Improving Text Retrieval Efficiency with Pattern Structures on Parse Thickets

Boris A. Galitsky[1,2], Dmitry Ilvovsky[2], Fedor Strok[2] and Sergei O. Kuznetsov[2]

[1] eBay Inc San Jose CA USA
[2] Higher School of Economics, Moscow Russia
{bgalitsky@hotmail.com; dilv_ru@yahoo.com; fdr.strok@gmail.com; skuznetsov@hse.ru}

Abstract. We develop a graph representation and learning technique for parse structures for paragraphs of text. We introduce Parse Thicket (PT) as a sum of syntactic parse trees augmented by a number of arcs for inter-sentence word-word relations such as co-reference and taxonomic relations. These arcs are also derived from other sources, including Speech Act and Rhetoric Structure theories. The operation of generalizing logical formulas is extended towards parse trees and then towards parse thickets to compute similarity between texts. We provide a detailed illustration of how PTs are built from parse trees, and generalized. The proposed approach is subject to preliminary evaluation in the product search domain of eBay.com, where user queries include product names, features and expressions for user needs, and query keywords occur in different sentences of an answer. We demonstrate that search relevance is improved by PT generalization.

Keywords: graph representation of text, learning syntactic parse tree, syntactic generalization, search relevance

1 Introduction

Parse trees have become a standard form of representing the linguistic structures of sentences. In this study we will attempt to represent a linguistic structure of a *paragraph of text* based on parse trees for each sentence of this paragraph. We will refer to the sum of parse trees plus a number of arcs for inter-sentence relations between nodes for words as Parse Thicket (PT). A PT is a graph which includes parse trees for each sentence, as well as additional arcs for inter-sentence relationship between parse tree nodes for words.

In this paper we will define the operation of *generalization of text paragraphs* to assess similarity between portions of text. Use of generalization for similarity assessment is inspired by structured approaches to machine learning versus unstructured, statistical where similarity is measured by a distance in feature space. Our intention is to extend the operation of least general generalization (unification of logic formula) towards structural representations of paragraph of texts. Hence we will define the operation of generalization on Parse Thickets and outline an algorithm for it.

This generalization operation is a base for number of text analysis application such as search, classification, categorization, and content generation [3]. *Generalization of text paragraphs* is based on the operation of generalization of two sentences, explored in our earlier studies [6,7,8]. In addition to learning generalizations of individual sentences, in this study we explore how the links between words in sentences other than syntactic ones can be used to compute similarity between texts. We will investigate how to formalize the theories of textual discourse such as Rhetoric Structure Theory [12] to improve the efficiency of text retrieval.

General pattern structures consist of objects with descriptions (called patterns) that allow a semilattice operation on them [9]. In our case, for paragraphs of text to serve such objects, they need to be represented by structures like parse thickets, which capture both syntactic level and discourse-level information about texts. Pattern structures arise naturally from ordered data, e.g., from labeled graphs ordered by graph morphisms. In our case labeled graphs are parse thickets, and morphisms are the mappings between their maximal common sub-graphs.

One of the first systems for the generation of conceptual graph representation of text is described in [18]. It uses a lexicon of canonical graphs that represent valid (possible) relations between concepts. These canonical graphs are then combined to build a conceptual graph representation of a sentence. Since then syntactic processing has dramatically improved, delivering reliable and efficient results.

[11] describes a system for constructing conceptual graph representation of text by using a combination of existing linguistic resources (VerbNet and WordNet). However, for practical applications these resources are rather limited, whereas syntactic level information such as syntactic parse trees is readily available. Moreover, building conceptual structure from individual sentences is not as reliable as building these structures from generalizations of two and more sentences.

In this study we attempt to approach conceptual graph level [15, 17] using pure syntactic information such as syntactic parse trees and applying learning to it to increase reliability and consistency of resultant semantic representation. The purpose of such automated procedure is to tackle information extraction and knowledge integration problems usually requiring deep natural language understanding [2] and cannot be solved at syntactic level.

Whereas machine learning of syntactic parse trees for individual sentences is an established area of research, the contribution of this paper is a structural approach to learning of syntactic information at the level of paragraphs. A number of studies applied machine learning to syntactic parse trees [1], convolution kernels being the most popular approach [10].

To represent the structure of a paragraph of text, given parse trees of its sentences, we introduce the notion of Parse Thicket (PT) as a union of parse trees. The union $G = G_1 \bigcup G_2$ of trees G_1 and G_2 with disjoint node sets V_1 and V_2 and edge sets X_1 and X_2 is the graph with $V = V_1 \bigcup V_2$ and $X = X_1 \bigcup X_2$.

2 Finding similarity between two paragraphs of text

We will compare the following approaches to assessing the similarity of text paragraphs:

- Baseline: bag-of-words approach, which computes the set of common keywords/n-grams and their frequencies.
- Pair-wise matching: we will apply syntactic generalization to each pair of sentences, and sum up the resultant commonalities. This technique has been developed in our previous work [3].
- Paragraph-paragraph match.

The first approach is most typical for industrial NLP applications today, and the second is the one of our previous studies. Kernel-based approach to parse tree similarities [20], as well as tree sequence kernel [19], being tuned to parse trees of individual sentences, also belongs to the second approach.

We intend to demonstrate the richness of the approach being proposed, and in the consecutive sections we will provide a step-by-step explanation. We will introduce a pair of short texts (articles) and compare the above three approaches. This example will go through the whole paper.

"Iran refuses to accept the UN proposal to end the dispute over work on nuclear weapons",
"UN nuclear watchdog passes a resolution condemning Iran for developing a second uranium enrichment site in secret",
"A recent IAEA report presented diagrams that suggested Iran was secretly working on nuclear weapons",
"Iran envoy says its nuclear development is for peaceful purpose, and the material evidence against it has been fabricated by the US",

∧

"UN passes a resolution condemning the work of Iran on nuclear weapons, in spite of Iran claims that its nuclear research is for peaceful purpose",
"Envoy of Iran to IAEA proceeds with the dispute over its nuclear program and develops an enrichment site in secret",
"Iran confirms that the evidence of its nuclear weapons program is fabricated by the US and proceeds with the second uranium enrichment site"

The list of common keywords gives a hint that both documents are on nuclear program of Iran, however it is hard to get more specific details

Iran, UN, proposal, dispute, nuclear, weapons, passes, resolution, developing, enrichment, site, secret, condemning, second, uranium

Pair-wise generalization gives a more accurate account on what is common between these texts: -+

[NN-work IN-* IN-on JJ-nuclear NNS-weapons], [DT-the NN-dispute IN-over JJ-nuclear NNS-*], [VBZ-passes DT-a NN-resolution],
[VBG-condemning NNP-iran IN-*],
[VBG-developing DT-* NN-enrichment NN-site IN-in NN-secret]],
[DT-* JJ-second NN-uranium NN-enrichment NN-site]],
[VBZ-is IN-for JJ-peaceful NN-purpose],

[DT-the NN-evidence IN-* PRP-it], [VBN-* VBN-fabricated IN-by DT-the NNP-us]

Parse Thicket generalization gives the detailed similarity picture which looks more complete than the pair-wise sentence generalization result above:

[NN-Iran VBG-developing DT-* NN-enrichment NN-site IN-in NN-secret]
[NN-generalization-<UN/nuclear watchdog> * VB-pass NN-resolution VBG condemning NN- Iran]
[NN-generalization-<Iran/envoy of Iran> Communicative_action DT-the NN-dispute IN-over JJ-nuclear NNS-*
[Communicative_action - NN-work IN-of NN-Iran IN-on JJ-nuclear NNS-weapons]
[NN-generalization <Iran/envoy to UN> Communicative_action NN-Iran NN-nuclear NN-* VBZ-is IN-for JJ-peaceful NN-purpose],
Communicative_action - NN-generalize <work/develop> IN-of NN-Iran IN-on JJ-nuclear NNS-weapons]*
[NN-generalization <Iran/envoy to UN> Communicative_action NN-evidence IN-against NN Iran NN-nuclear VBN-fabricated IN-by DT-the NNP-us]
condemn^proceed [*enrichment site*] <leads to> *suggest^condemn* [*work Iran nuclear weapon*]

One can feel that PT-based generalization closely approaches human performance in terms of finding similarities between texts. To obtain these results, we need to be capable of maintaining coreferences, apply the relationships between entities to our analysis *(subject vs relation-to-this subject),* including relationships between verbs *(develop* is a partial case of *work).* We also need to be able to identify communicative actions and generalize them together with their subjects according to the specific patterns of speech act theory. Moreover, we need to maintain rhetoric structure relationship between sentences, to generalize at a higher level above sentences.

The focus of this paper will be to introduce parse thicket and their generalization as paragraph-level structured representation. It will be done with the help of the above example. Fig.1 and Fig.2 show the dependency-based parse trees for the above texts T1 and T2. Each tree node has labels as part-of-speech and its form (such as SG for 'single'); also, tree edges are labeled with the syntactic connection type (such as 'composite').

3 Introducing Parse Thickets

Is it possible to find more commonalities between these texts, treating parse trees at a higher level? For that we need to extend the syntactic relations between the nodes of the syntactic dependency parse trees towards more general text discourse relations.

Which relations can we add to the sum of parse trees to extend the match? Once we have such relations as "the same entity", "sub-entity", "super-entity" and anaphora, we can extend the notion of phrase to be matched between texts. Relations between the nodes of parse trees which are other than syntactic can merge phrases from different

sentences, or from a single sentence which are not syntactically connected. We will refer to such extended phrases as *thicket phrases.*

If we have to parse trees P_1 and P_2 of text T_1, and an arc for a relation r

r: $P_{1j} \rightarrow P_{2j}$ between the nodes P_{1j} and P_{2j}, we can now match $...,P_{1,i-2}$, $P_{1,\,i-1}$, $P_{1,\,i}$, $P_{2,j}$, $P_{2,j+1}$, $P_{2,j+2}$, ... of T_1 against a chunk of a single sentence of merged chunks of multiple sentences from T_2.

3.1 Phrase-level generalization

Although the generalization is defined as maximum common sub-trees, its computation is based on matching phrases. To generalize a pair of sentences, we perform chunking and extract all noun, verb, prepositional and other types of phrases from each sentence. Then we perform generalization for each type of phrases, attempting to find a maximum common sub-phrase for each pair of phrases of the same type. The resultant phrase-level generalization can then be interpreted as paths in resultant common sub-trees [3].

Generalization of parse thickets, being a maximal common sub-graph (sub-parse thicket) can be computed at the level of phrases as well, as a structure containing a maximal common sub-phrases. However, the notion of phrases is extended now: *thicket phrases* can contain regular phrases from different sentences. The way these phrases are extracted and formed depend on the source of non-syntactic link between words in different sentences: thicket phrases are formed in a different way for communicative actions and RST relations. Notice that the set of regular phrases for a parse thicket is a sub-set of the set of thicket phrases (all phrases extracted for generalization). Because of this richer set of phrases for generalization, the parse thicket generalization is richer than the pair-wise thicket generalization, and can better tackle variety in phrasings and writing styles, as well as distribution of information through sentences.

3.2 Algorithm for forming thicket phrases for generalization

We will now outline the algorithm of forming thicket phrases. Most categories of thicket arcs will be illustrated below.

For each sentence S in a paragraph P
  Form a list of previous sentences in a paragraph S_{prev}
  For each word in the current sentence:
    - If this word is a *pronoun*: find all nouns or noun phrases in the S_{prev} which are
      * The same entities (via anaphora resolution)
    - If this word is a *noun*: find all nouns or noun phrases in the S_{prev} which are
      * The same entities (via anaphora resolution)
      * Synonymous entity
      * Super entities
      * Sub and sibling entities

- If this word is a *verb*:
  * If it is a communicative action:
    Form the phrase for its subject $VBCA_{phrase}$, including its verb phrase Vph
    Find a preceding communicative action $VBCA_{phrase0}$ from S_{prev} with its subject
    and form a thicket phrase [$VBCA_{phrase}$, $VBCA_{phrase0}$]
  * If it indicates RST relation
    Form the phrase for the pair of phrases which are the subjects [$VBRST_{phrase1}$, $VBRST_{phrase2}$], of this RST relation, $VBRST_{phrase1}$ belongs to S_{prev}.

Notice the three categories of the formed thicket phrases:

- Regular phrases;
- Thicket phrases;
- SpActT phrases;
- CA phrases.

Once we collected the thicket phrases for texts T1 and T2, we can do the generalization. When we generalize thicket phrases from various categories, the following constraints should be taken into account:

	Regular phrases	Entity-based thicket phrases	RST-based thicket phrases	SpActT-based thicket phrases
Regular phrases	Obeying phrase type +	+	+	+
Entity-based thicket phrases	+	+	-	-
RST-based thicket phrases			+	-
SpActT-based thicket phrases				+

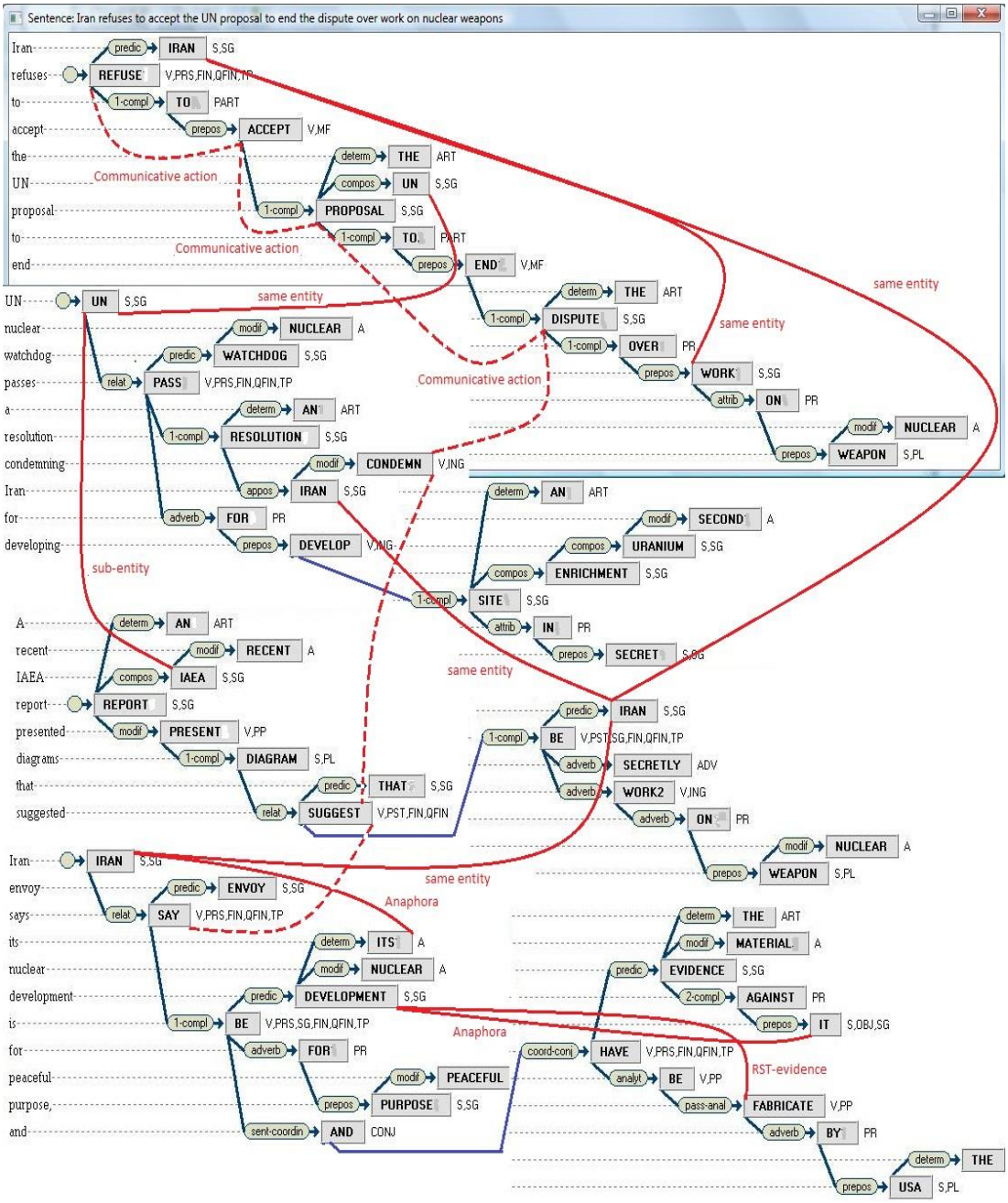

Fig. 1: Parse thicket for text T1.

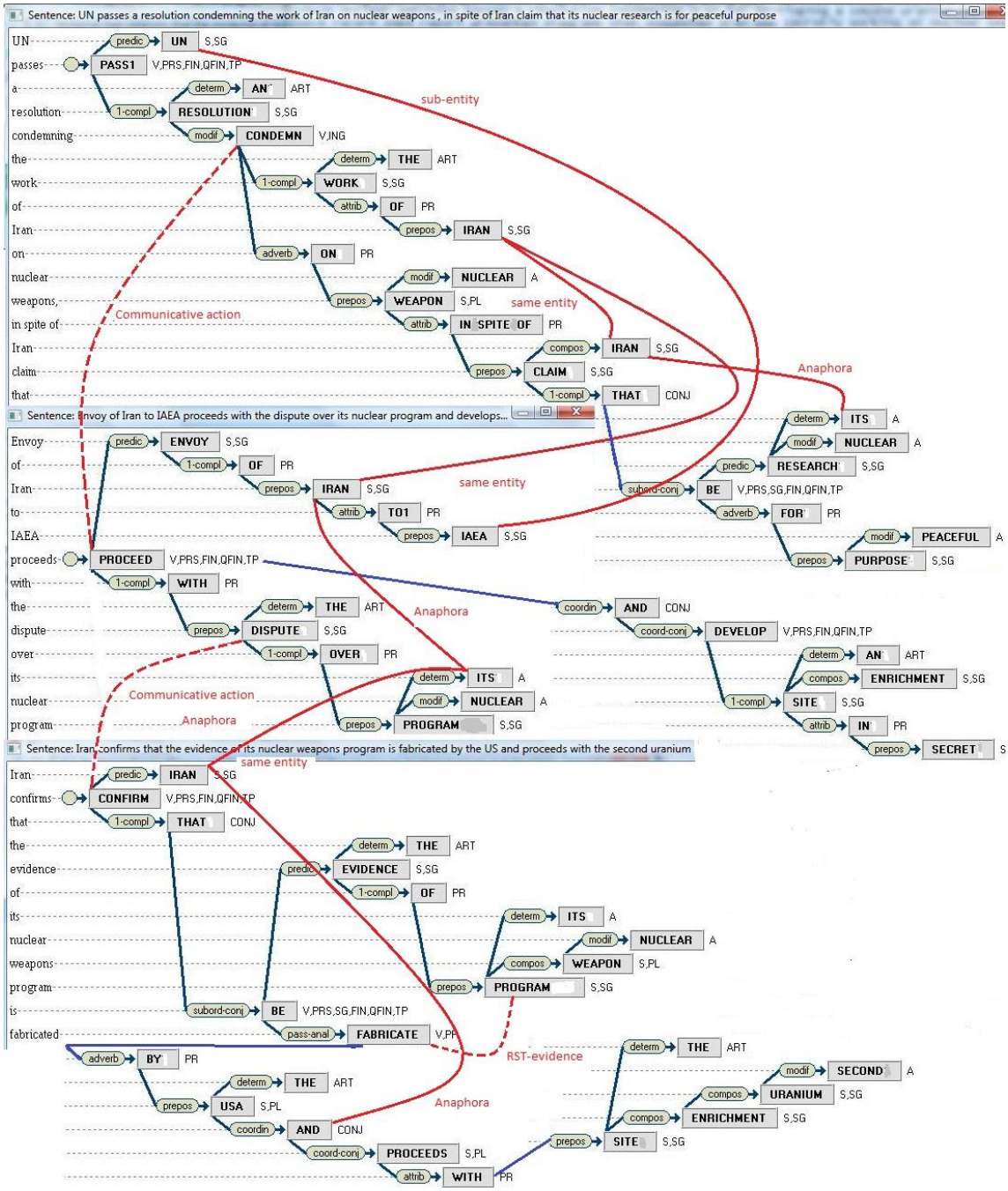

Fig. 2: Parse thicket for text T2.

3.3 Sentence-level generalization algorithm

Below we outline the algorithm on finding a maximal sub-phrase for a pair of phrases, applied to the sets of thicket phrases for T1 and T2.

1) Split parse trees for sentences into sub-trees which are phrases for each type: *verb*, *noun*, *prepositional* and others; these sub-trees are overlapping. The sub-trees are coded so that information about occurrence in the full tree is retained.
2) All sub-trees are grouped by phrase types.
3) Extending the list of phrases by adding equivalence transformations
4) Generalize each pair of sub-trees for both sentences for each phrase type.
5) For each pair of sub-trees yield an alignment, and then generalize each node for this alignment. For the obtained set of trees (generalization results), calculate the score.
6) For each pair of sub-trees for phrases, select the set of generalizations with highest score (least general).
7) Form the sets of generalizations for each phrase types whose elements are sets of generalizations for this type.
8) Filtering the list of generalization results: for the list of generalization for each phrase type, exclude more general elements from lists of generalization for given pair of phrases.

3.4 Arcs of parse thicket based on theories of discourse

We attempt to treat computationally, with a unified framework, two approaches to textual discourse:

- Rhetoric structure theory (RST, Mann et al 1992);
- Speech Act theory (SpActT, [16] 1969);

Although both these theories have psychological observation as foundations and are mostly of a non-computational nature, we will build a specific computational framework for them [4,5]. We will use these sources to find links between sentences to enhance indexing for search. For RST, we attempt to extract an RST relation, and form a thicket phrase around it, including a placeholder for RST relation itself [6]. For SpActT, we use a vocabulary of communicative actions to find their subjects [7], add respective arcs to PT, and form the respective set of thicket phrases.

3.5 Generalization based on RST arcs

Two connected clouds on the right of Fig.3 show the generalization instance based on RST relation “RCT-evidence”. This relation occurs between the phrases

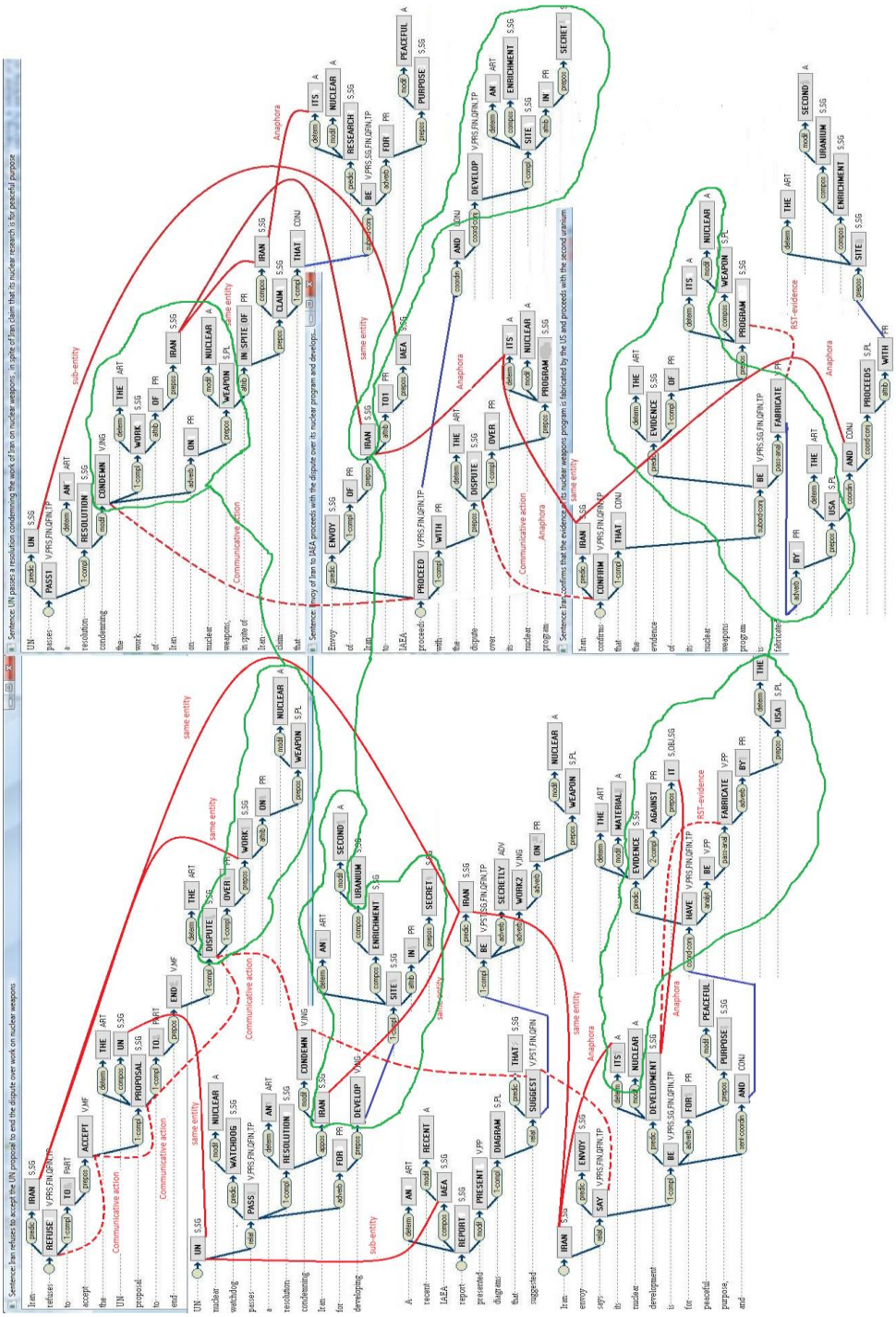

Fig.3: Three instances of matching between sub-PTs shown as connected clouds

evidence-for-what [Iran's nuclear weapon program] and *what-happens-with-evidence [Fabricated by USA]* on the right-bottom, and

evidence-for-what [against Iran's nuclear development] and *what-happens-with-evidence [Fabricated by the USA]* on the right-top.

Notice that in the latter case we need to merge (perform anaphora substitution) the phrase ' *its nuclear development'* with *'evidence against it'* to obtain *'evidence against its nuclear development'*. Notice the arc *it - development*, according to which this anaphora substitution occurred. *Evidence* is removed from the phrase because it is the indicator of RST relation, and we form the subject of this relation to match. Furthermore, we need another anaphora substitution *its- Iran* to obtain the final phrase.

As a result of generalizations of two RST relations of the same sort (evidence) we obtain

Iran nuclear NNP – RST-evidence – fabricate by USA.

Notice that we could not obtain this similarity expression by using sentence-level generalization.

Green clouds indicate the sub-PTs of T_1 and T_2 which are matched. We show three instances of PT generalization.

3.6 Generalization based on communicative action arcs

Communicative actions are used by text authors to indicate a structure of a dialogue or a conflict (Searle 1969). Hence analyzing the communicative actions' arcs of PT, one can find implicit similarities between texts. We can generalize:

1. one communicative actions from with its subject from T1 against another communicative action with its subject from T2 (communicative action arc is not used) ;
2. a pair of communicative actions with their subjects from T1 against another pair of communicative actions from T2 (communicative action arcs are used) .

In our example, we have the same communicative actions with subjects with low similarity:

condemn ['*Iran for developing second enrichment site in secret'*] vs *condemn* ['*the work of Iran on nuclear weapon*'] ,

or different communicative actions with similar subjects.

Looking on the left of Fig.3 one can observe two connected clouds: the two distinct communicative actions *dispute* and *condemn* have rather similar subjects: '*work on nuclear weapon'*. Generalizing two communicative actions with their subjects follows the rule: generalize communicative actions themselves, and 'attach' the result to generalization of their subjects as regular sub-tree generalization. Two communicative actions can always be generalized, which is not the case for their subjects: if their generalization result is empty, the generalization result of communicative actions with these subjects is empty too. The generalization result here for the case 1 above is:

condemn^dispute [work-Iran-on-nuclear-weapon].

Generalizing two different communicative actions is based on their attributes and is presented elsewhere [4].

T1 T2

condemn [second uranium enrichment site] ↔ *proceed [develop an enrichment site in secret]*

↓ communicative action arcs ↓

suggest [Iran is secretly working on nuclear weapon] ↔ *condemn [the work of Iran on nuclear weapon]*

which results in

condemn^proceed [*enrichment site*] <leads to> *suggest^condemn* [*work Iran nuclear weapon*]

Notice that generalization

condemn [second uranium enrichment site] ↔ *condemn [the work of Iran on nuclear weapon]*

↓ communicative action arcs ↓

suggest [Iran is secretly working on nuclear weapon] ↔ *proceed [develop an enrichment site in secret]*

gives zero result because the arguments of *condemn* from T1 and T2 are not very similar. Hence we generalize the subjects of communicative actions first before we generalize communicative actions themselves.

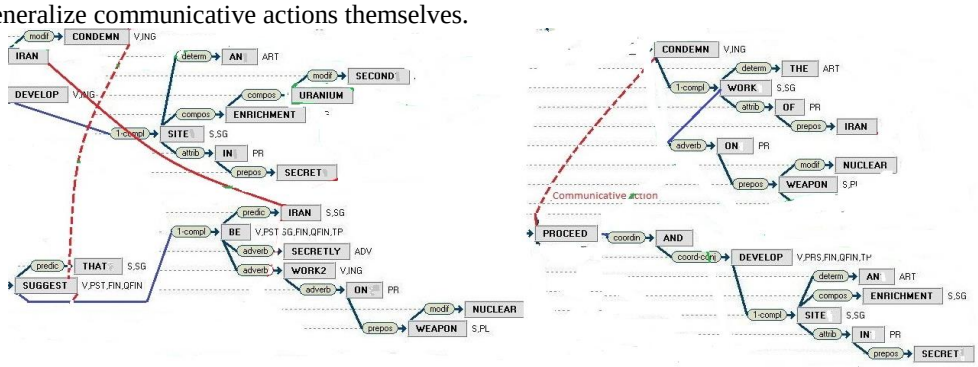

Fig.4: A fragment of PT showing the mapping for the pairs of communicative actions

4 Preliminary Evaluation of Parse Thicket generalization

Parse forests and their generalizations are important for domain-independent text relevance assessment. In our earlier studies we explored generalization of a PT against a single sentence: this is the case of question answering [4]. To find the best answers, we assess the similarity between the question and candidate answers represented as PTs [5] in the settings of eBay product search. In this section, we provide evaluation of these reduced cases of PT generalization, which can serve as a preliminary evaluation for the general case of PT generalization.

Query	Answer	Relevancy of baseline Yahoo search, %, averaging over 20 searches	Relevancy of baseline Bing search, %, averaging over 20 searches	Relevancy of re-sorting by pair-wise sentence generalization, % averaging over 40 searches	Relevancy of re-sorting by forest generalization based on RST, % averaging over 20 searches	Relevancy of re-sorting by forest generalization based on	Relevancy of re-sorting by hybrid RST+SpActT forest generalization % averaging over 40	Relevancy improvement for parse thicket approach, comp. to pair-wise generalization
3-4 word phrases	1 comp. sentence	81.7	82.4	86.6	88.0	87.2	91.3	1.054
	2 sent	79.2	79.9	82.6	86.2	84.9	89.7	1.086
	3 sent	76.7	75.0	79.1	85.4	86.2	88.9	1.124
	Average	79.2	79.1	82.8	86.5	86.1	90.0	1.087
5-10 word phrases	1 comp. sentence	78.2	77.7	83.2	87.2	84.5	88.3	1.061
	2 sent	76.3	75.8	80.3	82.4	83.2	87.9	1.095
	3 sent	74.2	74.9	77.4	81.3	80.9	82.5	1.066
	Average	76.2	76.1	80.3	83.6	82.9	86.2	1.074
1 sentence	1 comp. sent	77.3	76.9	81.1	85.9	86.2	88.9	1.096
	2 sent	74.5	73.8	78.	82.5	83.1	86.3	1.101
	3 sent	71.3	72.2	76.5	80.7	81.2	83.2	1.088
	Average	74.4	74.3	78.7	83.0	83.5	86.1	1.095
2 sentences	1 comp. sent	75.7	76.2	82.2	87.0	83.2	83.4	1.015
	2 sent	73.1	71.0	76.8	82.4	81.9	82.1	1.069
	3 sent	69.8	72.3	75.2	80.1	79.6	83.3	1.108
	Average	72.9	73.2	78.1	83.2	81.6	82.9	1.062
3 sentences	1 sentence	73.6	74.2	78.7	85.4	83.1	85.9	1.091
	2 sentences	73.8	71.7	76.3	84.3	83.2	84.2	1.104
	3 sentences	67.4	69.1	74.9	79.8	81.0	84.3	1.126
	Average	71.6	71.7	76.6	83.2	82.4	84.8	1.107
Average for all Query and Answer type								1.085

Table 1: Evaluation results for search where answers occur in different sentences.

Discovering trivial (in terms of search relevance) links between different sequences (such as coreferences) is not as important for search as finding more implicit links provided by text discourse theories. We separately measure search relevance when PT is RST-based and SpActT-based. Since these theories are the main sources for establishing non-trivial links between sentences, we limit ourselves to measuring the contributions of these sources of links. Our hybrid approach includes both these sources for links. We consider all cases of questions (phrase, one, two, and three sentences) and all cases of search results occurrences (compound sentence, two, and three sentences) and measured how PT improved the search relevance, compared to original search results ranking averaged for Yahoo and Bing.

One can see (Table 1) that even the simplest cases of short query and a single compound sentence gives more than 5% improvement. PT-based relevance improvement stays within 7-9% as query complexity increases by a few keywords, and then increases to 9-11% as query becomes one-two sentences. For the same query complexity, naturally, search accuracy decreases when more sentences are required for answering this query. However, contribution of PTs does not vary significantly with the number of sentences the answer occurs in (two or three).

While single-sentence syntactic match gives 5.6% improvement [4] multi-sentences parse thickets provides 8.7% for the comparable query complexity (5.4% for single-sentence answer) and up to 10% for the cases with more complex answers. One can see that parse thicket improves single sentence syntactic generalization by at least 2%. On average through the cases of Table 1, parse thickets outperforms single sentence syntactic generalization by 6.7%, whereas RST on its own gives 4.6% and SpActT-4.0% improvement respectively. Hybrid RST + SpActT gives 2.1% improvement over RST-only and 2.7% over SpActT only. We conclude that RST links compliment SpActT links to properly establish relations between entities in sentences for the purpose of search.

5 Conclusions

In this study we introduced the notion of syntactic generalization to learn from parse trees for a pair of sentences, and extended it to learning parse thickets for two paragraphs. Parse thicket is intended to represent syntactic structure of text as well as a number of semantic relations for the purpose of indexing for search. To accomplish this, parse thicket includes relations between words in different sentences, such that these relations are essential to match queries with portions of texts to serve as an answer.

We considered the following sources of relations between words in sentences: coreferences, taxonomic relations such as sub-entity, partial case, predicate for subject etc., rhetoric structure relation and speech acts. We demonstrated that search relevance can be improved, if search results are subject to confirmation by parse thicket generalization, when answers occur in multiple sentences.

Traditionally, machine learning of linguistic structures is limited to keyword forms and frequencies. At the same time, most theories of discourse are not computa-

tional, they model a particular set of relations between consecutive states. In this work we attempted to achieve the best of both worlds: learn complete parse tree information augmented with an adjustment of discourse theory allowing computational treatment.

Graphs have been used extensively to formalize ranking of NL texts [13]. Graph-based ranking algorithms are a way of deciding the importance of a vertex withina graph, based on global information recursively drawn from the entire graph. The basic idea implemented by a graph-based ranking model is that of "voting": when one vertex links to another one, it is basically casting a vote for that other vertex. In the current papers graphs are used for representation of meaning rather than for ranking; the latter naturally appears based on the similarity score.

We believe this is a pioneering study in learning a union of parse trees. Instead of using linguistic information of individual sentences, we can now compute text similarity at the level of paragraphs. We plan to extend the functionality of the similarity component of OpenNLP [14] by the contribution of PT-based algorithms.

References

1. Collins, M., and Duffy, N. 2002. Convolution kernels for natural language. In Proceedings of NIPS, 625–632.
2. Galitsky, B. Natural Language Question Answering System: Technique of Semantic Headers. Advanced Knowledge International, Australia (2003).
3. Galitsky, B., Josep Lluis de la Rosa, Gábor Dobrocsi. Inferring the semantic properties of sentences by mining syntactic parse trees. Data & Knowledge Engineering. Volume 81-82, November (2012) 21-45.
4. Galitsky, B., Daniel Usikov, Sergei O. Kuznetsov: Parse Thicket Representations for Answering Multi-sentence questions. 20th International Conference on Conceptual Structures, ICCS 2013 (2013).
5. Galitsky, B., G. Dobrocsi, J.L. de la Rosa, Kuznetsov, S.O.: From Generalization of Syntactic Parse Trees to Conceptual Graphs, in M. Croitoru, S. Ferré, D. Lukose (Eds.): Conceptual Structures: From Information to Intelligence, 18th International Conference on Conceptual Structures, ICCS 2010, Lecture Notes in Artificial Intelligence, vol. 6208, pp. 185-190.(2010)
6. Galitsky, B., Gabor Dobrocsi, Josep Lluís de la Rosa, Sergei O. Kuznetsov: Using Generalization of Syntactic Parse Trees for Taxonomy Capture on the Web. 19th International Conference on Conceptual Structures, ICCS 2011: 104-117 (2011).
7. Galitsky, B., Kuznetsov SO Learning communicative actions of conflicting human agents. J. Exp. Theor. Artif. Intell. 20(4): 277-317 (2008).
8. Galitsky,B., Machine Learning of Syntactic Parse Trees for Search and Classification of Text. Engineering Application of AI, http://dx.doi.org/10.1016/j.engappai.2012.09.017, (2012).

9. Ganter, B, Kuznetsov SO Pattern Structures and Their Projections. In: Conceptual Structures: Broadening the Base. Lecture Notes in Computer Science Volume 2120, 2001, pp 129-142.
10. Haussler, D. 1999. Convolution kernels on discrete structures.
11. Hensman, S. and Dunnion, J. Automatically building conceptual graphs using VerbNet and WordNet. International Symposium on Info and Comm. tech, Las Vegas, Nevada, June 16–18, 2004.
12. Marcu, D. (1997) 'From Discourse Structures to Text Summaries', in I. Mani and M.Maybury (eds) Proceedings of ACL Workshop on Intelligent Scalable Text Summarization, pp. 82–8, Madrid, Spain.
13. Mihalcea R., and Tarau P. 2004 TextRank: Bringing Order into Texts. Empirial Methods in NLP 2004.Punyakanok, V., Roth, D., & Yih, W. (2004). Mapping dependencies trees: an application to question answering. In: Proceedings of AI & Math, Florida, USA.
14. OpenNLP 2013. apache.org/opennlp/documentation/manual/opennlp.htm.
15. Polovina S. and John Heaton, "An Introduction to Conceptual Graphs," AI Expert, pp. 36-43, 1992.
16. Searle, John. 1969. Speech acts: An essay in the philosophy of language. Cambridge, England: Cambridge University.
17. Sowa JF, Eileen C. Way: Implementing a Semantic Interpreter Using Conceptual Graphs. IBM Journal of Research and Development 30(1): 57-69 (1986)
18. Sowa JF, Information Processing in Mind and Machine. Reading, MA: Addison-Wesley Publ., 1984.
19. Sun, J., Min Zhang, Chew Lim Tan. Tree Sequence Kernel for Natural Language. AAAI-25, 2011.
20. Zhang, M.; Che, W.; Zhou, G.; Aw, A.; Tan, C.; Liu, T.; and Li, S. 2008. Semantic role labeling using a grammar-driven convolution tree kernel. IEEE transactions on audio, speech, and language processing 16(7):1315–1329.