Vietnam J Comput Sci (2018) 5:67–81
https://doi.org/10.1007/s40595-017-0102-8

REGULAR PAPER

Empirical methods for computing phrasal and sentential semantics in Vietnamese

Son The Pham[1] · **Dang Tuan Nguyen**[2]

Received: 16 October 2016 / Accepted: 4 September 2017 / Published online: 10 November 2017

Abstract The purpose of this paper is to build the sets of clauses in definite clause grammar, which can express the phrasal and sentential semantics. We build these sets based on the semantic models of computational and inferential methods for analyzing the phrasal and sentential semantics in Vietnamese Question-Answering System Model (VietQASM), and implementation techniques for computing the semantics in Pham and Nguyen (Int J Simul Syst Sci Technol 17, 2016; A computational and inferential method for analyzing the semantics of phrase and sentence in Vietnamese Question-Answering System Model (VietQASM), 2015). We also present the technical novel, which expresses the process of creating the semantic expressions according to these models. Those semantic expressions will be used to make an event of data-knowledge base in the Vietnamese Question-Answering System (VietQAS). We built three sets of clauses: (1) the first set is used to define the semantic forms of lexicons, (2) the second set is used to compute and infer the semantics for phrases, (3) the third set is used to compute and infer the semantics for sentences. We only use all these sets for building a reading-comprehension mechanism for all statement sentences in the VietQAS.

✉ Son The Pham
sonpt@uit.edu.vn

Dang Tuan Nguyen
dangnt@uit.edu.vn

[1] Department of Information Science and Engineering, University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam

[2] Faculty of Computer Science, University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam

Keywords Vietnamese Question-Answering System · Definite clause grammar · Computational semantics · Inferable semantics · Semantic representation · Semantic expression

1 Introduction

With the objective of expressing the meaning of various types of Vietnamese phrases and sentences, we proposed several models of computational and inferential semantics [1,2] as well as implementation techniques for building our VietQAS (Vietnamese Question-Answering System). In fact, we based it on the traditional and fundamental research [3–8] to conceive these applicative semantic models in [1,2], which were implemented in definite clause grammar (DCG) [4,5,7,9].

In this research, the DCG grammars strongly support the implementation processes to represent the semantics of sentences by combining between syntactic structures and semantic models in the Vietnamese language (see [1,2] for more details). In the interest of English Natural Language Understanding (NLU), the proposal of Blackburn and Bos [6,8] plays an important base for using the DCG formalism to compute and infer the semantics of a sentence to make logical forms. The logical form of a sentence shows the way and expresses the relation between the logical form and meaning of the sentence together. In the VietQAS, the results of the semantic processing were called a semantic expression which was combined with semantic forms together to create a semantic structure (a type of first-order logic). Based on the motivations stated previously, we focus on a solution for designing and building many sets of DCG clauses to represent the semantics of input sentences in the VietQAS.

The QA system is developed under a reading mechanism feature [1,2] (a part of the Reading–Answering mechanism),

which can read statement sentences and create corresponding semantic expressions. Each semantic expression represents an event of sentence [1] which is knowledge in Data-Knowledge Base of the VietQAS. The semantic expressions can be called a logical form [10–12] of NLU represented as a semantic tree form (Figs. 1, 2).

In the scope of this research, we present novel techniques for analyzing, designing, and building the DCG clauses in VietQAS to compute and infer semantics (in the sense of context-free, intention-free) of phrases, including noun phrases, quantity phrases, verb phrases, time phrases, adjective phrases, and place phrases (see Refs. [1,2] for more details). In the first step of the building process, we define one or several semantic forms for each type of lexicon (including the POS—part of speech). In the second step, we use these forms to infer the semantics and make the semantic expressions of phrases.

First, for illustration, we first express the computational and inferential semantics of some phrases and sentences in some news website, such as *"ICTNEWS"*,[1] *"VnExpress"*[2] and "*Tuổi Trẻ*"[3] (English title: *"Tuoi Tre News" or "Youth News"*). The sentences have analyzed the semantics to make a semantic expression for proving the correctness of the DCG rules (experimental proof, not prove the theory). The sentences which were collected from all three news pages could be seen a raw corpus of the VietQAS. Second, we compare semantic expressions in this research with semantic expressions in the research [13] to evaluate (qualitative proof/experiment) a diversified semantic structure in the VietQAS.

2 Proposed method

In this paper, we based our research on two main approaches to build the sets of clauses in DCG: (1) the computational and inferential method for analyzing the semantics of phrase and sentence in VietQASM [1,2]; (2) improving implementation techniques for computing the semantics [13]. That means that we will improve the DCG clauses to return semantic expressions which are similar to the results of the proposed semantic models in [1,2]. We use SWI-Prolog [14–16] and Pengines [17,18] to execute/run the DCG clauses.

We built three classification sets of DCG clauses to compute and infer the semantics for phrases, sentences as follows:

- The first classification set is used to define the specific semantic form for each lexicon belonging to verb, common noun, proper noun, number word, number, order number, adjunct... All the parts of speech and the semantic forms were also proposed in [1,2] with a semantic model form. The sets include the following Tables 1, 3, 5, 7, 9, and 12.
- The second classification set is used to compute and infer semantic form belonging to verb, common noun, proper noun, number word, number, order number, adjunct... The result of the processing is a semantic expression of noun phrase, quantity phrase, time phrase, adjective phrase, and place phrase that was also proposed in [1,2] with a semantic model. With the verb phrase, we computed and inferred the semantics of smaller phrases together to choose the most suitable semantic models. Tables 2, 4, 6, 8, 10, 11, and 13 are a DCG set to compute and infer the semantic form together to make new phrases stated previously.
- The third classification set is used to represent the semantics of sentences, which are a special case. First, the system computes and infers the semantics of phrases belonging to a noun phrase, quantity phrase, time phrase, and adjective phrase place phrase together. Second, we compute and infer the semantics between verb phrases and other phrases. Finally, we choose the most suitable semantic model of the sentence. Table 14 is a DCG set to compute and infer the semantic processing to make semantic expressions of a sentence.

The result of building computational and inferable semantics is a semantic expression that expresses the meaning of phrases and sentences. For the three classification, sets mentioned above at this research used to represent most of DCG clauses for fitting representation of illustration. When really developing the question-answering system into a specific domain, we have to define the three classification sets of DCG clauses more largely. Furthermore, we will use a few phrases to illustrate each DCG clauses and determine the semantic expressions in linguistic modeling.

2.1 Designing and building a set of DCG clauses to represent semantic expressions of phrases

In this section, we based these clauses on semantic models proposed [1,2] to design a set of new DCG clauses, which run in SWI-Prolog [14–16] and Pengines [17,18]. We only build the set of DCG clause for the following Vietnamese phrases: noun phrase (NP), quantity phrase (QuaP), place phrase (PlaceP), time phrase (TimeP), adjective Phrase (AdjP), and verb phrase (VP). The building process includes two basic steps as follows:

[1] ICTNEWS. http://ictnews.vn/. Accessed August 18, 2016.

[2] VnExpress. http://vnexpress.net. Accessed August 18, 2016.

[3] Tuổi Trẻ (Translate: Youth). http://tuoitre.vn/. Accessed August 18, 2016.

- *Step 1* Designing and building the semantic form of lexicon belonging to common noun, proper noun, proper name, common noun of place, and proper name of place… Each lexicon has a different semantic form, and thus, the DCG clauses of the semantic form are also different. In this step 1, we only present sets of DCG clauses for illustrating some specific lexicons. In the other lexicons, we defined a general semantic form to put.
- *Step 2* Building the DCG clauses for phrases to compute and infer the semantic forms of lexicon together. The mechanism processing is executed by logic programming in Prolog's engine such as SWI-Prolog [14–16] and Pengines [17,18]. We then make a semantic expression of the phrase. Each phrase has only one special grammar clause.

2.1.1 Noun phrase (NP)

We based noun phrases on semantic models proposed [1,2] to build a set of DCG clauses to represent the semantics and then create semantic expressions which can be used with other phrases to infer.

First, we define DCG clauses to represent semantic forms of a lexicon (belonging to common noun and proper noun), which is a predicate with an argument. We use the common nouns *"công ty"*, *"viễn thông"*, *"tập đoàn"*, *"bưu chính"*, *"quân đội"*, and *"chi nhánh"* (English: *"company"*, *"telecommunications"*, *"corporation"*, *"post"*, *"army"*, and *"branch"*), and proper nouns *"Viettel"* (name of a telecom company in Vietnam) and *"HSBC"* (name of a bank corporation in Vietnam) to illustrate a design processing. Table 1 shows the DCG clauses:

Table 1 DCG clauses of the semantics form of lexicon (common noun and proper noun)

No.	DCG clauses
1	`cn(X^công_ty(X)) --> [công, ty].`
2	`cn(X^viễn_thông(X)) --> [viễn, thông].`
3	`cn(X^tập_đoàn(X)) --> [tập, đoàn].`
4	`cn(X^bưu_chính(X)) --> [bưu, chính].`
5	`cn(X^quân_đội(X)) --> [quân, đội].`
6	`cn(X^chi_nhánh(X)) --> [chi, nhánh].`
7	`pn(['Viettel'], Mean) --> ['Viettel'], {Mean = [công_ty(['Viettel']), viễn_thông(['Viettel']), tập_đoàn(['Viettel'])]}.`
8	`pn(['HSBC'], Mean) --> ['HSBC'], {Mean = [công_ty(['HSBC']), ngân_hàng(['HSBC']), thế_giới(['HSBC'])]}.`

In which:

- From (1) to (6) are the DCG clauses to define a semantic form of the common noun. Argument "*X*" will receive a value of the proper noun or the proper name (if yes). We can define many new clauses of common nouns with the following form: *"cn(X^lexicon_of_common_noun(X)) --> [lexicon, of, common, noun]."*.
- Clauses (7) and (8) are semantic forms of proper noun. We define an argument *"Mean"* to represent the proper name, with the form is a list of Prolog. In addition, we can define many clauses with the following form: *"pn(['Proper_Noun'], Mean) --> ['Proper', 'Noun'], {Mean = [lex_1(['Proper', 'Noun']), lex_2(['Proper', 'Noun']), …, lex_N(['Proper', 'Noun'])]}."*. In which, the *"'Proper_Noun"'* is lexicon of a proper noun or a proper name, the *"lex_N(['Proper', 'Noun'])"* is the term in prolog for modifying the proper noun, the proper name.

Second, we define DCG clauses to compute and infer the semantic form of lexicon together. Each noun phrase has a different grammar structure. Therefore, the DCG clauses also have different structures. The results of semantic processing are semantic expressions that we design with the following form *"semNP(atom(PN), frame(Sem))"*, in which argument *"PN"* is value of a proper noun or a proper name, and argument *"Sem"* is complex semantic forms, puts in the term *"frame"*. The DCG clauses in Table 2 are used to handle and infer the semantics of noun phrase. In which:

- Clause (1) is a function to execute two processes: *"flatten(Input, Temp)"* convert *"Input"* to a non-nested version of *"Input"* list and return the result to argument *"Temp"*; *"list_to_set(Temp, Output)"* convert argument *"Temp"* to a set *"Output"*.
- Clauses from (2) to (4) are DCG clauses to represent a complex semantic expression from/with semantic forms of one, or two, or three consecutive common nouns.
- Clauses from (5) to (9) are DCG clauses to represent semantic expressions of complex noun phrases after computing semantics of the semantic form in Table 1 together.

The DCG clauses in Table 2 are fundamental clauses to handle the semantics for many simple noun phrases. In addition, we can define many DCG clauses to handle the semantics for types of noun phrases.

The process for handling the semantics of noun phrases includes two stages. The first stage is to determine the semantic forms of a lexicon in noun phrases. The second stage is to compute and infer these forms together to make semantic expressions. The semantic expressions express a detailed semantics of noun phrases, which are referred by the quantity phrase and verb phrase. As an example, we use the

Table 2 DCG clauses for handling the semantics of noun phrase

No.	DCG clauses
1	`process_to_flatten(Input, Output):- flatten(Input, Temp), list_to_set(Temp, Output).`
2	`common_noun(X, Meaning) --> cn(X^Sem),{Meaning = [Sem]}.`
3	`common_noun(X, Meaning) --> cn(X^Sem1), cn(X^Sem2), {Meaning = [Sem1, Sem2]}.`
4	`common_noun(X, Meaning) --> cn(X^Sem1), cn(X^Sem2), cn(X^Sem3), {Meaning = [Sem1, Sem2, Sem3]}.`
5	`np(SemNP) --> common_noun(PN, M), {process_to_flatten([M], Sem), SemNP = semNP(atom(PN), frame(Sem))}.`
6	`np(SemNP) --> pn(PN, Meaning), {process_to_flatten([Meaning], Sem), SemNP = semNP(atom(PN), frame(Sem))}.`
7	`np(SemNP) --> common_noun(PN, M), pn(PN, Meaning), {process_to_flatten([M, Meaning], Sem), SemNP = semNP(atom(PN),frame(Sem))}.`
8	`np(SemNP) --> common_noun(PN, M), pn(PN, Meaning), pn(PN2, Meaning2), {process_to_flatten( [M,Meaning, Meaning2], Sem), SemNP = semNP(atom(PN,PN2),frame(Sem))}.`
9	`np(SemNP) --> common_noun(PN, M1), pn(PN, Meaning), common_noun(PN,M2), {process_to_flatten( [M1, Meaning, M2], Sem), SemNP = semNP(atom(PN), frame(Sem))}.`

noun phrase *"tập toàn Viettel"* (English: *"the Viettel Corporation"*) to illustrate the mechanism for computing and inferring the semantics of a noun phrase.

At the first stage, the VietQAS reads the noun phrase *"tập toàn Viettel"* and identifies two lexicons *"tập toàn"* and *"Viettel"*. Next, we determine the semantic form of *"tập toàn"* and *"Viettel"* as *"X^tập_đoàn(X)"* (according to the 3rd clause in Table 1) and *công_ty(['Viettel'])*, *viễn_thông(['Viettel'])*, *tập_đoàn(['Viettel'])*" (according to the 7th clause in Table 1).

During the second stage, the VietQAS is based on the semantic forms to make a semantic expression as follows:

"semNP(atom(['Viettel']),
frame([công_ty(['Viettel']),
viễn_thông(['Viettel']),tập_đoàn(['Viettel'])])
)"

(according to the seventh clause in Table 2).

2.1.2 Quantity phrase (QuaP)

We based these phrases on semantic models proposed in [1, 2] to build a set of DCG clauses to handle the semantics of quantity phrase in a sentence. The results of the process are semantic expressions which express the meaning of the quantity phrase.

Table 3 DCG clauses of the semantic forms of number word, number, and order number

No.	DCG clauses
1	`num_word(một) --> [một]. %Eng.: one`
2	`num_word(nhất) --> [nhất]. % Eng.: first`
3	`num_word(hai) --> [hai]. % Eng.: two`
4	`num_word(nhì) --> [nhì]. % Eng.: second`
5	`num_word(ba) --> [ba]. %Eng.: three`
6	`num_word(tư) --> [tư]. %Eng.: fourth`
7	`num_word(năm) --> [năm]. %Eng.: five`
8	`num_word(sáu) --> [sáu]. %Eng.: six`
9	`num_word(bảy) --> [bảy]. %Eng.: seven`
10	`num_word(tám) --> [tám]. %Eng.: eight`
11	`num_word(chín) --> [chín]. %Eng.: nine`
12	`num_word(mười) --> [mười]. %Eng.: ten`
13	`num(Num) --> [N], {number(N) -> Num = N; !}.`
14	`qua(infinite) --> [với,mọi]; [mọi]; [các]; [những]; [hơn]; [tất,cả]; [hầu,hết]; [nhiều]; [đa,số]; [số,ít]; [một,số,ít]; [một,số]; [một,ít]; [một,vài]; [vài].`
15	`order(order)--> [thứ]; [lần]; [lần, thứ].`

First, we defined DCG clauses to represent semantic forms of number words, numbers, and order numbers. We chose most of the number words and order numbers to illustrate semantic form in Table 3. In which:

- Clauses from (1) to (12) are DCG clauses to describe number words in quantity phrases. We can define many DCG clauses for representing the semantic form of the number word with the following form: *"num_word(NW) --> [NW]."*, in which *"NW"* is number words. For example, a DCG clause of number word *hai mươi* (English: *"twenty"*) has the following clause *num_word(hai_mươi) --> [hai, mươi]*".
- Clause (13) is a DCG clause, which is used to identify numbers (a type of number value). For example: 1, 2, 3, 4, 5...
- Clause (14) is used to define lexicons about infinite quantities that are like the words *"many, much, all…"* (plural) in English.
- Clause (15) is used to define order lexicons such as *"the second conference"*, *"the 8th content"*. We can use them to define different orders.

Second, we define DCG clauses to compute and infer the semantic forms of number words, numbers, order numbers, and noun phrases together. The semantic expression

Table 4 DCG clauses process the semantic of quantity phrase

No.	DCG clauses
1	`quaP(SemQ) --> num(Num), np(SemNP), {SemQ = modifyQNP(SemNP, Num)}.`
2	`quaP(SemQ) --> num_word(Num), np(SemNP), {SemQ = modifyQNP(SemNP, Num)}.`
3	`quaP(SemQ) --> qua(Qua), np(SemNP), {SemQ = modifyQNP(SemNP, Qua)}.`
4	`quaP(SemQ) --> np(SemNP), order(order), num(N), {SemQ = modify_Ordinal_NP(SemNP, N)}.`

of quantity phrases is designed and built with two general forms: (1) *"modify_Ordinal_NP(SemNP, N)"* for quantity phrases describing a number, or a number word; (2)*"modifyQNP(SemNP, Num)"* for quantity phrases describing infinite quantity or order. In this, the argument *"SemNP"* of both expressions will receive the values of the semantic expressions of the noun phrases, and the argument "*N*" of the first expression will receive a value of number or number word. However, the argument *"Num"* of the second expression will receive a quantity value of a quantity phrase. DCG clauses in Table 4 are used to compute and infer the semantics. In which:

- Argument *"SemQ"* of the term *"quaP(SemQ)"* will return the semantic expressions of quantity phrase.
- Clause (1) and clause (2) are DCG clauses to express the semantics of the quantity phrase, in which clause (1) determines the number of objects (by number) in the phrase, and clause (2) determines the number of objects (by number word) in the phrase.
- Clause (3) is used to compute and determine the infinite number of objects.
- Clause (4) is used to compute and determine an order of any event in the quantity phrase.

As an example, we use the quantity phrase *"chi nhánh thế 3"* and *"nhiếu công ty"* (English: *"the third/3rd branch"* and *"many companies"*) to illustrate the mechanism for computing and inferring the semantics of the quantity phrase.

After the two quantity phrases *"chi nhánh thứ 3"* and *"nhiếu công ty"* are read, there are many lexicons: "*chi nhánh*", "*thứ*", "3", "*nhiều*", and "*công ty*". We determine the following semantic forms: the lexicon *"chi nhánh"* with semantic form *"X^chi_nhánh(X)"* (according to the 6th clause in Table 1), the word *"thế"* with a semantic value *"order"* (according to the 15th clause in Table 3), the number "3" with a semantic value "3" (according to the 13th clause in Table 3), the word *"nhiều"* with a semantic value *"infinite"* (according to the 14th clause in Table 3), and the common noun *"công ty"* with semantic form *"X^công_ty(X)"* (according to the 1st clause in Table 1).

Table 5 Semantic form of adjective lexicons

No.	DCG clauses
1	`adj([tốt],[phẩm,chất])-->[tốt]. %Eng.: good`
2	`adj([đỏ],[màu,sắc])-->[đỏ]. %Eng.: red`
3	`adj([lớn],[kích,thước])-->[lớn]. %Eng.: big`
4	`adj([tròn],[hình,dáng])-->[tròn]. %Eng.: circular`
5	`adj([ồn_ào],[âm,thanh])-->[ồn, ào]. %Eng.: noisy`
6	`adj([chua],[hương,vị])-->[chua]. %Eng.: sour`
7	`adj([chậm],[mức,độ])-->[chậm]. %Eng.: slow`
8	`adj([hấp_dẫn],[unidentified])-->[hấp,dẫn]. %Eng.: interested`

After having examined the semantic forms, the VietQAS will compute and infer semantics of the semantic forms together to make a semantic expression of quantity phrases. For the quantity phrase *"chi nhánh thứ 3"* which has a semantic expression as follows: *"modify_Ordinal_NP (semNP(atom(X), frame(X^chi_nhánh(X))), 3)"* (the 5th clause in Table 2 determines the semantics of noun phrase, according to the 4th clause in Table 4 which computes and infers the semantics to make a semantic expression). Similarly, the quantity phrase *"nhiều công ty"* has a semantic expression as follows: *"modifyQNP(semNP(atom(X), frame(X^công_ty(X))), infinite)"*.

2.1.3 Adjective phrase (AdjP)

Adjective phrases are based on the previous semantics model proposed in [1,2]. We define DCG clauses to compute the semantics of adjective phrases according to the proposed models. The result of computing the semantics is a semantic expression, which expresses the meaning of the adjective phrase. In some cases, the semantics of an adjective phrase modifies the noun phrase and has complex forms of semantic expressions.

First, based on the limitation of the types of the adjective in [1,2], we designed and built with defining DCG clauses to represent semantic forms of adjective lexicons. We choose most of the adjectives to illustrate semantic forms in Table 5 (translated to English from 1st clause to 8th clause). In which:

- Clause (1) to clause (7) are used to define adjective lexicons, in which the first argument of each clause is an adjective lexicon and the second argument is a type of adjective.

Table 6 DCG clauses express the semantic of an adjective phrase

No.	DCG clauses
1	`wordAdj -->[rất];[quá];[cực, kỳ]; [thật];[vô, cùng].`
2	`adjP(SemAdjP) --> adj(Word, Type), {SemAdjP = semAdjP(Word, Type)}.`
3	`adjP(SemAdjP) -->wordAdj, adj(Word, Type), {SemAdjP = semAdjP(Word, Type)}.`
4	`np(SemNP) --> common_noun(PN, M1), adjP(SemAdjP), {process_to_flatten([M1], Sem), SemNP = semNP(atom(PN), frame(Sem), modifyAdjP_NP(SemAdjP))}.`
5	`np(SemNP) --> common_noun(PN, M1), pn(PN, Meaning), adjP(SemAdjP), {process_to_flatten([M1, Meaning],Sem), SemNP=semNP(atom(PN), frame(Sem), modifyAdjP_NP(SemAdjP))}.`

- Adjectives do not determine a type of adjective; the second argument value of each clause received the value *"[unidentified]"*.
- We can further define semantic form of adjective lexicons with the following form *"adj([lexicon], [type, of, adjective]) --> [lexicon]."*, in which the first *"[lexicon]"* is an adjective (Ex: *"đẹp"*, "*xấu*"... English: *"beautiful"*, *"bad"* ...), and the second argument *"[type, of, adjective]"* is a type of adjective (Ex: *phẩm chất* , *"hình dáng"* ... English: *"quality"*, *"shape"*...).

Second, we define DCG clauses to express the semantics of adjective phrases. The semantic expression is designed with the following general form *"semAdjP(Word, Type)"*, in which *"Word"* is a value of adjective and *"Type"* is a type of adjective.

An adjective phrase follows and modifies the noun phrases. The semantic expression is designed with the following form *"semNP(atom(PN), frame(Sem), modifyAdjP_NP (SemAdjP))"* where the argument *"PN"* is the value of the proper noun, the argument *"Sem"* is the complex semantic forms, put in *"frame"* and argument *"SemAdjP"* gets the semantic expression of the adjective phrase, put in *"modifyAdjP_NP"*. The DCG clauses are used to process the semantic in Table 6. In which:

- Clause (1) is a DCG clause to determine the adjunct/adverb *"rất, quá, cực kỳ, thật, vô cùng"* (English: *very, too, so*), which in front of an adjective is an identification mark of the adjective.
- Clause (2) and clause (3) are DCG clauses to compute and infer the semantics of adjective phrases.
- Clause (4) and clause (5) are DCG clauses to compute and infer the semantics of adjective phrases, which are behind noun phrases and modify the noun phrase.

In addition, we use the adjective phrase *"những công ty lớn"* (English: *"many big companies"*) to illustrate the mechanism for computing and inferring the semantics of the adjective phrase. This is a complex phrase between the noun phrase and the adjective phrase.

After the VietQAS read each lexicon in the phrase *"những công ty lớn"*, including three lexicons *"những"*, *"công ty"*, "lớn", we determine the semantic form as *"infinite"* (according to the 14th clause in Table 3), *"X^công_ty(X)"* (according to the 1st clause in Table 1), and *"[lớn], [kích, thước]"* (according to the 3 clause in Table 5). Next, the semantic forms compute and infer the semantics together to make a semantic expression as follows:

"semNP(atom(PN),
frame(Sem), modifyAdjP_NP(SemAdjP)
)"

2.1.4 Place phrase (PlaceP)

Place phrases describe positions or locations in sentences. We based these phrases on the semantic models of the place phrase, which are proposed in [1,2] to design DCG clauses for computing the semantics of place phrases.

We define a lexicon set to represent a semantic form, in which: common nouns describe positions and locations, and proper names describe positions and locations. Most of the common nouns and the proper names are used to design semantic forms: *"thành phố"*, "*thủ đô*", "*tỉnh* " (English: *"city"*, *"capital"*, *"province"*), *"Hà Nội"* (capital of Vietnam), and *"Cần Thơ"* (a province in Mekong Delta of Vietnam). The DCG clauses in Table 7 are defined for the semantic forms. In which:

Table 7 DCG clauses represent semantic forms of positions, locations

No.	DCG clauses
1	`cn_place(X^thành_phố(X)) --> [thành, phố].`
2	`cn_place(X^thủ_đô(X)) --> [thủ, đô].`
3	`cn_place(X^tỉnh(X)) --> [tỉnh].`
4	`pn_place(['Hà_Nội'], Mean) --> ['Hà', 'Nội'],{Mean = [thủ_đô('Hà_Nội'), thành_phố('Hà_Nội')]}.`
5	`pn_place(['Cần_Thơ'], Mean) --> ['Cần', 'Thơ'],{Mean = [tỉnh('Cần_Thơ'), thành_phố('Cần_Thơ')]}.`

- Clause (1), clause (2), and clause (3) are used to define semantic forms for common nouns, which describe positions and locations in phrases. We can define the semantic form of common nouns with the following form: *"cn_place(X^common_noun(X))-->[common, noun]."*, in which *"[common, noun]"* is a common noun to describe positions and locations. For example, the common noun *"quốc gia"*, *"quận"*, *"phường"*… (English: *"nation"*, *"district"*, *"ward"*…) which describe positions and locations in a sentence are represented the semantic form as follows: *"cn_place(X^quốc_gia(X)) --> [quốc, gia]."*, *"cn_place(X^quận(X)) --> [quận]."*, and *"cn_place(X^phướng(X)) --> [phướng]."*...
- Clause (4) and clause (5) are used to define semantic form for proper names, which are place-names. Each place-name is modified by the semantic forms of common nouns, which describe positions and locations. Furthermore, we can define the semantic form of place-names with the following form: *"pn_place(['PN'], Mean) --> ['PN'], {Mean = [lex1('PN'), lex2('PN'), …, lexN('PN')]}."*, in which *"PN"* is Place-name and *"lex1('PN'), lex2('PN'), …, lexN('PN')"* is a set of the common nouns which describe positions and locations in phrases. For example, the place-name *"Đà Nẵng"* (a city of Central Vietnam) is represented by the semantic form: *"pn_place(['Đà_Nẵng'], Mean) --> ['Đà', 'Nẵng'], {Mean = [thành_phố('Đà_Nẵng')]}."*.

Next, we build DCG clauses to compute and infer the semantics for place phrases. The semantic forms in Table 7 infer semantics together to make the semantic expression that is designed and built with the following general form: *"semPlaceP(frame(Meaning, CN_Place), PN_Place)"*, in which *"frame(Meaning, CN_Place)"* is a term, *"Meaning"* with a set of common nouns describes positions and locations for *"CN_Place"*. DCG clauses in the below Table 8 are used to compute and infer the semantics of place phrases. In which:

- Clause (1) is a DCG clause to determine the adjunct *"tại, ở, ở tại"* in Vietnamese, which is the same as the prepositions *"in, at,..."* in English. It is in front the place phrase.
- Clause (2), clause (3), and clause (4) are DCG clauses to compute and infer the semantics of place phrases together.
- The semantic expression of the place phrase is returned by the argument *"SemPlaceP"*.

As an example, we use the place phrase *"tại thành phố Cần Thơ"* (English: *"in/at Can Tho city"*) to illustrate the mechanism for computing and inferring the semantics of the place phrase.

Table 8 DCG clauses are used to compute and infer the semantics of place phrase

No.	DCG clauses
1	`loc_word --> [tại]; [ở]; [ở, tại]; [].`
2	`placeP(SemPlaceP) --> loc_word, cn_place(CN_Place), {SemPlaceP = semPlaceP(frame(CN_Place),_)}.`
3	`placeP(SemPlaceP) --> loc_word, pn_place(PN_Place, Meaning),{SemPlaceP = semPlaceP(frame(Meaning), PN_Place)}.`
4	`placeP(SemPlaceP) --> loc_word, cn_place(PN_Place^CN_Place), pn_place(PN_Place, Meaning), {SemPlaceP =semPlaceP( frame(Meaning,CN_Place),PN_Place)}.`

The place phrase *"tại thành phố Cần Thơ"* includes two lexicons *"thành phố"* and *"Cần Thơ"*. After reading the lexicons successfully, the VietQAS will determine the semantic form for each lexicon word as *"X^thành_phố(X)"* (according to the 1st DCG clause in Table 7) and *" [tỉnh('Cần_Thơ'), thành_phố('Cần_Thơ')]"* (according to the 5th DCG clause in Table 7).

When it has the semantic form, the VietQAS continue to compute and infer semantics of the semantic forms together to choose a suitable semantic model and make a semantic expression as follows:

"semPlaceP(
frame(thành_phố('Cần_Thơ')]), [tỉnh('Cần_Thơ'),
thành_phố('Cần_Thơ')]), ['Cần_Thơ']
)" (according to the 4th DCG clause in Table 8).

2.1.5 Time phrase (TimeP)

Time phrases describe time or an object related to time in a sentence. There are two types of time phrases: (1) time phrase with date form in a year, (2) quantitative time in context. Based on the previous semantic model proposed in [1,2], we first designed and defined DCG clauses to represent semantic forms of day, month, year, and unit of time in the time phrase with the semantic forms in Table 9. In which:

- Clause (1) defines numbers which present a quantification of a day, or a month, or a year.
- Clause (2), clause (3), and clause (4) define day, month, and year, respectively, which are types of numbers. With the year, we limit a scope of year values from 1900 to 2100.
- Clause (5), clause (6), and clause (7) define the unit of day, month, and year, respectively.

For the first case, we define DCG clauses to handle the semantics of time phrases with date forms in years. To solve

Table 9 DCG clauses define number and unit of day–month–year

No.	DCG clauses
1	`num(Num)--> [N], {number(N) -> Num = N}.`
2	`num_day(Num)--> [Number], {number(Number)-> (Num=Number,Number>=1,Number<32)}.`
3	`num_month(Num)-->[Number],{number(Number)-> (Num=Number,Number>=1,Number<13)}.`
4	`num_year(Num)--> [Number],{number(Number)-> (Num=Number,Number>=1900,Number<2100)}.`
5	`unit_day([ngày])--> [ngày].`
6	`unit_month([tháng])--> [tháng].`
7	`unit_year([năm])--> [năm].`

Table 10 DCG clauses are used for computing and inferring the semantics of time phrase with date form in year

No.	DCG clauses
1	`word_time--> []; [vào];[trong];[lúc];[khoảng];[đúng].`
2	`timeP(SemTimeP) --> word_time, unit_day(Dd), num_day(Nday), unit_month(Mm),num_month(Nmonth), unit_year(Yy),num_year(Nyear), {SemTimeP = semTimeP( date(Nday, Nmonth, Nyear))}.`
3	`timeP(SemTimeP) --> word_time, unit_day(Dd),num_day(Nday), unit_month(Mm),num_month(Nmonth), {SemTimeP = semTimeP( date(Nday, Nmonth, Nyear))}.`
4	`timeP(SemTimeP) --> word_time, unit_month(Mm),num_month(Nmonth), unit_year(Yy),num_year(Nyear), {SemTimeP = semTimeP( date(Nday, Nmonth, Nyear))}.`
5	`timeP(SemTimeP) --> word_time, unit_year(Yy),num_year(Nyear), {SemTimeP = semTimeP( date(Nday, Nmonth, Nyear))}.`
6	`timeP(SemTimeP) -->word_time, unit_month(Mm),num_month(Nmonth), {SemTimeP = semTimeP( date(Nday, Nmonth, Nyear), _)}.`

this problem, we based the clauses on the method of computational semantics model which is proposed in [1,2]. In Table 10, the DCG clauses are built to analyze and compute the semantics with the structure of time in [1,2]. The semantic expression is designed and built with the following form: *"semTimeP(date(Nday, Nmonth, Nyear))"*, in which the argument *"Nday"*, "*Nmonth*", *"Nyear"* will receive the value of day–month–year (number value) in years. The DCG clauses in Table 10 are clauses to compute and infer the semantic. In which:

Table 11 DCG clauses are used to compute and infer the semantics of time phrase with quantitative time in years

No.	DCG clauses
1	`word_time_ --> []; [trong];[khoảng];[vào, khoảng];[đúng].`
2	`timeP(SemTimeP) --> word_time_, num(D), unit_year(_), {SemTimeP = date(ngày(D),tháng(M), năm(Y))}.`
3	`timeP(SemTimeP) --> word_time_, num(M), unit_month(_), {SemTimeP = date(ngày(D), tháng(M), năm(Y))}.`
4	`timeP(SemTimeP) --> word_time_, num(Y), unit_day(_), {SemTimeP = date(ngày(D), tháng(M), năm(Y))}.`

- Results of semantic expressions after computing and inferring are returned by the argument *"SemTimeP"*.
- Clause (1) defines the adjunct "*vào, trong, lúc, khoảng,...*" in Vietnamese, which is the same as the preposition *"in, on, at,..."* in front of day, month, and year in English.
- Clause (2) to clause (6) are used to compute and infer the semantics of date and year together.

The second case is time phrases with quantitative time in years. We based this on the proposed semantic model [1,2] and improved the computing technics in [13] to define DCG clauses for computing the semantics. The result of semantic computing is a semantic expression which is designed and built with the following general form *"semTimeP(date(ngày(D), tháng(M), năm(Y)))"*, in which, the arguments "*D*", "*M*", "*Y*" will receive quantitative values of time such as day number, month number, and year number. The DCG clauses are defined to compute and infer the semantics in Table 11. In which:

- The result of the semantic expression is received by the argument *"SemTimeP"*.
- Clause (1) defines the adjunct "*trong, khoảng, vào khoảng, đúng* " in Vietnamese, which is the same as the preposition *"about, on..."* in English.
- Clause (2), clause (3), and clause (4) are DCG clauses to compute and infer the semantics. The clauses can quantify the day, month, year in the quantity phrase.

As an example, we use the time phrase *"vào tháng 6 năm 2016"* and "*khoảng 10 năm*" (English: *"in June 2016"* and *"about 10 years"*) to illustrate the mechanism for computing and inferring semantics of time phrases.

The time phrases *"vào tháng 6 năm 2016"* and "*khoảng 10 năm*" include lexicons with the semantic form

Table 12 DCG clauses are used to represent the semantic for types of verbs

No.	DCG clauses
1	`v(Y^X^mở(X, Y)) -->[mở].`
2	`v(Y^X^chuẩn_bị(X, Y)) -->[chuẩn, bị].`
3	`passive(bị) --> [bị].`
4	`passive(được) --> [được].`
5	`v(SemV, vpPro(_, pass(P))) --> passive(P), v(Sem), {Y^X^V = Sem, SemV = X^Y^V}.`
6	`v(Sem, vpPro(adv(Value, Type), _)) --> adv(Value, Type), v(Sem).`
7	`v(SemV,vpPro(adv(Value, Type),pass(P)))--> adv(Value, Type), passive(P), v(Sem), {Y^X^V = Sem, SemV = X^Y^V}.`

Table 13 DCG clauses are used to compute the semantics for verb phrase

No.	DCG clauses
1	`vp(SemV, VP_Pro) --> v(SemNP^SemV, VP_Pro), np(SemNP).`
2	`vp(SemV, VP_Pro) --> v(SemNP^SemV, VP_Pro), [bởi], np(SemNP).`
3	`vp(SemV, VP_Pro) --> v(SemNP^SemV,VP_Pro), [bằng], np(SemNP).`
4	`vp(SemV, VP_Pro) --> v(SemV2^SemV, VP_Pro), v(SemNP^SemV2,_), np(SemNP).`
5	`sub_clause(Sem) --> np(NP), vp(NP^VP, _), {Sem = VP}.`
6	`vp(SemV, VP_Pro) --> v(Sem^SemV, VP_Pro), sub_ clause(Sem).`

or the semantic value: *"tháng"* with *"[tháng]"* (according to the 6th clause in Table 9), "6" with "6" (according to the 3rd clause in Table 9), *"năm"* with *"[năm]"* (according to the 7th clause in Table 9), *"2016"* with "2016" (according to the 4th clause in Table 9), *and* "10" with "10" (according to the 1st clause in Table 9).

After determining the semantic forms of lexicons in time phrase, the VietQAS will compute and infer the semantic forms of lexicons together and choose the best suitable semantic model to make a semantic expression. According to the 4th clause in Table 10, the semantic expression of the time phrase *"vào tháng 6 năm 2016"* is: *"semTimeP(date(_, 6, 2016))"*. Similarly, according to the 2nd clause in Table 11, the time phrase "*khoảng 10 năm* " has the semantic form: *"date(ngày(_), tháng(_), năm(10))"*.

2.1.6 Verb phrase (VP)

The verb phrase is a complex phrase that combines a verb phrase and a noun phrase (this noun phrase must be behind the main verb in a sentence). In the semantic processing for verb phrases, the trouble is how to link the semantic model of a verb and the semantic model of a noun phrase together. To do this, we based the clause on the proposed model [1,2] and innovated the implementation techniques in [13]. The result of computing the semantics is a semantic expression, which expresses a semantic processing (semantic properties) [1,2] of adverbs and passives in verbs, and relates the semantic model link of verb and noun phrases. DCG clauses in Table 12 are designed and defined to present semantic forms of verbs, attributive to the verb. In which:

- Clause (1) and clause (2) are DCG clauses used to define a semantic form for a verb. We can further define DCG clauses for different verbs with the following clause *"v(Y^X^lexicon_of_verb(X, Y)) --> [lexicon, of, verb]."*, in which *"[lexicon, of, verb]"* is a lexicon of the verb (list type of Prolog) and the two arguments "*X*", "*Y*" will receive values of the semantic expression of noun phrases. For example, the verb *"thành lập"*, *"thông báo"*... (English: *"establish"*, *"announce"*...) have the semantic form as follows: *"v(Y^X^thành_lập(X, Y)) --> [thành, lập]."*, *"v(Y^X^thông_báo(X, Y)) --> [thông, báo]."*.
- Clause (3) and clause (4) are used to define the word "*bị*" and *"được"* to process semantics of passive verbs. (The word "*bị*" and *"được"* are in front of an active verb to form a passive verb).
- Clause (5) is used to compute the semantics for passive verbs.
- Clause (6) is used to compute the semantics for adverbs which are in front of the verb.
- Clause (7) is used to compute the semantics for verbs which combine passive verbs and adverbs.

Next, we based this phrase on the proposed semantic models to compute the semantics for verb phrases. Results of the semantic computing processing include an expression of the semantic form of verb (the result is returned by the argument *"SemV"* of *"vp(SemV, VP_Pro)"* from the 1st clause to 4th clause, and clause 6 in Table 13) and a expression of an attributive value of the verb (result is returned by the argument *"VP_Pro"* of*"vp(SemV, VP_Pro)"* from the 1st clause to 4th clause, and clause 6 in the Table 13). DCG clauses in Table 13 are defined to determine the semantic form and attributive value of the verb phrase in a sentence. In which:

- Clause (1) is used to compute and infer the semantics for verb phrase.
- Clause (2) and (3) are used to compute the semantic of verb phrase, which expresses a passive meaning of a verb in the verb phrases.

- Clause (4) is defined to compute the semantics for two consecutive verbs in a verb phrase.
- Clause (5) and clause (6) are used to compute the semantics for complex verb phrases.

Notice that the fist argument *"SemV"* receives a value of the semantic form of a verb and the second argument *"VP_Pro"* receives an attributive value (active or passive verb; determining the attributive existence of adverb, and the adverb modifies verb) of the verb phrase. The full expression has the following form: *"vpPro(adv(Value, Type), pass(P))"*.

As an example, we used the verb phrase "*được mở bởi tập toàn Viettel* toàn Viettel" (English: *"be opened by the Viettel Corporation"*) to illustrate the mechanism for computing and inferring the semantics of the verb phrase. The structure of the verb phrase, which includes a verb phrase and a noun phrase, is the passive voice. Semantic processing of verb phrases includes two stages.

The first stage is to determine the semantic expressions of a noun phrase which follow a verb in a sentence. According to the above section, the semantic expression of *"tập toàn Viettel"* is

```
"semNP(
    atom(['Viettel']),
    frame([công_ty(['Viettel']),viễn_thông(['Viettel']),
        tập_đoàn(['Viettel'])])
)".
```

Next, we determine the semantic form and the attributive value of verb. The semantic form of verb "*mở*" is "*Y^X^mở(X, Y)*" (according to 1st DCG clause in Table 12). With the passive voice, the attributive value of verb is "*vpPro(_, pass([được]))*" (according to the 4th–5th DCG clause in Table 12).

The second stage is to compute and infer the semantic forms from the first stage and the semantic expression of noun phrase together to make a semantic expression of the verb phrase. According to the 2nd DCG clause in Table 13, the semantic expression of "*được mở bởi tập toàn Viettel*" is as follows:

```
"semVP(
    semVerb(X^mở(X,
        semNP(atom(['Viettel']),
            frame([công_ty(['Viettel']),
                viễn_thông(['Viettel']),
                tập_đoàn(['Viettel'])])))),
    vpPro(_, pass([được]))
)"
```

2.2 Building a set of DCG clauses for computing and inferring the semantics of sentence

In this section, we based on semantic expressions in the above Sect. [2.1], the semantic models in [1,2], and techniques in [13] to build a set of DCG clauses to compute and infer the semantics of a sentence.

The semantic expression of a sentence is designed as the following form: *"sem(semVP(semVerb(VP), VP_Pro), Comp)"*. This is the best general model, in which the term *"semVP(semVerb(VP), VP_Pro)"* is a semantic expression of the verb phrase, and the argument *"Comp"* is a semantic expression of the component (including the place phrase, adjective phrase, and time phrase).

The DCG clauses in Table 14 were defined to compute and infer the semantics of a place phrase, adjective phrase, time phrase, noun phrase, and verb phrase together. In which:

- From clause (1) to clause (4) are DCG clauses, which are used to compute the semantics for adjective phrases, place phrases, and time phrases. We designed the form of semantic expression to represent the semantics with the following form *"components(SemAdjP, SemPlaceP, SemTimeP)"*, in which, the argument *"SemAdjP"* gets semantic expressions of adjective phrases, the argument *"SemPlaceP"* gets semantic expressions of place phrases, and the argument *"SemTimeP"* gets semantic expressions of time phrases.
- Clause (5) is a DCG clause to compute and infer the semantics of noun phrases (the noun phrase in front of the main verb in a sentence), verb phrases, and component together. Results of computational and inferable semantics are semantic expressions, in which the argument *"SemS"* gets.

As an example, we use the sentence "*Viettel chuẩn bị mở chi nhánh thứ 3 tại thành phố Cần Thơ vào tháng 8 năm 2016"* (English: *"Viettel prepares to open the third branch in Can Tho city in August 2016"*) to particularly illustrate the clause sets of the Definite Clause Grammar of phrases and sentences. The phrase *"Viettel"* (NP), *"chi nhánh thế 3"* (QuaP), "*tại thành phố Cần Thơ*" (PlaceP), *"vào tháng 8 năm 2016"* (TimeP), which are similar to the illustratable examples of the phrases above, have the semantic expressions in Table 15.

According to the 3rd DCG clause in Table 14, the VietQAS determines the semantic expression of components of both place phrases and time phrases in sentences as follows:

```
"components(_,
    semPlaceP(
        frame(thành_phố('Cần_Thơ')]),[tỉnh('Cần_Thơ'),
            thành_phố('Cần_Thơ')]),['Cần_Thơ']),
    semTimeP(date(_, 8, 2016))
)"
```

which we provisionally call the semantic expression **"A"**.

Next, with two consecutive verbs *chuẩn bị* and "*mở*", the VietQAS, according to the 2nd and 1st in Table 12, determine their semantic form as follows: "*Y1^X1^chuẩn_bị(X1, Y1)*" and "*Y2^X2^mở(X2, Y2)*". This combines with the semantic expression of the noun phrase (No. 2 in Table 15), which follow the verbs (according to the 4th DCG clause in Table 13) to compute and infer together. After that, making a semantic expression for the verb phrase "*chuẩn bị mở chi nhánh thứ 3*" (English: *"prepare to open the third branch"*) as follows:

```
"semVP(
    semVerb(X1^chuẩn_bị(X1, X2^mở(X2,
        modify_Ordinal_NP( semNP(atom(X),
            frame(X^chi_nhánh(X))), 3))))),
    vpPro(_, _)
)"
```

which is the semantic expression of the noun phrase in front of the verb to make the semantic expression of *"Viettel chuẩn bị mở chi nhánh thứ 3*" (English: *"Viettel prepares to open the third branch"*) (according to the 5th in Table 14). We provisionally call this the semantic expression **"B"**.

Next, according to the 5th in Table 14, the VietQAS also computes and infers from the two semantic expressions **"A"** and **"B"** to make a semantic form of a sentence as follows:

```
"sem(
    semVP(
        semVerb(X1^chuẩn_bị(X1, X2^mở(X2,
            modify_Ordinal_NP( semNP(atom(X),
                frame(X^chi_nhánh(X))), 3))))),
        vpPro(_, _)
    ),
    components(_,
        semPlaceP(
            frame(thành_phố('Cần_Thơ')]),
                [tỉnh('Cần_Thơ'),
                thành_phố('Cần_Thơ')]),
            ['Cần_Thơ']),
        semTimeP(date(_, 8, 2016)))
)"
```

which is the semantic expression of the sentence *"Viettel chuẩn bị mở chi nhánh thứ 3 tại thành phố Cần Thơ vào tháng 8 năm 2016"*.

Table 14 DCG clauses are used to compute and infer the semantics for sentence

No.	DCG clauses
1	`components(Comp) --> placeP(SemPlaceP),{Comp = components( SemAdjP, SemPlaceP, SemTimeP)}.`
2	`components(Comp) --> timeP(SemTimeP), {Comp = components( SemAdjP, SemPlaceP, SemTimeP)}.`
3	`components(Comp) --> placeP(SemPlaceP), timeP(SemTimeP), {Comp = components( SemAdjP, SemPlaceP, SemTimeP)}.`
4	`components(Comp) --> adjP(SemAdjP), placeP(SemPlaceP), timeP(SemTimeP), {Comp = components( SemAdjP,SemPlaceP,SemTimeP)}.`
5	`s(SemS) --> np(NP),vp(NP^VP,VP_Pro), components(Comp), {SemVP= semVP(semVerb(VP),VP_Pro), SemS = sem(SemVP,Comp)}.`

Table 15 Summary of semantic expressions of phrase

No.	Phrase	Semantic expression
1	*"Viettel"* (NP)	*"semNP(atom(['Viettel']), frame([công_ty(['Viettel']), viễn_thông(['Viettel']), tập_đoàn(['Viettel'])]))"*
2	*"chi nhánh thứ 3"* (QuaP)	*"modify_Ordinal_NP(semNP(atom(X), frame(X^chi_nhánh(X))), 3)"*
3	*"tại thành phố Cần Thơ"* (PlaceP)	*"semPlaceP(frame(thành_phố('Cần_Thơ')]), [tỉnh('Cần_Thơ'), thành_phố('Cần_Thơ')]), ['Cần_Thơ'])"*
4	*"vào tháng 8 năm 2016"* (TimeP)	*"semTimeP(date(_, 8, 2016))"*

3 Results and discussion

To evaluate the result of analyzing a sentence (not only a statement), we used the Vietnamese news titles (cf. Sect. 1) which were collected from the news website. In the millions of news titles, we carefully selected 500 general sentences of research cases to build the data set to evaluate the possibility of the semantic processing. That means that we randomly selected a large number of titles, which were simple sentences. Then, we removed the wrong syntax (ungrammatical sentences). Finally, we retained 500 sentences that our current model could handle them. At the present, the experiments in the field of Natural Language Understanding (NLU), most of the authors together chose a certain number

Table 16 Detail results of experiments

No.	Data Set	Number of testing sentences (statement)	Precision of testing results (%)
1	Sub data set 1	100	90
2	Sub data set 2	100	93
3	Sub data set 3	100	92
4	Sub data set 4	100	89
5	Sub data set 5	100	92

of sentences for evaluating the ability of sentence understanding [19–22]. Therefore, in this paper, we used 500 sentences to evaluate possible. The sentences have a suitable structure for above presentation. The system automatically reads the sentences and calculates the precision.

Analytical results of this semantic processing in this research have two states: the system successfully (the first state) or unsuccessfully (the second state—fail) computes the semantics of sentences to make the semantic expression. The precision of the testing result is rated between the number of successful sentences and the date set.

In the present empirical methods, we defined on the DCG clauses (about 3000 DCG clauses) for computing phrasal and sentential semantics in the Vietnamese question-answering system on the domain of Business, Economy, Science-Technology [1]. In addition, the lexicons of VietQAS are about 1500 words (which defined semantic forms). The input sentence types in data set are about 500 grammar structures which cover all cases of the experiment for understanding natural language.

The data set, including 500 sentences, was split into five sub data sets, each having 100 items, which were tested with the proposed methods. The purpose of splitting the data set is to prove the stability of our system. The system automatically reads each sentence in the data set, then calculating the precision. In SWI-Prolog (see Refs. [9,14,18] for more details to call the DCG clauses in Prolog), we used the following clause *"?- s(SemS, InputS, [])."* (according to the 5th in Table 14) to execute sentences, in which argument *"SemS"* returns semantic expressions of the input sentence and argument *"InputS"* is input sentence. Results of all experiments were presented in the Table 16.

The average result of the experiments on the five data sets for computing and inferring the semantics to make semantic expressions (logical form) successfully is around 91.20%. With the natural language understanding problem, this result is accepted and satisfied with the system to understand natural language sentences.

In addition, in the result of this research, we similarly used some phrases and sentences to compare the semantic expressions in [13] with the semantic expressions in this research. The result shows that: (1) the semantic expression of phrases in this research is far more detailed than the semantic expression of phrases in [13]; (2) the semantic expression of sentences in this research is more logical than the semantic expression of phrases in [13].

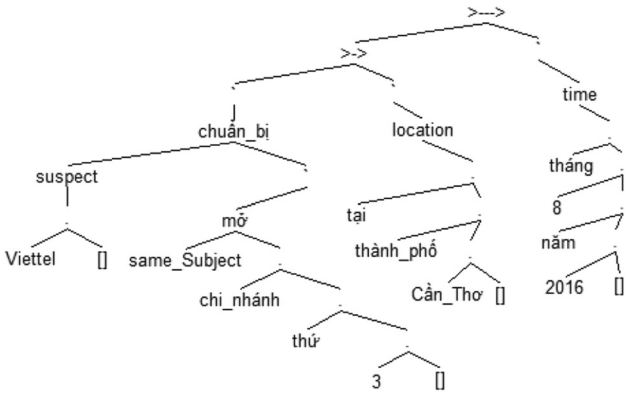

Fig. 1 Semantic tree form of *"Viettel chuẩn bị mở chi nhánh thứ 3 tại thành phố Cần Thơ vào tháng 8 năm 2016"* with the proposed method in [13]

We again used the following sentence sample: *"Viettel chuẩn bị mở chi nhánh thứ 3 tại thành phố Cần Thơ vào tháng 8 năm 2016",* which analyzed the semantics of a sentence with the proposed method in [1,2] and the implementation techniques of this research. The result was two semantic expressions, which were shown in SWI-Prolog [14–16] with the semantic tree form for the two approaches in Figs. 1 and 2.

In Fig. 2, we see that the semantics of the sentence is represented in more detail than the semantic in Fig. 1. In the proposed semantic models, the result properly evaluated that the semantic expression obtained so far is promising, and we need to improve our clause sets of the Definite Clause Grammar further to apply the building processing of the Vietnamese Question-Answering System.

In summary, the semantic expressions of sentences in this research express much more detailed semantics than the semantic expressions of sentences in [13]. The semantic expressions (semantic models) of phrases literally solved synonyms of two phrases. For example, the three noun phrases *"Viettel"*, *"viễn thông Viettel"*, and *"tập đoàn viễn thông Viettel"* have only a semantic expression like the result in No.1-Table 15. Or, with two place phrases

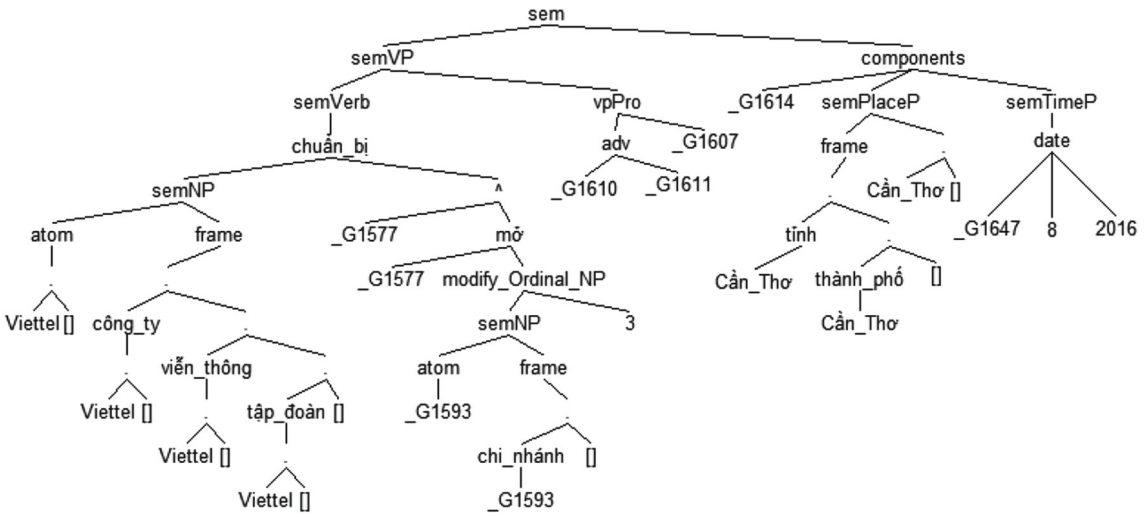

Fig. 2 Semantic tree of *"Viettel chuẩn bị mở chi nhánh thứ 3 tại thành phố Cần Thơ vào tháng 8 năm 2016"* with method in this research

"*tại tỉnh Cần Thơ*" and "*tại thành phố Cần Thơ*" having only a semantic expression express the semantics.

Next, we discuss the correctness of the DCG rule sets in Sect. 2. For the DCG-based approaches to implement grammar structures of linguistics, there were two discussion issues as follows: (1) The accuracy of DCG grammar sets was used to compute and infer the semantics of phrases and sentences; (2) The number of DCG grammar rules was in QA system.

First, we based on the proposed semantic models in [1,2] to build DCG grammars for computing the semantics of a sentence. Thus, we must have, according to the results of the experiment to properly evaluate for implementing the grammar rules of phrases and sentences. The average result of the experiments successfully is around 91.20%, which proved that the correctness of the DCG rule sets is acceptable (results of this research were proved by experiment method/experimental study). With regard to the other words, we proposed and provided most of the DCG grammar samples for other lexicons. The samples were described in the explanation of Tables 1, 3, 5, 7, and 12 (a semantic form of lexicons), which is a general rule to define many similar DCG grammars.

Second, the number of DCG grammar rules depends on the scope of the system and rules of Vietnamese diversified grammar. In addition, construction processes of DCG grammar which were designed reasonably inherited together from the first DCG grammars (the same as Object-Oriented Approach, the principle of inheritance). Thus, we only develop new DCG grammars to analyze complex sentences. In addition, the development processes of DCG grammar are not difficult with the general formalism (in the interpretation of Tables 1, 3, 5, 7, and 12). With the grammatical rules of phrases and sentences, we have knowledgeable about Vietnamese grammar that can define the new DCG grammars of the phrases and sentences.

4 Related works

In this section, we summary most of related works for using Definite Clause Grammar (DCG) to process the semantics and syntactic analysis of phrase, sentence in the interest of Natural Language Understanding (applied in some languages, such as English, Japanese, Romanian, Arabic, and Vietnamese). In addition, we also represent a linguistic approach to Vietnamese grammar in this research for building the DCG grammar reasonably. Whereby, the related works proved the importance of DCG grammars to implement rules of the Vietnamese grammar. First, most of the studies used DCG grammar to implement linguistic rules for understanding natural language in Fig. 1 and Fig. 2.

Collec-Clerc [23] introduced an approach method using NooJ platform[4] into Prolog to analyze and generate many sentences in the Japanese language. In a part of the implementation method, Collec-Clerc used Definite Clause Grammar formalism to compute grammatical rules of the Japanese language. The study resolved issues for generating sentence validly in the given context. In addition, the strength of this study's findings successfully combined between Nooj and Prolog to compute the semantic features for generating sentences.

Patrut [24] developed the DIASEXP system [25]. The system was able to analyze the syntactic of sentences on the Romanian language. The author represented a special

[4] NooJ: A Linguistic Development Environment. http://www.nooj-association.org/. Accessed February 18, 2017.

grammar for processing structures of a complex sentence (a type of sentence expresses events in the current life), in which ideas of research were based characteristics of the Romanian language on building a set of special grammar represented a Context-Free Grammar (CFG) in formal language theory, and, next, using Definite Clause Grammar formalism to implement the CFG grammar.

Ball [26] described the main issue of applying the ACT-R[5] cognitive architecture into Prolog environment for analyzing linguistic modeling, and theory for simulating and understanding human cognition. ACT-R (Adaptive Control of Thought-Rational) which is a cognitive architecture developed by John R. Anderson (1973-present) at Carnegie Mellon University. To resolve a part of the mentioned issue, the author also used the Definite Clause Grammar formalism. In which, DCG formalism supported to execute PCFG (probabilistic context-free grammar) in probabilistic mechanisms of research.

Crabbé et al. [27] introduced XMG 2 (XMG version 2) framework which was developed underlying XMG [27] (eXtensible MetaGrammar) framework. In which, between linguistic grammar and formal grammar relate together. The grammar rules in XMG and XMG 2 were presented by a formal grammar. The formal grammars call a meta-grammars (a logic program) which were defined by DCG grammars.

Blackburn, Bos [6] presented and guided to develop, build DCG grammar from simple/complex CFG for understanding natural language in English language (Pham, Nguyen [13] is too, but Vietnamese). A part of the research, the authors described most of the approach methods of parsing syntactic structures, computing semantics to analyze phrase and sentences. The results of processing were a logical form to express the meaning of a sentence. In addition, the authors pointed out to relate between First-Order Logic and First-Order Inference in the logical form. The strength of this study's findings explored an inference mechanism for supporting to find a new knowledge.

Second, we represent novel and excellent studies of Vietnamese grammar in linguistics such as Hạo [28,29]. In the two works of research, the author built most of the theory of linguistics such as phonetics, syntax, semantics, and brief of functional grammar with Vietnamese grammar. From the linguistic theories of functional grammar, we were able to define linguistic models to represent the semantics of phrases and sentences for the given grammar rules. The study strengths show semantic form models for lexicons and proposed grammar rules for phrases and sentences to represent semantics (a type of semantic form).

In summary, all the related studies, we proved that the Definite Clause Grammar formalism is very important and one of the good computational tools for implementing grammar rules of linguistics.

5 Conclusions

This paper presented a method for building clause sets in Definite Clause Grammars to express the semantics of phrases and sentences in the Vietnamese Question-Answering System by the approach of computational and inferential semantics, namely based on the proposed semantic models [1,2].

We also implemented the novel implementation techniques to represent the semantics for phrases and sentences by the semantic expression, which creates the events of Data &Knowledge in our Vietnamese Question-Answering System.

These novel implementation techniques overcome the traditional building method of question-answering systems that use the database management system to store data for question-answering system. It is a big challenge to design an event management system in the Vietnamese Question-Answering System.

In future work, we will further study the following issues: (1) building clause sets in Definite Clause Grammar to determine the theme [1] of phrases and sentences in the Vietnamese Question-Answering System; (2) building clause sets to assert the semantic expressions of sentences into the event of Data-Knowledge Base by the Definite Clause Grammar.

Acknowledgements This research is funded by Vietnam National University-Ho Chi Minh City (VNU-HCM) under Grant number C2015-26-04.

References

1. Pham, S.T., Nguyen, D.T.: Semantic analysis and theme determination of Vietnamese phrase and sentence based on computational and inferable methods. Int. J. Simul. Syst. Sci. Technol. **17**(32) 27.1–27.10 (2016). https://doi.org/10.5013/IJSSST.a.17.32.27
2. Pham, S.T., Nguyen, D.T.: A computational and inferential method for analyzing the semantics of phrase and sentence in Vietnamese Question Answering System Model (VietQASM). In: Asia Modelling Symposium 2015 (AMS 2015), Ninth Asia International Conference on Mathematical Modelling and Computer Simulation, Kuala Lumpur 2015, p. 107 (2015)
3. Bos, J.: Open-domain semantic parsing with boxer. In: The 20th Nordic Conference of Computational Linguistics (NODALIDA 2015), Vilnius, Lithuania 2015, pp. 301–304
4. Nugues, P.M.: An introduction to language processing with perl & prolog. Springer, Berlin (2006)

[5] ACT-R. http://act-r.psy.cmu.edu/. Accessed February 18, 2017.

5. Pereira, F.C.N., Shieber, S.M.: Prolog and natural-language analysis. Microtome Publishing, Brookline (2005)
6. Blackburn, P., Bos, J.: Representation and iInference for natural language: a first course in computational semantics (studies in computational linguistics). University of Chicago Press, Chicago (2005)
7. Pereira, F.C.N., Warren, D.H.D.: Definite clause grammars for language analysis–a survey of the formalism and a comparison with augmented transition networks. Artif. Intell. **13**(3), 231–278 (1980). https://doi.org/10.1016/0004-3702(80)90003-X
8. Blackburn, P., Bos, J.: Working with discourse representation theory: an advanced course in computational semantics. CSLI press, Stanford, CA (1999). http://www.let.rug.nl/bos/comsem/book2.html
9. Ogborn, A.: Using definite clause grammars in SWI-Prolog. http://www.pathwayslms.com/swipltuts/dcg/#anch6. Accessed 18 Aug 2016
10. Evans, G.: Semantic structure and logical form. Truth and Meaning. Oxford University Press. pp. 49–75 (1976)
11. May, R.: Syntax, semantics, and logical form. The Chomskyan Turn, pp. 334–359. Blackwell, Oxford (1991)
12. Ovchinnikova, E.: Integration of world knowledge for natural language understanding, vol. 3. Springer Science & Business Media, New York (2012)
13. Pham, S.T., Nguyen, D.T.: Implementation techniques for computing the semantics of Vietnamese news titles in Reading Answering System Model (RASM). In: The Third Asian Conference on Information Systems (ACIS 2014), Nha Trang 2014, pp. 209–216 (2014)
14. Wielemaker, J., Schrijvers, T., Triska, M., Lager, T.: SWI-prolog. Int. J. Theory Pract. Logic Program. **12**(1–2), 67–96 (2012)
15. Wielemaker, J.: An overview of the SWI-prolog programming environment. In: The 13th International Workshop on Logic Programming Environments (WLPE-03), Heverlee 2003, pp. 1–16 (2003)
16. Wielemaker, J.: SWI-Prolog version 7 extensions. in: Proceedings of the International Joint Workshop on Implementation of Constraint and Logic Programming Systems and Logic-based Methods in Programming Environments 2014, pp. 109–123 (2014)
17. Lager, T., Wielemaker, J.: Pengines: web logic programming made easy. Theory Pract. Logic Program. **14**(4–5), 539–552 (2014)
18. Wielemaker, J., Huang, Z., Meij, L.V.D.: SWI-Prolog and the web. Theory Pract. Logic Program. **8**(3), 363–392 (2008)
19. Bos, J.: The“ La Sapienza” question answering system at TREC 2006. In: TREC 2006. Citeseer
20. Bates, M., Bobrow, R., Ingria, R., Stallard, D.: The Delphi natural language understanding system. In: Proceedings of the Fourth Conference on Applied Natural Language Processing, pp. 132–137. Association for Computational Linguistics (1994)
21. Li, L., Dahl, D.A., Norton, L.M., Linebarger, M.C., Chen, D.: A test environment for natural language understanding systems. In: Proceedings of the 17th International Conference on Computational Linguistics, Vol. 2, pp. 763–767. Association for Computational Linguistics (1998)
22. Guida, G., Mauri, G.: A formal basis for performance evaluation of natural language understanding systems. Comput. Linguist. **10**(1), 15–30 (1984)
23. Collec-Clerc, V.: Mixed Prolog and NooJ approach in Japanese benefactive. In: Okrut, T., Hetsevich, Y., Silberztein, M., Stanislavenka, H. (eds.) Automatic Processing of Natural-Language Electronic Texts with NooJ: 9th International Conference, NooJ 2015, Minsk, Belarus, June 11–13, 2015, Revised Selected Papers, pp. 218–225. Springer International Publishing, Cham (2016)
24. Patrut, B.: Syntactic analysis based on morphological characteristic features of the Romanian language. CoRR arXiv:1301.1950 [abs] (2013)
25. Patrut, B.: Using prefixes and endings for syntactic analysis of Romanian sentences describing events: the DIASEXP system. In: 2013 International Conference on Control, Decision and Information Technologies (CoDIT), 6–8 May 2013, pp. 804–809
26. Ball, J.T.: Advantages of ACT-R over prolog for natural language analysis. In: Proceedings of the 22nd Annual Conference on Behavior Representation in Modeling and Simulation (2013)
27. Crabbé, B., Duchier, D., Gardent, C., Roux, J., Parmentier, Y.: XMG: eXtensible MetaGrammar. Computat. Linguist. **39**(3), 591–629 (2012). https://doi.org/10.1162/COLI_a_00144
28. Hạo, C.X.: Tiếng Việt: Mấy vấn đề ngữ âm, ngữ pháp, ngữ nghĩa (Vietnamese: Issues in Phonetics, Syntax, and Semantics). Nhà Xuất Bản Giáo Dục (Education Publisher) (1998)
29. Hạo, C.X.: Tiếng Việt: Sơ thảo ngữ pháp chức năng (Vietnamese: Brief of Functional Grammar). Nhà xuất bản Giáo dục (Education Publisher), (2004)