two or more verbs in a sentence without any marker indicating the relationships between them. Such peculiar construct is called Serial verb constructions (SVCs) [Li and Thompson 1981]. For example, in the sentence: "被告 希望 原告 諒解" (the defendant hope the plaintiff forgive) (The defendant hoped that the plaintiff could forgive him.), there are two verbs: "希望" (hope) and "諒解" (forgive); however, there are no such markers as subordination markers, conjunctions, preposition, or other morphological cues, which indicate the relationships between them. In developing a parser, SVCs cause considerable problems. We have designed a modified chart parser using *theta grids* information. In parsing sentences with SVCs, different verbs will *compete* in searching the chart for their own theta roles. Thus, some mechanism for arbitrating among the competing verbs for the ownership of each constituent in the chart must be designed. The theta grid chart parser is to be described in the next section.

The study of SVCs is still primitive. Most previous work [Chang and Krulee 1991] [Yeh and Lee 1992] were based on Li and Thompson's classification of SVCs [Li and Thompson 1981]. Surveying their work, we find there are some limitations. Yang [1987] and Chang et al. [Chang and Krulee 1991] dealt with only subsets of SVCs. Moreover, it is not clear how the implementations of Yang [1987], Chang et al. [Chang and Krulee 1991], and Yeh et al. [Yeh and Lee 1992] can be extended to handle *long SVCs*, i.e., those sentences containing *more than two occurrences of verbs*. It is because their work were based on the classification of SVCs, and the classification was based on two-verbs cases only. Pun [1991] claimed that his work could handle long SVCs; however, did not report how to systematically extend his method to SVCs with three or more verbs. In our model, there are three characteristics: **First**, instead of classifying SVCs into several types, we make use of a numerical *scoring function* to determine a preferred structure. It is an attempt to make the SVCs handling process more *systematic*. The information encoded in theta grids are used as bases for scoring. **Second**, it can handle *long SVCs*. **Third**, category ambiguities can be taken into consideration at the same time. Namely, we can simultaneously determine whether a verb candidate actually plays a verb or not. While in previous work, *before* the SVC handling processes are triggered, it must determine the *actual* verbs in the sentence.

This work is part of our long-term research for building a natural language front-end of a verdict understanding system. Thus, the corpora we use are judicial verdict documents from the Kaohsiung district court [Taiwan 1990a][Taiwan 1990b], which were written in a special official-document style. Thus, our analysis is based on such kind of sub-language.

2 A Theta-grid Chart Parser

Since the mechanism we propose is under the framework of a theta-grid chart parser, in this section, we introduce the parser briefly. *Thematic information* is one of the information sources that can bridge the gap between syntactic and semantic processing phases. In theta-grid theory [Tang 1992], rich thematic information is incorporated for the analysis of human languages. The idea of theta-grid theory is as follows: we use a predicate, say, a verb, as the *center* of a "grid" and, by finding the theta-roles registered in the lexical entries of this predicate, we can construct a grid formed by this predicate and then construe the sentence (or clause) spanned by this predicate. We think the theta-grid representation suitable for processing Chinese. This shares similar viewpoint with other work of designing Chinese parser which uses thematic information, such as ICG parser [Chen and Huang 1990]. To computationalize theta-grid theory, some control strategies for parsing must be implemented.

The well-known chart parser [Kay 1980], which utilizes the data structure "chart" to record the partial parsing results, is suitable for our work. Since it keeps all possible combination of constituents, it can accept sentences with missing theta roles. Thus, we designed a modified chart parser called TG-Chart parser [Lin and Soo 1993] by combining theta-grid theory and chart parser. Note that currently in our work, only the theta

grids for verbs are considered. For each verb, there are two kinds of theta roles registered: the *obligatory roles*, which must be found for this verb to construct a legal "grid"; the *optional roles*, with their appearance being optional. Take "告訴" for example, its theta roles are registered as: +[Th (Pd) Ag]; thus, two NPs *must* be found in the chart for the construction of a legal grid (From *syntactic clues*, both "Ag" and "Th" are always played by *NPs*. [Liu and Soo 1993].), while the appearance of a clause to serve as a "Pd" role is optional. A brief description of our parsing algorithm is as follows:

[Step 1] Search the sentence for positions of all "*verb candidates*". (What we call verb candidates are those words that have the verb-category as one of its syntactic categories in the dictionary.)
[Step 2] By considering all possible combination, the chart parser groups the words into syntactic constituents. Syntactic knowledge is used in this step.
[Step 3] If only one verb candidate is found in [Step 1], search the chart for constituents which can play the theta roles of this verb.
[Step 4] If more than one verb candidate are found, call *S-model* to determine the most preferred structure. S-model will be described in section 3.

3 The S-model

We design a model which utilizes scoring functions and theta-grid theory to handle the SVCs problem. This model, called S-model (an abbreviation of "SVCs handling model"), consists of four modules: a combination generator, a combination filter, a score evaluator, and a structure selector as shown in [figure 1]. We now describe these modules as follows:

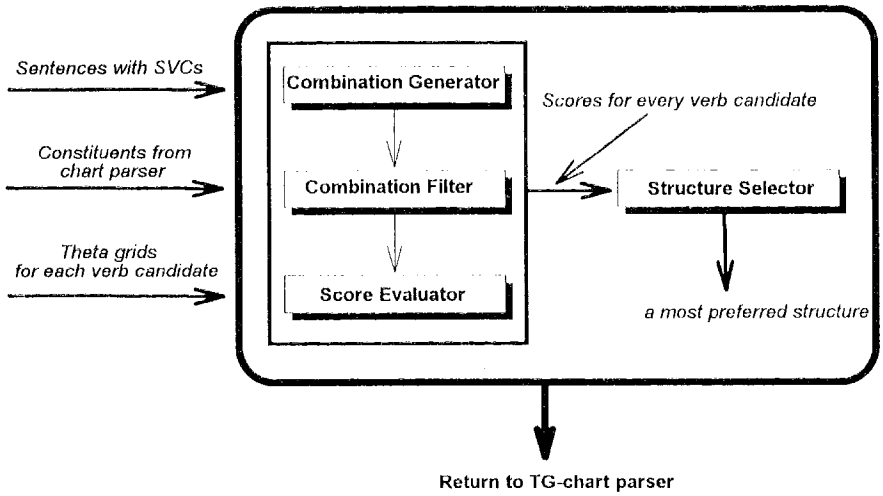


Figure 1 Modules of S-model

As we know, all verb candidates *compete* to act as verbs. The question is: "which candidates can actually act as verbs?" and, "what is their correlation?". If we can enumerate all possible combination and evaluate their scores respectively, we can determine the most preferred construction. Take the two-verb-candidates case as an example, let the two verb candidates be v1, v2, there are five combination: (1) only v1 is a verb while v2 is not, (2) only v2 is a verb, (3) both v1 and v2 are verbs, while there is not any subordination relation between them. (4) both are verbs, and v1 is subordinate to v2. (5) both are verbs, and v2 is subordinate to v1.

3.1 Combination Generator

Combination Generator consists of two submodules: *Verb-string Generator* and *Subordination-relation Tagger*. We illustrate a case with three verb candidates: Verb-string Generator generates all possible verb strings by sequentially enumerating the binary string: *001, 010, 011, 100, 101, 110, 111*. The verb string "101" represents the situation where v1 and v3 act as verbs, while v2 doesn't. And then, Subordination-relation Tagger tags these verb strings with possible subordination relations. It divides these strings into three classes according to the occurrences of 1's in the string, that is, the number of verb candidates in the sentence. These three classes are: **(1)** For the one-1 class (i.e., *001, 010, 100*), there is obviously no subordination relation. That is, there is only one possible case to consider: this candidate acts as the only verb in this sentence. **(2)** For the two-1 class (i.e., *011, 101, 110*), there are three possibilities to consider: **v1=v2**, **v1<v2**, and **v1>v2**. We follow the notations used by Pun [Pun 1991], where "v1>v2" means v2 is subordinate to v1; "v1=v2", no subordination relations exist between the two verbs. **(3)**

For the three-1 class (i.e., *111*), there are *seventeen* cases. We use abbreviated notations to represent them, where "><" is the abbreviation of "v1 > [v2<v3]", with square brackets being represented by underlines, meaning that locally v2 is subordinate to v3, and they together form a clause, which then plays a prepositional role for v1, and, for another example, "=<" is the abbreviation of "[v1=v2] < v3". These seventeen cases are: ==, =<, =<, =>, =>, <=, <=, <<, <<, <>, <>, >=, >=, ><, ><, >>, and >>. These cases are generated simply by enumerating possible combinations of these three symbols: =, <, and >. For each pair of symbols $S_1 S_2$, two combinations are possible: $S_1 S_2$ and $S_1 S_2$. Note that "==" and "==" represents the same case; thus, only a single "==" is generated. Therefore, $3 \times 3 \times 2 - 1 = 17$ cases are possible. By summarizing classes (1), (2), and (3), Combination Generator generates $C_1^3 \times 1 + C_2^3 \times 3 + C_3^3 \times 17 = 29$ cases. It is *easy* to design a routine which *systematically* enumerates these possibilities.

3.2 Combination Filter

The Combination Generator above does not take linguistic knowledge into consideration. Actually, there are some cases which will never happen in a real sentence according to syntactic constraints. Thus, it is not necessary to pass it to the score evaluator. Combination Filter is responsible for filtering out impossible cases. We illustrate three circumstances. Firstly, for "v1 > v2", the Combination Filter will check the theta grid for v1; if there is a Pd or Pe role registered in v1, it is possible, since v2 can be subordinate to v1 only if v1 also expects a prepositional role; otherwise, such a case is filtered out. The second circumstance is, when v1 has only a single syntactic category, verb, it must act as a verb in the sentence. Thus, the case that v2 acts as a verb while v1 doesn't is removed. The third circumstance regards the three-candidates situations. Combination Generator generates seventeen cases; however, under some circumstances, there are four cases which are impossible: <<, <>, <>, and >>. These circumstances happens when the main verb of the prepositional part (i.e., the part marked by a underline.) expects an *animate agent*. In such circumstances, a VP cannot be subordinate to an "event". Thus, these four will be filtered out by Combination Filter. For example, the following sentence, with the relation "<>" (i.e., 打<〔希望>參加〕), is impossible: "打 雷 希望 參加 勞工保險" (play thunder hope attend the labor insurance) (Thundering hoped to attend the labor insurance.). It is because "希望" expects an animate NP to act as its *Ag*, the VP "打 雷" thus cannot act as its *Ag* role.

There are still many linguistic knowledge and constraints which can be used by Combination Filter. However, some of them, such as the third circumstance mentioned above, are too specific and thus must be used carefully to avoid over-constraints. Therefore, how to collect and select those constraints and knowledge which are general enough is still our future concern.

The main function of Combination Filter is to improve the performance of the S-model. Note that in this paper, for the beneficiary of brevity, Combination Generator and Combination Filter are designed as two separate modules. However, Combination Filter can behave as an embedded module of Combination Generator so that it can cut off some generating branches which are impossible as early as possible. It is also our future concern.

3.3 Score Evaluator

Whenever Combination Filter passes a feasible case into Score Evaluator, the Score Evaluator utilizes a scoring function to compute the score of the input case and then, passes the evaluated score to the structure selector. We will now describe it:

3.3.1 The S-function

In our legal domain corpora, there are many occurrences of SVCs. Since our parser is based on the theta grids, in case of SVCs, different verbs will *compete* in finding their own theta roles. Thus, some mechanism for arbitrating among verbs for the ownership of each constituent in the chart must be designed. Just as what Yorick Wilks said, *language does not always allow the formation of "100%-correct" theories* [Hirst 1981]; therefore, we attempt to find a more flexible method for recognizing SVCs. We propose a scoring function to select a "preferable" construction for the sentence with SVCs. (For the "preference" notion, see [Wilks 1975] [Fass and Wilks 1983].) The scoring function is called S-function, an abbreviation for "SVCs scoring function". S-function is defined as in [figure 2], where RWR is the abbreviation of "Ratio of Words included in some phrase with Roles assigned", RRF, "Ratio of Roles Found", OBR, "OBligatory Role", and OPR, "OPtional Role" (Note that OBR and OPR indicate those roles *registered in theta grids*.):

$$Score = \frac{\sum_{\text{every verb}} Score - Per - Verb}{\text{number of verbs}} \quad (1)$$

$$Score - Per - Verb = RRF \times RWR \quad (2)$$

$$RRF = \frac{|(\text{number of OBR found}) \times k + (\text{number of OPR found})|}{Base} \quad (3)$$

$$RWR = \frac{\text{number of words included in some phrase with roles assigned}}{\text{number of words in the clause}} \quad (4)$$

$$Base = k \times (\text{number of OBR}) + (\text{number of OPR}) \quad (5)$$

Figure 2. The S-function

The score is calculated as the average value of scores obtained by each verb in the sentence (as in equation 1). For each verb, the score is estimated by two factors: *first*, the ratio of theta roles found, i.e., RRF, and, *second*, the ratio of words with roles assigned, i.e., RWR. For detailed formula, see equation (2). The relative significance between obligatory roles and optional roles is heuristically weighted by **2:1**, as in (3) and (5); thus, the value of k is set to be 2. In some cases, the verb finds many theta roles in the clause it constructs, but the words in this clause are not all assigned roles. We consider such assignment doesn't construe the real construction of the sentence. Thus, to reflect such cases, we calculate RWR by dividing the number of words which are included in some phrase with a role assigned by the total number of words in the clause (see equation 4).

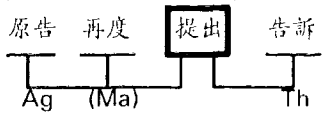
3.3.2 Illustration of S-function

Now, let's illustrate the calculation of S-function by the following examples:

Example 1: "原告 再度 提出 告訴"
(the plaintiff again file a lawsuit)
The plaintiff filed a lawsuit again.

In this example, we demonstrate *how to determine whether a verb candidate can actually act as a verb.* In [Step 1], "提出" (file) and "告訴" (tell) are both found as "verb candidates". Here "告訴" has two syntactic categories registered in its lexical entry: the verb and the noun, while "提出" has only one category, the verb. The theta grid for "提出" is +[Th Ag], "告訴", +[Th (Pd) Ag]. So, to decide whether "告訴" is treated as a verb or a noun, there are four cases to be considered:

(1) "提出" is treated as a verb, while "告訴" a noun.



In the above, "提出" enveloped by a *box* means it plays a verb. When it searches for theta roles, "原告" and "告訴" are respectively found as its Ag and Th, the two obligatory theta roles registered in its lexical entry. The score is calculated as follows: For "提出", there are two obligatory roles, so Base = 2 × 2 = 4. Moreover, in this sentence, "原告","再度","提出",and "告訴" are all assigned some roles; thus, RWR = 4/4 = 1. And then, Score-Per-Verb = {|(number of OBR found)*2 + (number of OPR found)|/Base} * RWR = {[2×2+0|/4} ×1=1. Finally, Score = 1/1 = 1.00.

(2) "提出" and "告訴" both are treated as verbs. Score = (0.5+0.4)/2 = 0.45.

(3) "提出" and "告訴" both are treated as verbs, while "告訴" is subordinate to "提出". Score = (0.375+0)/2= 0.1775

(4) "提出" and "告訴" both are treated as verbs, while "提出" is subordinate to "告訴". Score = (0.5+0.2)/2 = 0.35.

From the above discussions, **case(1)** apparently gets the highest score (1.00). So, the parsed structure in case(1) is preferable to those in the other cases. That is, in this sentence, "提出" plays the only verb, while "告訴" plays a *noun*. Therefore, *the right syntactic category for "告訴" in this sentence is determined.*

Example 2: "請求 與 被告 離婚"
(request with the defendant divorce)
Request to divorce the defendant.

In this example, we will demonstrate *how to determine the relationship between verbs.* In [Step 1], "請求" (request) and "離婚" (divorce) are both found as "verb candidates". Here "請求" and "離婚" both have two syntactic categories registered in its lexical entry: the verb and the noun. The theta grid for "請求" is +[(Th) Pe Ag], "離婚" +[Ag (Ag)]. So, there are five cases to be considered:

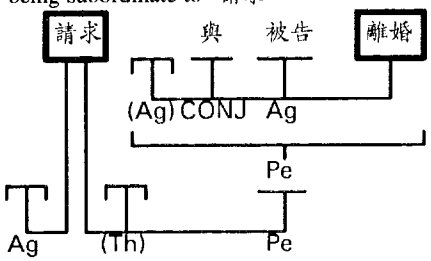
(1) "請求" is treated as the only verb, while "離婚" a noun. Score = 0.15/1 = 0.15.

(2) "離婚" is treated as a verb, while "請求" a noun.

For "離婚", Base=3. Note that although "請求" is an NP, it cannot play as Ag for "離婚". It is because it doesn't satisfy the constraint for playing as Ag: an Ag must has a feature "+animate", according to Gruber's theory that an agent must be *an entity with intentionality* [Gruber J. S. 1976]. The situation that a verb cannot find a theta role is represented by the symbol "┌┐". So, RWR = 3/4 = 0.75, and Score-Per-Verb = {[1*2+0]/3}*0.75 = 0.5. Score = 0.5/1 = 0.5.

(3) "請求" and "離婚" both are treated as verbs. Score = (0.134+0.67)/2 = 0.402.

(4) "請求" and "離婚" both are treated as verbs, with "離婚" being subordinate to "請求"



For "請求", Base=5. RWR=4/4=1. Score-Per-Verb = {[1*2+0]/5} = 0.4. For "離婚", Base=3. RWR=3/3=1. Score-Per-Verb = {[1*2+0]/3} = 0.67. Score = (0.4+0.67)/2 = 0.535.

(5) "請求" and "離婚" both are treated as verbs, with "請求" being subordinate to "離婚". Score = (0.134+0)/2 = 0.067.

From the above discussions, case(4) apparently gets the highest score (0.535). So, the parsed structure in case(4) is preferable to those in the other cases. That is, in this sentence, "請求" and "離婚" both are treated as verbs, while "離婚" is subordinate to "請求". The clause constructed by "離婚" is assigned the Pe role for "請求". Thus, this is a SVC sentence; moreover, this kind of SVC is commonly called "*sentential objects*".

3.4 Structure Selector

Structure Selector plays a final arbitrator. It collects all feasible cases and their scores. After scores of all cases are evaluated, the competition of all cases is arbitrated by Structure Selector. Structure Selector selects the case with the highest score as the most preferred one. The final result is returned to the parser.

4 Experimental Results

4.1 Results of More Sample Sentences

In table 2, we show the results of more sentences with SVC in the legal documents which are parsed by this scheme in our TG-Chart parser. The sample sentences are shown in table 1:

Table 1. Some sample sentences with SVCs

S1: 原告 訴請 被告 給予 三十萬元 (the plaintiff petition the defendant give three hundred thousand dollars) The plaintiff petitioned the defendant to give him three hundred thousand dollars.
S2: 原告 請求 被告 清償 債務 (the plaintiff request the defendant repay debts) The plaintiff requested the defendant to repay his debts.
S3: 被告 未 到 場 爭執 (the defendant didn't arrive the court argue) The defendant didn't arrive at the court to argue.
S4: 被告 突 無故 離家 出走 (the defendant suddenly causelessly left home desert his family) The defendant deserted his family suddenly and causelessly.
S5: 被告 未 返 家 與 原告 同居 (the defendant didn't return home with the plaintiff cohabit) The defendant didn't return home to cohabit with the plaintiff.
S6: 原告 聲請 訊問 證人 (the defendant petition interrogate the witness) The defendant petitioned to interrogate the witness.
S7: 被告 希望 原告 能 諒解 (the defendant hope the plaintiff can forgive) The defendant hoped that the plaintiff could forgive him.
S8: 被告 申請 參加 勞工保險 (the defendant apply attend the labor insurance) The defendant applied to attend the labor insurance.
S9: 原告 平時 待 人 很 和睦 (the plaintiff ordinarily treat people very friendly) Ordinarily, the plaintiff treats people friendly.
S10: 原告 打破 了 一個 花瓶 很 值錢 (the plaintiff break Asp one vase very valuable) The plaintiff broke a vase that was valuable.

Table 2. Results of S-function calculation for sample sentences

Sen. No.	Verb Candidates	Verb Players	Relationships	Highest Score
S1	v1: 訴請,v2: 給予	v1,v2	v1>v2	1.00
S2	v1: 請求,v2: 清償	v1,v2	v1>v2	1.00
S3	v1: 到, v2: 爭執	v1,v2	v1=v2	0.84
S4	v1: 離家,v2: 出走	v1,v2	v1=v2	1.00
S5	v1: 返, v2: 同居	v1,v2	v1=v2	0.83
S6	v1: 聲請,v2: 訊問	v1,v2	v1>v2	0.70
S7	v1: 希望,v2: 諒解	v1,v2	v1>v2	0.84
S8	v1: 申請,v2: 參加	v1,v2	v1>v2	0.75
S9	v1: 待, v2: 和睦	v1,v2	v1<v2	1.00
S10	v1: 打破,v2: 值錢	v1,v2	v1=v2	1.00

4.2 Demonstrating How to Handle Three-Verbs SVCs

Let's consider the following *three-verbs* sentence: "原告 返 家 提醒 其 妻 繳 費" (the plaintiff return home remind his wife pay fees) (The plaintiff returned home to remind his wife to pay fees.). There are *three* verbs in this sentence: 返 (return), 提醒 (remind), and 繳 (pay). At the first stage, Combination Generator generates 29 possible combination; and then, Combination Filter filters out 26 of them, and only three cases remained to be considered: "返 = 提醒 = 繳","返 = [提醒 > 繳]", and "[返 = 提醒] > 繳". Thus, Score Evaluator only needs to calculate the scores for these three remained cases. At the final stage, Structure Selector accepts the evaluated scores for these cases and selects the one with the highest score. In this example, the structure "=>" gets the highest score: 0.94; it is the correct structure for this sentence.

Consider another interesting example, "他 以為 我 嘲笑 他 是 錯的" (he think I mock he is wrong) [Pun 1991]. This sentence is *ambiguous to native speakers*, since there are two possible readings: (1) "[他 以為 我 嘲笑 他] 是 錯的" (His thinking that I mocked him is wrong.), and (2) "他 以為 [我 嘲笑 他 是 錯的]" (He thinks that I mocked him for being wrong.). In S-model, *both* these two readings get the highest score: *1.0*, and thus *both* are selected by Structure Selector as the final output. S-model doesn't attempt to select a "uniquely-correct" structure, but just selects what are *preferred*. It matches humans' behavior since even a human may not be able to tell which of these two is better.

5 Conclusion

In this paper, we propose a systematic method for analyzing SVCs. The method is based on the information offered by theta grids. Many possible correlation relations may exist between verbs, we use a numerical scoring function to determine the most preferred one. To utilize the S-function defined, we design a S-model, which consists of four modules: a combination generator, a combination filter, a score evaluator, and a structure selector, to realize it. For the examples we have tested so far, taken from the legal documents [Taiwan90a] [Taiwan90b], our mechanism always produces the correct reading.

Li and Thompson [1981] classified SVCs into four types: (1) two or more separate events (2) a VP or a clause plays the subject or direct object of another verb (3) pivotal construction (4) descriptive clauses. We usually split type (2) into two sub-types: (2)-1 sentential subjects, and (2)-2 sentential objects. Most work for handling SVCs are based on this classification. In our design of S-function, information about this classification is not used. However, in our testing sentences, it turns out that these five types are actually covered by the S-model which selects a preferred structure based on only scoring functions. For example, S5 in table 1 belongs to type (1), S9, type (2)-1, S6, type (2)-2, S2, type (3), and S10, type (4). The reason why S-model may cover the classification is due to the rich information encoded in theta grids. As an example, consider the sentence "被告 聲請 訊問 證人". (The defendant petitioned to interrogate the witness.) By Li and Thompson's classification, it belongs to the "sentential objects" type. If we can classify the sentence into the correct type, the structure "*聲請 (petition) > 訊問 (interrogate)*" will be determined. This is the idea used in most previous work. However, in S-model, we achieve the same result without relying on the classification. In S-model, since "聲請" needs a "*Pe*" which implies that it expects an "event", i.e., a "sentential object" to play the theta role, after calculating S-function, the structure where "訊問" is subordinate to "聲請" *naturally* gets the highest score and thus becomes the "*winner*". As the previous example in section 4.2, for the ambiguous sentence S-model also yields more than one highest score. We can conclude that S-model could be a very general and sound mechanism to handle SVC sentences.

Acknowledgment

This research is supported in part by National Science Council of R.O.C. under the grant NSC83-0408-E-007-008.

References

[Chang and Krulee 1991] Chao-Huang Chang and Gilbert K. Krulee, *Prediction Ambiguity in Chinese and Its Resolution*. Proc. of ICCPCOL 1991, pp.109-114.

[Chen and Huang 1990] Keh-jiann Chen and Chu-Ren Huang, Information-based Case Grammar. In Proc. of COLING-90.

[Fass and Wilks 1983] Dan Fass and Yorick Wilks, *Preference Semantics, Ill-Formedness, and Metaphor*. American Journal of Computational Linguistics, Vol. 9 (3-4), July-December 1983, pp.178-187.

[Gruber J. S. 1976] Gruber J. S., *Lexical Structures in Syntax and Semantics*, North-Holland Publishing Company. 1976.

[Hirst 1981] In Graeme Hirst, *Lecture Notes in Computer Science, Anaphora in Natural Language Understanding, A Survey*. Springer-Verlag Berlin Heidelberg. 1981.

[Kay 1980] Martin Kay. *Algorithm Schemata and Data Structures in Syntactic Processing*. In Proc. of the Nobel Symposium on Text Processing, Gothenburg, 1980.

[Li and Thompson 1981] C. N. Li and S. Thompson, *Mandarin Chinese: a Functional Reference Grammar*, University of California Press, Berkeley. 1981.

[Lin and Soo 1993] Koong H.C. Lin and Von-Wun Soo, *Toward Discourse-guided Chart Parsing for Mandarin Chinese -- A Preliminary Report*. ROCLING VI, 1993.

[Liu and Soo 1993] Rey-Long Liu and Von-Wun Soo, *An Empirical Study of Thematic Knowledge Acquisition Based on Syntactic Clues and Heuristics*. In Proceedings of ACL 1993.

[Pun 1991] K. H. Pun, *Analysis of Serial Verb Constructions in Chinese*. ICCPCOL 1991, pp.170-175. 1991.

[Taiwan 1990a] Taiwan Kaohsiung district court, *Summary of Kaohsiung District Court Criminal Verdict Documents*, Vol. 1, 1990.

[Taiwan 1990b] Taiwan Kaohsiung district court, *Summary of Kaohsiung District Court Civil Verdict Documents*, Vol. 1, 1990.

[Tang 1992] Ting-Chi Tang, *Syntax Theory and Machine Translation: Principle and Parameter Theory*. In Proc. of ROCLING V, pp.53-83. 1992.

[Yang 1987] Yiming Yang, *Combining Prediction, Syntactic Analysis and Semantic Analysis in Chinese Sentence Analysis*. IJCAI 1987, pp.679-681.

[Yeh and Lee 1992] Ching-Long Yeh and Hsi-Jian Lee, *A Lexicon-Driven Analysis of Chinese Serial Verb Constructions*. In Proc. of ROCLING V, pp.195-214. 1992.

[Wilks 1975] Yorick Wilks, *An Intelligent Analyzer and Understander of English*. In B. J. Grosz, K. S. Jones, and B. L. Webber, "Reading in Natural Language Processing", 1975.