International Journal of Cognitive Informatics and Natural Intelligence
Volume 15 • Issue 2 • April-June 2021

An AI Using Construction Grammar to Understand Text: Parsing Improvements

Denis Kiselev, Hikima.Net, Sapporo, Japan

ABSTRACT

This paper describes an AI that uses construction grammar (CG)—a means of knowledge representation for deep understanding of text. The proposed improvements aim at more versatility of the text form and meaning knowledge structure, as well as for intelligent choosing among possible parses. Along with the improvements, computational CG techniques that form the implementation basis are explained. Evaluation experiments utilize a Winograd schema (WS)—a major test for AI—dataset and compare the implementation with state-of-the-art ones. Results have demonstrated that compared with such techniques as deep learning, the proposed CG approach has a higher potential for the task of anaphora resolution involving deep understanding of the natural language.

KEYWORDS

Anaphora Resolution, Artificial Intelligence, Cognitive Linguistics, Knowledge Representation, Natural Language Understanding (NLU), Winograd Schema

INTRODUCTION

This section first introduces background research in CG. Next, to explain the problem to be solved in the evaluation experiments, the section deals with background research in WS anaphora resolution. Finally, in view of the background research the proposed implementation is briefly discussed and improvements over a former implementation are outlined.

The CG theory is based upon describing "a linguistic sign" (e.g., a word) by pairing its form (or sound structure) with the corresponding meaning, an idea found in early linguistics (De Saussure et al., 2011). Nowadays the label CG is used for various approaches—about seven research directions in total—to describing form-meaning pairings not only for morphemes and words but also for "idioms and abstract phrasal patterns" (Hoffmann and Trousdale, 2013). The CG has also been used to analyze speech discourse by chunks larger than a sentence (Antonopoulou and Nikiforidou, 2011). Moreover, an existing CG approach takes into account phenomena of breaking linguistic conventions to express new meanings i.e. the "fluidity" of the natural language (Steels, 2011a). Another direction in CG research examines how the language is acquired and used form the neurolinguistic perspective and models relations among concepts, i.e. among word meanings (Feldman, 2008). The CG finds its practical applications in robotics from experiments in robot linguistic communication (Steels, 2015) to experiments in entity recognition and acting on natural language commands (Khayrallah et al., 2015) to designing a humanoid robot with language acquisition capability (Hild et al., 2012).

The WS, on which the proposed implementation is evaluated, is "a small reading comprehension test" that can be used to determine if a machine is capable of acting similarly to people's thinking

DOI: 10.4018/IJCINI.20210401.oa4

(Levesque et al., 2011). A separate Schema typically consists of a descriptive part, a question and two possible answers. The descriptive part presents a reference (pronoun anaphora, etc.) resolution challenge such as that in the example: "The city councilmen refused the demonstrators a permit because they [feared/advocated] violence." (Winograd, 1972). Depending on which alternative in the square brackets is used in the sentence, an adult proficient in English would easily understand that "they" refers to "the city councilmen" or "the demonstrators". Understanding this, however, can pose serious problems for a machine as not only word matching and syntactic analysis but also examining "facts about the world" is needed (Winograd, 1980). Moreover, the WS is considered "Google-proof", i.e. if the solution is not yet posted on-line, accessing a large text corpus is unlikely to help much (Levesque et al., 2011; Bailey et al., 2015).

Along the same lines, anaphora resolution research—dealing with what an ambiguous pronoun refers to in a sentence—demonstrates that the pronoun reference phenomenon should be explained from the semantic and pragmatic viewpoints (Bach, 1994; Peterson, 1997). Moreover, corpus-based statistical methods of anaphora resolution are considered shallow in semantics (Mitkov, 2014; Richard-Bollans et al., 2018). Most of the WS anaphora resolution approaches known to the author appear to rely on finding matches in large textual data, and show only modest resolution accuracy. Several such approaches (Kruengkrai et al., 2014; Sharma et al., 2015) generate Google queries for the WS, the solution is found by comparing query results returned by Google and the text that has been used for query generation. Similar works (Budukh, 2013; Peng et al., 2015; Emami et al., 2018) along with results returned by a search engine utilize other data such as the human knowledge base. A different study (Hofford, 2014) deals with parsing entities, such as people, and entity features, such as something people can do, in the WS. Other works (Kiselev, 2017; Raghuram et al., 2017) demonstrate that CG can be more resourceful in terms of the knowledge structure and reasoning, than the mentioned approaches, for the difficult task of the WS anaphora resolution.

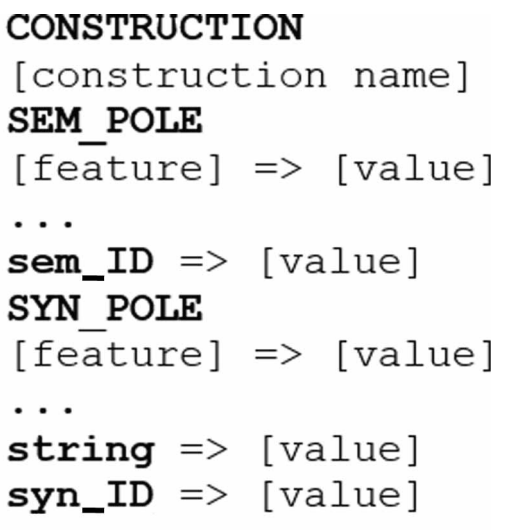
Considering the variety of aspects covered by the CG research described above, the proposed implementation stands against that research background as follows. The implementation is based upon the concept of using various parts of knowledge from as many sources as needed for a practical computational task. To perform the task, relations among the knowledge parts (arranged into constructions) are established and/or analyzed. The knowledge sources are not supposed to be limited to any particular CG field and can be related to, say, neurolinguistics or cognitive psychology. For example, meaning implication information for some adjectives has been incorporated and used for pronoun reference resolution. Moreover, the implemented knowledge structure allows incorporating data from such sources as a human knowledge base. For example, concept relations from ConceptNet (Speer and Havasi, 2013) can potentially be incorporated into the proposed format constructions.

Viewed from the WS anaphora resolution perspective, the proposed implementation has the following peculiarities. The resolution is not based upon statistical processing, e.g. finding and counting matches in big data such as corpora or knowledge bases. The proposed method is based upon automatically combining constructions that store fine-grained ("rich") information on the text form and meaning and, if combinable, automatically modifying constructions as needed for a practical task. At present, new constructions are made up and put into the construction database manually or semi-automatically. In terms of CG language acquisition research (Chang, 2008) creating a database of constructions is compared to teaching a human learner new words and how to use them. To design a construction is sometimes not a trivial task (Van Trijp, 2011) depending on the text it covers. However, a properly designed construction database can be reused for a variety of text that has similar properties.

The proposed implementation, coded in Perl, is publicly available[1] and is an improvement over one described by Kiselev (2017). The enhancements are as follows.

1. **Choosing Among Possible Parses:** Alternative parses are recursively generated (e.g., if text is ambiguous) and knowledge from constructions is applied to choose the appropriate parse.

Figure 1. The word construction template



Recursive parsing also reduces construction proliferation: some constructions can be reused on various text, so new ones do not need to be made.

2. **Use of Regex:** Perl regular expressions[2] (aka regex) can be parts of constructions, which allows more versatility in matching and combining constructions into networks used to "understand" text.
3. **Use of Rules:** Rules are used to resolve anaphora. Rules can also be reusable, e.g. the same rule can be used for multiple WS.

MAIN CONCEPTS

This section gives an overall idea of how the proposed implementation works. The section first explains the construction concept then gives examples of WS sentences processed. Next the input-output flow is described.

Text processing starts by matching the value of string in word constructions (Figure 1) with the input word. If there is a match the word is considered to be "known". Each word construction has form and meaning information about a single word and has the spelling of that word as the value of string. Constructions contain two major parts to which Steels (2011b) refers as "syntactic pole" and "semantic pole". He also says (ibid.) that the "syntactic pole" may describe not only syntax but also morphology and other text properties. Steels (2011b) ideas regarding construction parts have been utilized in developing the proposed implementation CG format. The construction part containing any data concerning the meaning is labeled SEM_POLE and the part containing any data concerning the form is labeled SYN_POLE (Figures 1 and 2). Differently from other research (Feldman et al., 2009), the proposed implementation does not use separate ontology files that represent possible combinations of text segments with certain meanings.

Figure 1 shows the word construction template the proposed implementation utilizes. The template for both phrase and sentence constructions is shown in Figure 2. The following conventions are used for the templates in Figure 1 and Figure 2. (1) Fixed-format (i.e. as a rule unchangeable) markup tags are bold in the figures. (2) The slot [construction name] is filled with a name identifying a construction, word construction names normally end in _w, phrase construction names in _phr or _p and sentence construction ones in _s. Construction names for word constructions normally have the word spelling,

Figure 2. The phrase/sentence construction template

```
CONSTRUCTION
[construction name]
CONSTRUCTION_COMPONENTS
[components]
SEM_POLE
__ [unit number]_[unit name]
[feature] => [value]
...
sem_ID => [value]
...
__ [unit number]_PU_[unit name]
[feature] => [value]
...
sem_ID => [value]
SYN_POLE
__ [unit number]_[unit name]
[feature] => [value]
...
syn_ID => [value]
...
__ [unit number]_PU_[unit name]
[feature] => [value]
...
syn_ID => [value]
```

sentence or phrase construction names have sentence or phrase types. (3) Text form and meaning properties are described by means of [feature] => [value] pairs which are filled with actual features and values, e.g. modality type => impossibility. The ... (ellipsis) indicates that there may be more feature-value pairs or units. (The unit concept is explained below). (4) sem_ID (semantic identifier), syn_ID (syntactic identifier) and string (a word character string, i.e. a word spelling) must be present in constructions. Values for the ID features are usually semantic or syntactic functions: e.g., the pair sem_ID => determiner in the construction for the word "the" represents the semantic function of the definite article. If, say, this function matches the same function in the determiner-noun phrase construction, the word construction for "the" can become a component of that phrase construction (which happens in the case of the determiner-noun phrase "the man"). In this way simple as well as complex networks of constructions are formed, which may be compared to the way humans recall and put together syntactic and semantic knowledge needed to understand text.

The following conventions are used for the template in Figure 2. (1) The slot [components] is filled with names of other constructions that have become components of a phrase or sentence construction. For instance, the word constructions named the_w and man_w (representing the known words "the" and "man") can be components of determiner_noun_phr, which is a name for a phrasal construction that can include a determiner (such as "the") and a noun (such as "man"). (2) [unit number]_[unit name] are filled with the number and name for each child unit representing a component of a phrase or sentence construction. For example, the construction for the phrase "the man" includes two child units: one for "the" and one for "man". In the SYN_POLE (syntactic pole)

the first child unit (describing the determiner "the" in the phrasal construction for "the man") can be marked 1_determiner. (The underscore is used for output readability.) (3) [unit number]_PU_[unit name], PU meaning "parent unit", are filled with the number and name for the parent unit of a phrase or sentence construction. The parent unit is used to describe form or meaning properties for the phrase or sentence as a whole and is matched with feature-value pairs of larger constructions. If they match, the smaller construction becomes a component of a larger one, i.e. a network used for text understanding is formed. In the SYN_POLE of the same "the man" example the parent unit can be marked 3_PU_np, np stands for "noun phrase".

An actual word construction for a known word (that matched the construction string value) is shown in Figure 3. An actual phrase construction in which ID and other values have matched those of two word constructions is shown in Figure 4. Because they matched the phrase network has been formed, and the PU in that phrase construction has been populated with a meaning property from the word construction heavy_w, i.e. the pair implies => HINDRANCE has been automatically copied because it met conditions explained in PROCESSING WORD AND PHRASE CONSTRUCTIONS. In human terms, the judgment behind this procedure is as follows. As the word "heavy" implies hindrance and "heavy" is a component of the phrase "so heavy", this phrase also implies hindrance.

In the rest of the paper a WS descriptive part "The man couldn't lift his son because he was so [weak/heavy]." will be used to illustrate text processing. The templates described above have been utilized by the author to make up constructions for the following two sentences that in the rest of the paper are referred to as Example 1 and Example 2.

Example 1

The man couldn't lift his son because he was so weak.

Example 2

The man couldn't lift his son because he was so heavy.

Knowledge needed to process Example 1 and Example 2 was manually and/or semi-automatically entered as feature-value pairs into word, phrase and sentence constructions. "Semi-automatically" refers to using a program that changes human input into the proposed construction format. The constructions were entered into the database for storing knowledge. Two key concepts, exemplified above, implemented for processing the construction network are as follows. (1) Feature-value pairs (ID ones and others) of smaller constructions are matched with those of larger constructions, if there are matches smaller constructions become components larger constructions. (2) When smaller constructions become components of larger constructions some knowledge contained (as feature-value pairs) in the smaller constructions can be passed over to be stored in the larger ones and vice versa: from larger to smaller ones. In previous CG research (Steels, 2011a; Steels, 2017) similar concepts of construction processing are referred to as "matching" and "merging".

The proposed implementation input-output flow is based upon the two concepts explained above and includes the following major stages.

1. **Processing Word Input and Word Constructions:** The input string (e.g., all words for Example 1 or Example 2) is matched with string values in word constructions. See Figure 1 for the word construction template. If complete matches are found in word constructions, those word constructions will be matched with phrase ones at stage 2 explained next. Perl regex that can be used as values of string add more word matching versatility. Alternative word parses for the same spelling that can belong to different parts of speech are recursively generated.
2. **Processing Word and Phrase Constructions:** If ID and other feature-value pairs in word and phrase constructions match, word constructions become components of phrase ones. Out of the alternative word parses resulting from stage 1 the best one is chosen by examining the number of matching phrase constructions. At this and the next stage some meaning and/or form properties,

Figure 3. A word construction example

```
CONSTRUCTION
heavy_w

SEM_POLE
implies => HINDRANCE
match_by_this_and_ID => yes
sem_ID => selector
meaning => heavy

SYN_POLE
match_by_this_and_ID => yes
syn_ID => adjective
string => heavy
```

i.e. feature-value pairs such as ones important for anaphora resolution, propagate within the network (are automatically copied from/to constructions). Phrase constructions are recursively matched with other phrase constructions to find phrases that are parts of other phrases. Regex can be also used to match phrase and sentence constructions to allow or disallow certain values or their parts to match.

3. **Processing Sentence and Sentence Component Constructions:** The phrase network from stage 2 is recursively matched with sentence constructions to determine what words, phrases and smaller sentences fit as components into sentences according to the rules of the natural language (English in the proposed implementation case). The language rules are reflected in construction feature-value pairs that can match or mismatch thus allowing or disallowing constructions to fit within others. In other words, those pairs control the machine "understanding": determine what text pieces can be components of the grammatically correct sentence. Finally, rules dealing with text logical structure are used for anaphora resolution and the feature-value pairs, showing what e.g. an ambiguous pronoun refers to, are inserted into appropriate constructions.

From the neurolinguistic perspective the above stages can be viewed to result in neural patterns of activity within a network (Dominey et al., 2006; Wellens, 2011) or established associations among constructions in the network for the respective input. After those associations are established the proposed implementation "understands" semantic and syntactic relations: what text chunks are grammatically correct components of others, how their meanings are related, etc. The available anaphora resolution data are retrievable from the sentence and/or sentence component constructions.

ROCESSING WORD INPUT AND WORD CONSTRUCTIONS

The purpose of this processing stage is to find familiar words in the input. Figure 3 shows a word construction for the adjective "heavy" in Example 2.

As mentioned above, values of string, such as "heavy" in Figure 3, in all word constructions stored in the construction database are matched with all input words. If complete matches (for instance, the input word "heavy" matching the spelling "heavy" in string => heavy) are found, the proposed

implementation has found familiar words, so it "knows" that these words possess form and meaning properties recorded in word constructions (as features and values separated by =>).

Regex that can be used as values of string allow matching word forms the designer believes acceptable. Say, the Perl regex met(er|re) or even met[er]{2} can match variant words spellings: "meter" and "metre".

Alternative word parses are recursively generated for the same character string that can represent different parts of speech. For instance, the apostrophe s ('s) ending can match two constructions: one for the possessive ending, the other for the verb "be". Both constructions will be passed to the next processing stage as representatives of the apostrophe s string.

Two feature-value pairs important for resolving the "he" pronoun anaphora in Example 1 and Example 2 are implies => ABILITY_LACK and implies => HINDRANCE respectively. The latter feature-value pair can be found in Figure 3, the former is present in a similar construction for the word "weak". At the subsequent processing stages the above values will be copied from word to phrase and sentence constructions and used to find out what "he" refers to in these two examples.

PROCESSING WORD AND PHRASE CONSTRUCTIONS

The purpose of this stage is to form phrasal networks i.e. to "understand" what parts of the text are grammatically acceptable phrases. The networks are formed if the ID and other feature-value pairs in the word constructions resulting from the previous stage (in which those constructions matched input words) match respective pairs of phrase constructions from the database. Some values are also copied. Recall the MAIN CONCEPTS section, i.e. the two network processing principles and the example for copying implies => HINDRANCE.

Figure 4 demonstrates the pron_adj_phr (pronoun and adjective phrase) construction after the phrase "so heavy" (Example 2) was "understood", in other words, after this constructions matched the ID and other feature-value pairs in constructions for the pronoun "so" and adjective "heavy", and after the HINDRANCE implication was copied into 3_PU_select_expr (Figure 4). The construction pron_adj_phr is versatile as it can also be applied to such phrases as "so weak" in Example 1. It can be seen from Figure 4 that the construction has units for the words "so" and "heavy" as well as a parent unit in each pole. In the semantic pole the word units are labeled 1_intensifier and 2_selector. In the syntactic pole they are labeled 1_pronoun and 2_adjective. The labeling reflects the words semantic and syntactic functions as parts of the phrase.

It has been said above that networks are formed if the ID and other feature-value pairs match, but speaking strictly the proposed implementation checks if the two conditions listed below are met. As an illustration, in Figure 4 see units 1 and 2 representing "so" and "heavy" in each pole, and see sem_ID and syn_ID pairs in each of these units. Constructions form networks i.e. word constructions become components of phrase ones if:

1. **The values of sem_ID and syn_ID in the word construction match the values of sem_ID and syn_ID in the unit representing that word construction in each pole of the phrase construction.** For instance, sem_ID => selector in the SEM_POLE of the construction heavy_w (Figure 3) matches the same pair in 2_selector unit of the SEM_POLE in pron_adj_phr (Figure 4) and so on. The construction for the word "so" matches similarly. Perl regex can also be used as values, e.g. to allow alternative constructions to match.
2. **In the units stipulated by the above condition at least one feature-value pair besides the sem_ID or syn_ID pair matches.** For example, in the case of the word "heavy" such pair is match_by_this_and_ID => yes (Figures 3 and 4). The pair is a "dummy" used by the author to satisfy this condition when it practically suffices to match only the ID pairs.

Figure 4. A phrase construction example

```
CONSTRUCTION
pron_adj_phr

CONSTRUCTION_COMPONENTS
so_w heavy_w

SEM_POLE
___1_intensifier
match_by_this_and_ID => yes
sem_ID => intensifier
___2_selector
match_by_this_and_ID => yes
sem_ID => selector
___3_PU_select_expr
implies => HINDRANCE
sem_ID => select_expr
match_by_this_and_ID => yes

SYN_POLE
___1_pronoun
match_by_this_and_ID => yes
syn_ID => pronoun
___2_adjective
match_by_this_and_ID => yes
syn_ID => adjective
___3_PU_adjp
syn_ID => adjp
match_by_this_and_ID => yes
```

As mentioned above, values like HINDRANCE are copied within a formed network. The idea behind this is to propagate a semantic (or some other) property form one construction to another. E.g., if "weak" (Example 1) implies "ability lack" then "so weak" implies the same: the implication property propagates from a smaller to a larger construction. Values are copied from a construction to another within a phrasal network if:

1. **One of the two constructions is a component of the other (and satisfies the construction matching conditions above).**
2. **The two constructions must have feature-value pairs in which the features are the same but the values are not and only one of those values (or value lists) must be empty or marked "undef".**

In the case of the phrase "so heavy" (Example 2) the value HINDRANCE (Figure 3) important for the anaphora resolution has been copied to the phrase construction (Figure 4). Before the value was copied, the construction pron_adj_phr shown in Figure 4 had implies => undef in the 3_PU_select_expr unit. (Recall that Figure 4 demonstrates the construction after it was used to process the phrase.) The same two conditions were met before the ABILITY_LACK implied by "weak" in Example 1 was copied to (another instance of) the pron_adj_phr construction. The values ABILITY_LACK and HINDRANCE are used for anaphora resolution at the next processing stage.

As for the alternative word parses representing the apostrophe and s combination mentioned in PROCESSING WORD INPUT AND WORD CONSTRUCTIONS, phrase constructions were

recursively applied to each of the parses. At this time a simplistic algorithm choosing the parse that matches more phrase constructions is implemented for this and similar cases. Constructions in the database are designed for matching grammatically acceptable text features so more constructions match grammatically acceptable parses. A grammatically acceptable phrase of, say, "because it's too large" interpreting 's as "be" would match more constructions than one interpreting it as the possessive ending. The same approach is used for choosing phrase constructions for components of other phrase constructions: the parse matching more constructions is passed to the next processing stage, or several parses with the same number of matching constructions are passed.

PROCESSING SENTENCE AND SENTENCE COMPONENT CONSTRUCTIONS

The purpose of this stage is to form sentence networks by choosing text parts (represented by constructions passed from the previous stage) to be sentence components. Grammatically acceptable components are chosen if they meet two conditions similar to those explained earlier:

1. **The values of sem_ID and syn_ID in the word construction, or those of sem_ID and syn_ID in the parent unit (PU) of a phrase or sentence construction match the values of sem_ID and syn_ID in the unit representing that construction in each pole of the sentence construction.** For instance, the ID values in the parent units of the sentence constructions for the main and subordinate clauses (Example 1, 2) have matched the ID values in the complex sentence construction units representing those clauses. Regex used as the values add more versatility in matching variant clauses for similar sentences.
2. **In the units stipulated by the above condition at least one feature-value pair besides the sem_ID or syn_ID pair matches.** As forming both sentence and phrasal networks is based on the same main concept, this condition is the same as for phrasal networks.

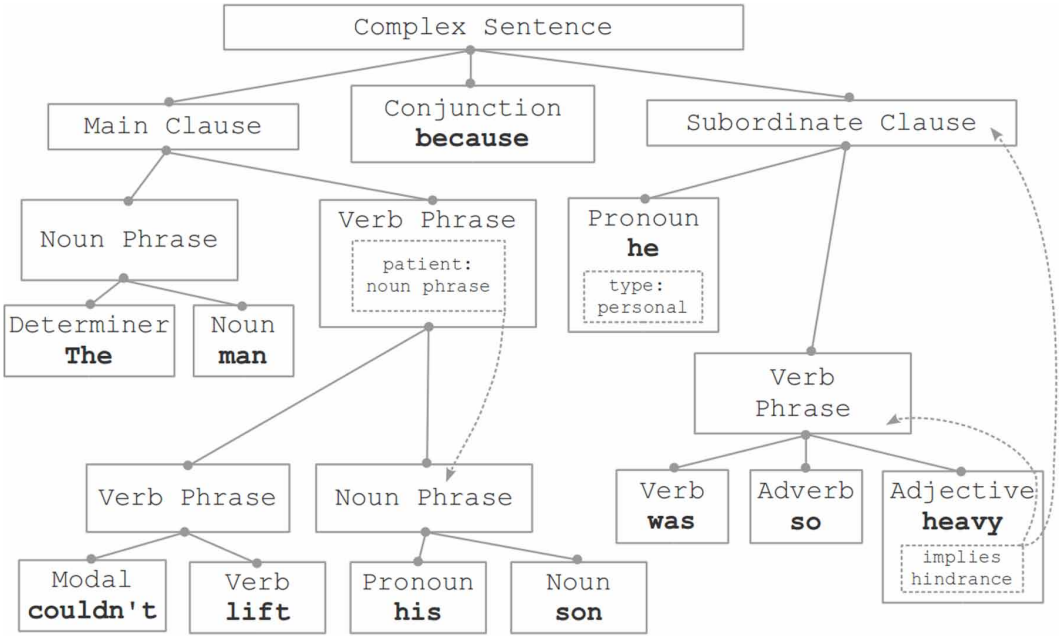
The same propagation (i.e. copying) conditions as explained in PROCESSING WORD AND PHRASE CONSTRUCTIONS are used for the sentence network. That section also explains the algorithm that is also used to choose the sentence parse in which more constructions match.

Figure 5 shows a formed network of constructions for a complex sentence (Example 2). Each solid line box represents a construction, lines with circles on ends connect constructions with their components, bold type words have been matched in the text by string feature-value pairs (Figures 1 and 3) in word constructions.

Rules manually designed by the author and stored alongside constructions in the database were applied to this network in order to determine what "he" refers to in the sentence. Information examined to perform this anaphora resolution task is inside the dashed line boxes. As explained earlier, this information is stored in constructions as feature-value pairs. Dashed arrows show how anaphora resolution information propagated. For instance, the implication information (actually stored as the pair implies => hindrance) was copied from the word construction for "heavy" to the subordinate clause sentence construction. It was copied because the "heavy" construction was designed to have implies => hindrance, but the clause construction to have implies => undef (recall the copying concept explained earlier). The receiver of the action (i.e. patient) information was copied from the "couldn't lift his son" verb phrase (VP) construction to the "his son" noun phrase (NP) one similarly: the unit representing "his son" in the VP construction had used_as_patient => yes, but the "his son" NP construction had used_as_patient => undef. (The author uses this simplistic feature-value format for clarity; a construction designer is free to use a different one.)

The anaphora resolution rule design[3] includes the condition and action part: an action is taken if a condition is met. A condition can be double, triple and so on. For an illustration of the rule explained next, see the information in the dashed boxes and how it propagated to other constructions, which is shown by the dashed arrows (Figure 5). The rule used for the Example 2 sentence network can

Figure 5. A scheme for a sentence network of constructions



be glossed using the human communication language as follows. In a complex sentence in which a subordinate clause implies hindrance (has a part like "heavy"), into a part that is a personal pronoun (like "he"), inset refers_to => pointing to a part that denoted the patient (like "his son"). This is one possible solution of the anaphora problem; one is free to design other constructions and rules.

EVALUATION EXPERIMENTS

This section first describes the experiment setting and systems used. Next, the output is analyzed. For the experiments a dataset of ten WS statements was used. The dataset, given below, was obtained from a WS collection publicly available on-line[4], no original WS wording has been altered.

1. The man couldn't lift his son because he was so weak.
2. The man couldn't lift his son because he was so heavy.
3. Jim yelled at Kevin because he was so upset.
4. Jim comforted Kevin because he was so upset.
5. John couldn't see the stage with Billy in front of him because he is so short.
6. John couldn't see the stage with Billy in front of him because he is so tall.
7. The city councilmen refused the demonstrators a permit because they feared violence.
8. The city councilmen refused the demonstrators a permit because they advocated violence.
9. The trophy doesn't fit into the brown suitcase because it's too small.
10. The trophy doesn't fit into the brown suitcase because it's too large.

Two systems (Table 1) were used in the experiments along with the proposed implementation. One of them is Berkeley Entity Resolution System (Durrett and Klein, 2013). This system detects mentions (e.g., words) with certain "shallow features" in certain positions in the text. Occurrence counts of those mentions are used in calculating the probability of the anaphoric relation. Such

Table 1. Anaphora resolution results

#	Pronoun	Berkeley Entity Resolution System	Stanford CoreNLP	Proposed
1	he	the man	~~his son~~	the man
2	he	~~the man~~	~~the man~~	his son
3	he	Jim	Jim	Jim
4	he	~~Jim comforted~~	~~Jim~~	Kevin
5	he	John	John	John
6	he	~~John~~	~~John~~	Billy
7	they	~~the demonstrators~~	The city councilmen	The city councilmen
8	they	the demonstrators	~~The city councilmen~~	the demonstrators
9	it	~~The trophy~~	~~The trophy~~	the brown suitcase
10	it	The trophy	The trophy	The trophy

shallow features as nominal or pronominal ones and such positions as precedence or antecedence are taken into consideration. The number of words between two anaphoric relation candidates is also considered. Position variations are modeled as templates which are applied to the text.

The other system is Stanford CoreNLP with the neural coreference annotator (Clark and Manning, 2016a; Clark and Manning, 2016b), which can be shortly described as follows. The annotator features a deep neural network with three hidden layers "learned" from CoNLL 2012 data. The layers contain such information as distance between word mentions, document genre and string-matching features. Clusters containing this information are generated for separate words in the input and the layers and are used for computing the score that determines the likelihood of words coreference. The algorithm used for computing attempts to maximize the score if in the processed source candidates for the anaphoric relation are found. The algorithm accuracy can be adjusted by using numeric parameters.

Table 1 shows referents output for the ambiguous anaphoric pronouns. The ambiguous pronouns in question are listed in the column "Pronoun" for each dataset sentence. The row numbers correspond to the dataset sentence numbers. The output that is entirely incorrect from the human perspective is crossed out. To sum up the data in Table 1, the total numbers of sentences in which the ambiguous anaphoric pronoun was correctly disambiguated are as follows: Proposed: 10, Berkeley Entity Resolution System: 5, Stanford CoreNLP: 4. The results demonstrate that, unlike the other two systems, the proposed implementation correctly disambiguated all the pronouns in question on the above dataset.

DISCUSSION

The Stanford and Berkeley systems based upon the statistical Natural Language processing (NLP) were chosen for the experiments because, as the author has observed (recall INTRODUCTION), contemporary research tends to use this NLP approach for difficult anaphora resolution rather than the deep NLU one. It should be also mentioned that a prior implementation, which the proposed one is aimed at improving, was compared (Kiselev, 2017) with Stanford CoreNLP that used different statistical-NLP-based annotators. The results for the Stanford system were very similar to those shown in Table 1.

It is not true that every phrase and sentence construction in the proposed implementation database was designed for some particular sentence only: the constructions were designed for form and meaning structures commonly found in the English text.

The purpose of this paper is not to show how many WS constructions form an entire WS collection the proposed implementation can successfully process. The primary purpose is to show the CG knowledge structure potential to store, propagate and reason on rich data in order to solve the difficult anaphora problem.

As for the scalability issue (for instance, how many certain problems a particular program or knowledge database is good at solving in an unsupervised way), work on it is in progress. The author intends to continue research into WS anaphora resolution that utilizes automatically learned constructions. Some modest scalability has, however, been observed in the present implementation: for instance, the same rule can be used on items 1 and 5 of the dataset (EVALUATION EXPERIMENTS); also, most complex sentence constructions are reused on multiple items of the dataset.

CONCLUSION

The paper has described an implementation that utilizes the CG potential for storing and manipulating syntactic, semantic and pragmatic knowledge to resolve the WS pronoun anaphora, a task that involves deep NLU and can be looked upon as especially difficult for systems based on statistical NLP.

Evaluation experiments have demonstrated that the proposed implementation potential for deep understanding of the natural language can be considered high compared with the mentioned systems. The improvements over a prior implementation can be seen to result in more processing versatility.

The lack of scalability of the proposed implementation can possibly be improved by automatic construction acquisition that is left for future work.

REFERENCES

Antonopoulou, E., & Nikiforidou, K. (2011). Construction Grammar and Conventional Discourse: A Construction-based Approach to Discoursal Incongruity. *Journal of Pragmatics*, *43*(10), 2594–2609.

Bach, K. (1994). Anaphoric Reference: Grammatical or Pragmatic? In *Thought and Reference*. Clarendon Press.

Bailey, D., Harrison, A., Lierler, Y., Lifschitz, V., & Michael, J. (2015). The Winograd Schema Challenge and Reasoning about Correlation. *Working Notes of the Symposium on Logical Formalizations of Commonsense Reasoning*.

Budukh, T. (2013). *An Intelligent Co-reference Resolver for Winograd Schema Sentences Containing Resolved Semantic Entities* (Doctoral Dissertation). Arizona State University.

Chang, N. (2008). *Constructing Grammar: A Computational Model of the Emergence of Early Constructions*. ProQuest.

Clark, K., & Manning, C. D. (2016a). Deep Reinforcement Learning for Mention-ranking Coreference Models. Empirical Methods on Natural Language Processing. doi:10.18653/v1/D16-1245

Clark, K., & Manning, C. D. (2016b). *Improving Coreference Resolution by Learning Entity-level Distributed Representations.* arXiv preprint arXiv:1606.01323

De Saussure, F. (2011). *Course in General Linguistics [1916]*. Columbia University Press.

Dominey, P., Hoen, M., & Inui, T. (2006). A Neurolinguistic Model of Grammatical Construction Processing. *Journal of Cognitive Neuroscience*, *18*(12), 2088–2107. PMID:17129193

Durrett, G., & Klein, D. (2013, October). Easy Victories and Uphill Battles in Coreference Resolution. In EMNLP (pp. 1971-1982). Academic Press.

Emami, A., Trischler, A., Suleman, K., & Cheung, J. C. K. (2018). A Generalized Knowledge Hunting Framework for the Winograd Schema Challenge. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop* (pp. 25-31). Academic Press.

Feldman, J. (2008). *From Molecule to Metaphor: A Neural Theory of Language*. MIT Press.

Feldman, J., Dodge, E., & Bryant, J. (2009). Neural Theory of Language and Embodied Construction Grammar. In B. Heine & H. Narrog (Eds.), *The Oxford Handbook of Linguistic Analysis* (pp. 111–138). Oxford University Press.

Hild, M., Siedel, T., Benckendorff, C., Thiele, C., & Spranger, M. (2012). Myon, a New Humanoid. In Language Grounding in Robots (pp. 25-44). Springer US. doi:10.1007/978-1-4614-3064-3_2

Hoffmann, T., & Trousdale, G. (2013). Construction Grammar: Introduction. In *The Oxford Handbook of Construction Grammar*. Oxford University Press.

Hofford, G. (2014). *Resolution of Difficult Pronouns Using the ROSS Method.* arXiv preprint arXiv:1411.4109

Khayrallah, H., Trott, S., & Feldman, J. (2015). Natural Language For Human Robot Interaction. *Proceedings of the Workshop on Human-Robot Teaming at the 10th ACM/IEEE International Conference on Human-Robot Interaction*.

Kiselev, D. (2017). A Construction-Grammar Approach to Computationally Solving the Winograd Schema: Implementation and Evaluation. In *Proceedings of the AAAI 2017 Spring Symposium on Computational Construction Grammar and Natural Language Understanding.* Stanford University.

Kruengkrai, C., Inoue, N., Sugiura, J., & Inui, K. (2014). An Example-Based Approach to Difficult Pronoun Resolution. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing* (pp. 358-367). Academic Press.

Levesque, H. J., Davis, E., & Morgenstern, L. (2011, March). The Winograd Schema Challenge. *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Mitkov, R. (2014). *Anaphora Resolution*. Routledge.

Peng, H., Khashabi, D., & Roth, D. (2015). Solving Hard Coreference Problems. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL* (pp. 809–819). doi:10.3115/v1/N15-1082

Peterson, P. L. (1997). Anaphoric Reference to Facts, Propositions, and Events. In *Fact Proposition Event*. Springer Netherlands.

Raghuram, V., Trott, S., Shen, K., Goldberg, E., & Oderberg, S. (2017). Semantically-Driven Coreference Resolution with Embodied Construction Grammar. In *Proceedings of the AAAI 2017 Spring Symposium on Computational Construction Grammar and Natural Language Understanding.* Stanford University.

Richard-Bollans, A. L., Gomez Alvarez, L., & Cohn, A. G. (2018, January). The Role of Pragmatics in Solving the Winograd Schema Challenge. *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning (Commonsense 2017). CEUR Workshop Proceedings.*

Sharma, A., Vo, N. H., Gaur, S., & Baral, C. (2015, March). An Approach to Solve Winograd Schema Challenge Using Automatically Extracted Commonsense Knowledge. *2015 AAAI Spring Symposium Series*.

Speer, R., & Havasi, C. (2013). ConceptNet 5: A Large Semantic Network for Relational Knowledge. In The People's Web Meets NLP (pp. 161-176). Springer Berlin Heidelberg.

Steels, L. (2011a). A First Encounter with Fluid Construction Grammar. In *Design Patterns in Fluid Construction Grammar* (pp. 31–68). John Benjamins.

Steels, L. (2011b). Introducing Fluid Construction Grammar. In *Design Patterns in Fluid Construction Grammar* (pp. 3–30). John Benjamins.

Steels, L. (2015). *The Talking Heads Experiment: Origins Of Words And Meanings (Computational Models Of Language Evolution 1)*. Language Science Press.

Steels, L. (2017). Basics of Fluid Construction Grammar. *Constructions and Frames*, *9*(2), 178–225.

Van Trijp, R. (2011). A Design Pattern for Argument Structure Constructions. Design Patterns in Fluid Construction Grammar, 11, 115. doi:10.1075/cal.11.07tri

Wellens, P. (2011). Organizing Constructions in Networks. In *Design Patterns in Fluid Construction Grammar* (pp. 181–201). John Benjamins.

Winograd, T. (1972). *Understanding Natural Language*. Academic Press.

Winograd, T. (1980). What Does It Mean to Understand Language? *Cognitive Science*, *4*(3), 209–241.

ENDNOTES

1 https://github.com/a-c-g-v1-and-on/code.git retrieved on February 11, 2019.

2 Parts of the programming language that are used for matching and manipulating character strings such as text.

3 Rule mechanics are explained in detail in the subsection "Adding rules" of the publicly available usage instructions for the proposed implementation, see Endnote 1 for the access URL.

4 https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WSCollection.html retrieved on February 11, 2019.

Denis Kiselev was born in Russia in 1973. Received a diploma in Translation, Interpretation and Teaching from Nizhny Novgorod State Linguistic University, Russia in 1997. Received an M.A. in Computational Linguistics from Hokkaido University, Japan in 2012 and a Ph.D. in Media Systems and Networks from the same university in 2015. Interested in Construction Grammar and more generally in any Artificial Intelligence meant to resemble human thinking and applied to Natural Language Processing.