Visualizing Russian kinship term possessive sequences as family trees

Anna Golub
Faculty of Infocommunication Technologies
ITMO University
St. Petersburg, Russia
anna.golub.923@gmail.com

Alyona Belova
Faculty of Infocommunication Technologies
ITMO University
St. Petersburg, Russia
belova.alyona.itmo@gmail.com

Gleb Bondarenko
Faculty of Infocommunication Technologies
ITMO University
St. Petersburg, Russia
bondoll2001@mail.ru

Darya Karmaz
Faculty of Infocommunication Technologies
ITMO University
St. Petersburg, Russia
dasha.karmaz@yandex.ru

***Abstract*—As frequently as they are encountered in texts of various genres, Russian kinship term possessive sequences remain confusing even for the native speakers. The paper presents the authors' original computer science project, whose goal was to suggest a method of extracting such word sequences from a piece of text and visualizing them in an easily comprehensible way. Such an attempt of kinship relations analysis automatization might contribute to future research in history, linguistics, and literary studies and be of use to those studying Russian as a foreign language.**

Keywords—kinship term, possessive structure, family tree, natural language processing, Russian

I. Introduction

Russian possessive sequences including several kinship terms (e. g. *сестра мужа моей тёщи — my mother-in-law's husband's sister*, *бабушка шурина его брата — his brother's brother-in-law's grandma*) often appear confusing in written text as well as in oral speech since it is difficult to quickly calculate the relations between the relatives mentioned. Besides, such phrases often include names of relatives by marriage (*тёща* — a wife's mother, *шурин* — a wife's brother, etc.). Those kinship terms are becoming increasingly obsolescent in modern Russian as they only constitute for 1.4 per cent of all kinship term entries in Russian National Corpus [1] in 1990—2020. Therefore, appearing in a sequence, they make it even harder to comprehend.

The goal of the authors' original computer science project was to create a computational tool that would extract the sequences in question from a given piece of text and visualize them in an easily comprehensible way. Such an attempt of kinship relations analysis automatization might contribute to future research in history, linguistics, and literary studies, e. g. scientific analysis and systematization of fiction and memoirs. Moreover, this technology might be utilized by those studying Russian as a foreign language in order to ease the process of Russian kinship terms' meaning comprehension and memorization.

The following text processing stages were suggested for the computational tool:

1. finding kinship term possessive sequences in the given text;
2. analyzing the family relations that the each of the sequences presents and building graphs upon them;
3. visualizing the graphs as family trees using the existing tools for data visualization.

II. State of the Art

A. Kinship Term Possessive Sequences

As to our knowledge, a comprehensive solution to the problem has not yet been suggested. A great deal of research has been done on kinship term systems and the grammar of possession in various languages. However, possessive structures with kinship terms have not been subject to in-depth investigation. A few researchers touched on the topic of kinship terms in their studies of possessive structures (Dahl & Koptjevskaja-Tamm 2001 [2], Paykin & Van Peteghem 2003 [3], Jones 2010 [4]). Unfortunately, the approach taken was exclusively theoretical, and in addition, the only type of phrases discussed was those consisting of a possessive pronoun and a single kinship term, such as *her sister*. Aside from that, some papers describing data collected through fieldwork mention kinship term possessive sequences, but that kind of research does not seem applicable to building a tool for their computational processing.

B. Family Tree Visualization

In terms of visualization, there are many software tools that are suitable for depicting family trees. To begin with, specialized packages (e. g. ggenealogy [5]) provide plotting methods for genealogical data. However, the hierarchical nature of a structure does not allow horizontal edges and node skipping, which are required for valid representation of relationships in kinship term possessive sequences. This is accurate for the libraries with more general visualizing functionality (e. g. Graphviz [6], Plotly [7], Toytree [8]) as well. Furthermore, these visualization tools have rather limited customization options while the goal was to display confusing lineages in the most efficient way.

Additionally, to our knowledge, some other family tree visualization tools appropriate for the assigned task [9] are unfortunately unavailable for public use.

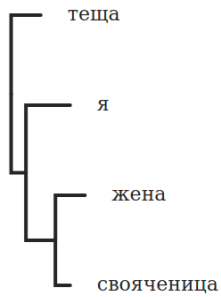


Fig. 1. Visualizing a family tree with Toytree

Moreover, there is genealogy software with intuitive family tree builders for illustrating pedigrees (e.g. Family Historian, Legacy Family Tree, etc. [10]). Despite the representation of kinship relationships in a comprehensive, visually organized manner, these applications require user interaction, which makes automatic visualization significantly more complicated.

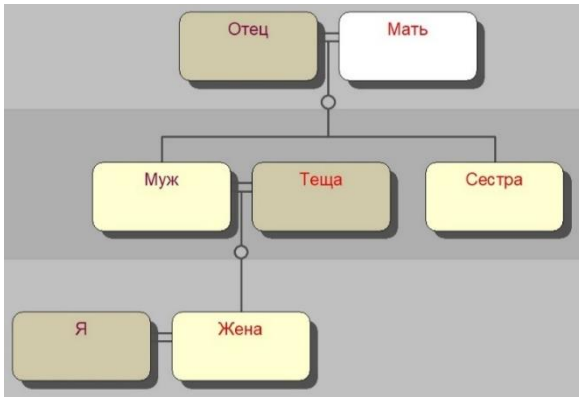


Fig. 2. Visualizing a kinship term sequence with Family Historian

Taking all of the aforementioned into account, we decided to use NetworkX [11] and Matplotlib [12] to develop our own visualization algorithm best fitting the requirements.

III. Solution

A. Text Search

All the source code is written in Python. At first, using NLTK, a Python library for natural language processing [13], the given piece of text is split into sentences; then each of them is tokenized into words. Next, employing pymorphy2 [14] for part-of-speech tagging and further morphological analysis, continuous word sequences are extracted from sentences. At this point the sequences consist of:

- one or more **kinship terms**, the first one of them in any case while all the rest in the genitive case exclusively;
- certain kinds of **modifiers**, namely long- or short-formed adjectives and participles, ordinal numerals and adjective pronouns;
- not more than one non-kinship **noun** in the genitive case. If included, this word is the last one in the sequence (see sequence type 4 below).

Afterwards, each of the sequences is recognized as one of the sequence types listed below, which depict most instances of kinship term possessive sequences in Russian. The GEN abbreviation stands for the genitive case while n signifies the possible number of the word's occurrences.

1. **possessive adjective / possessive pronoun**
   + **kinship term (GEN)**

   Example: *бабушкиному мужу*
   grandma's husband
   'grandma's husband'

2. **kinship term**
   + **kinship term (GEN)** $n = 0,1,2..$
   + **possessive adjective / possessive pronoun**
   + **kinship term (GEN)**

   Example: *муж бабушки моей сестры*
   husband grandma my sister
   'my sister's grandma's husband'

3. **kinship term**
   + **kinship term (GEN)** $n = 0,1,2..$
   + **possessive adjective / possessive pronoun**

   Example: *мужа бабушки сестры моей*
   husband grandma sister my
   'my sister's grandma's husband'

4. **kinship term**
   + **kinship term (GEN)** $n = 0,1,2..$
   + **noun (non-kinship) (GEN)**

   Example: *мужем бабушки подруги*
   husband grandma friend
   'friend's grandma's husband'

5. **kinship term**
   + **kinship term (GEN)** $n = 0,1,2..$

   Example: *мужу бабушки сестры*
   husband grandma sister's
   'sister's grandma's husband'

Then, the sequence is cast to the so-called normal form:

- kinship terms are put in the nominative singular form;
- non-kinship nouns are put in the nominative case with the number unchanged;
- possessive adjectives are replaced with their stem nouns in the nominative singular form;
- possessive pronouns are replaced with the corresponding personal ones in the nominative case (the forms of the *свой* pronoun are substituted by *некто*).

Afterwards, the sequence is reshuffled so as to put the words in the direct relation order (see examples below). If the sequence then starts with a kinship term, the first-person singular pronoun *я* is inserted into the beginning.

Examples:

a. *бабушкиному мужу — я бабушка муж*
grandma's husband — me grandma husband

b. *мужу бабушки сестры моей — я сестра*
husband grandma sister my — me sister
бабушка муж
grandma husband

Thus, all the words in the sequence, except for the first one, turn out to be kinship terms in the nominative singular form. Preprocessed this way, the sequence is fit for further analysis.

B. Kinship Relations Analysis

For each word in the sequence, except for the first one, the following actions are performed.

First, a graph fragment template is uploaded. The template presents the relationship between this and the previous character in the sequence. Here, by character we mean any relative in the chain of connections described by the sequence. Within the template all the characters are connected directly, either through **parent—child** or **wife—husband** type of connection. See the *сестра* (*sister*) template below as an example.

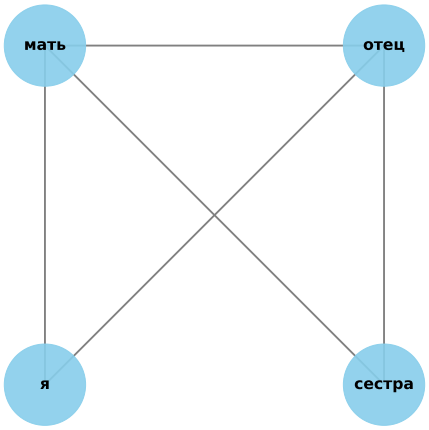


Fig. 3. "Сестра" ("sister") template

Next, the template is incorporated in the graph that has been built so far, namely the root of this template is aligned with the top of the previous one. Here, by the template top we mean the character signified by this template's corresponding word. In turn, the template's root is the character whose relation to the top of the template is being described by the template word. (In the *sister* template above *я* (*me*) is the root while *сестра* (*sister*) is the top.) Consequently, by the tree root we mean the root of the first template and by the tree top the top of the last template included in the graph. See example below: the *бабушка* (*grandmother*) template being aligned with the *сестра* (*sister*) template. (For now, we only discuss the maternal grandmother.)

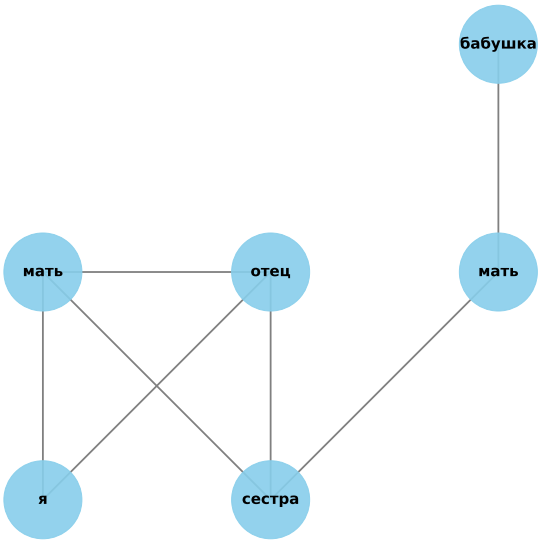


Fig. 4. Template alignment

Finally, if multiple nodes of the graph correspond to the same character of the sequence, they are merged into one. Such cases are unraveled based on the heuristic that each character may only have one mother and one father. The graph above is thus transformed into the following one.

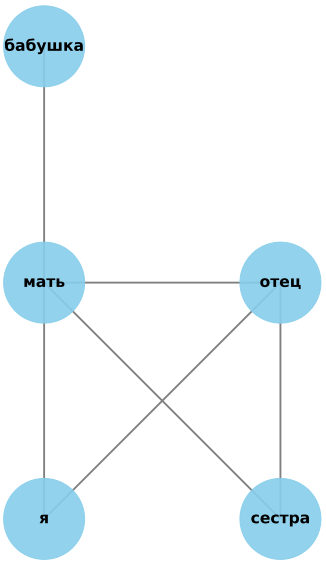


Fig. 5. The final graph structure

Thus, in the resulting graph all the characters are connected directly to each other. It is worth mentioning that due to linguistic polysemy, the relationship reflected in a kinship term might be described by a few different templates (e. g. in the above example the grandmother might be both maternal and paternal). Consequently, several graph structures are built, considering all the possible template combinations for the kinship terms in the sequence.

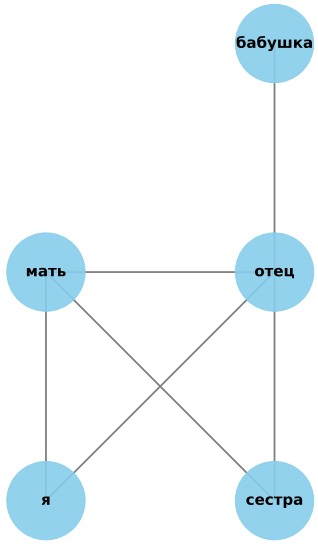


Fig. 6. The alternative graph structure

C. Graph Visualization

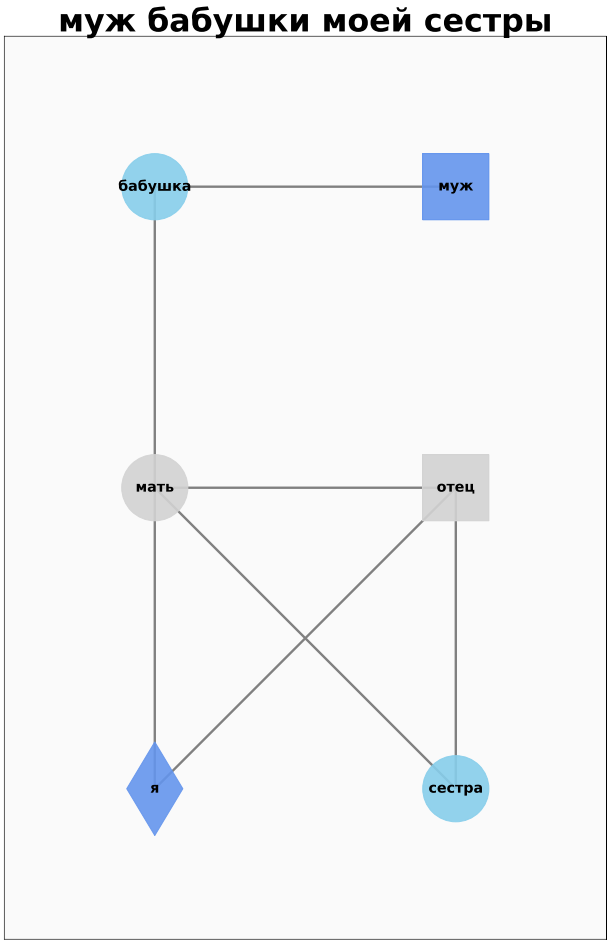
The family trees are drawn using NetworkX and Matplotlib in Python. All the edges of the graph are added to the list in the loop that goes through the characters and their connections. The root node gets zero coordinates, for other nodes the following rules apply:

- parent is positioned one point higher than their child;
- child is positioned one point lower than their parent;
- wife/husband is drawn at the same level and one point to the right from their spouse;
- if the determined spot is already taken, the node is shifted one point to the right.

As to colors, the following rules apply:

- for the tree top and the tree root nodes, blue color is used;
- otherwise, if the character is directly mentioned in the kinship term sequence, the node has a light blue color;
- if there is no direct mention of the character, the node is painted light gray.

The gender is displayed through the shape of the nodes: circles for females, squares for males; a rhombus is used for the root node. As a result, a PNG file presents the graph with a gray background, the kinship term sequence and the original sentence. The file is the final program output. See the visualization of the sequence *муж бабушки моей сестры* (*my sister's grandmother's husband*) as an example.



Я знал его с детства — это был муж бабушки моей сестры.

Fig. 7. The program output

IV. Evaluation

A. Evaluation Process

In order to evaluate the tool's performance, it was run on a purposefully collected corpus of texts, selected manually from the Russian National Corpus. The corpus consists of 3067 words and includes at least five entries of each of the kinship terms while keeping a rough balance between the sequence types.

For text search evaluation, each kinship term possessive sequence in the corpus was manually classified as follows:

- True Positive — the sequence was found, and its boundaries were identified correctly;
- False Positive — the sequence was found, but extra words were included;
- True Negative — there are no sequences in the sentence, and none were found;
- False Negative — the sequence was found, but some necessary words were excluded.

The precision and recall scores were then calculated, turning out 0.96 and 0.93 respectively.

Then, for each of the sequences found by the program the graphs were drawn to evaluate the kinship relations analysis and visualization. For each of the sequences, the number of expected and present correct visualizations was estimated manually with the resulting accuracy score being 0.95.

B. Discussion

As the evaluation test has outlined flaws in the tool's performance, a few areas for future work are suggested.

- Processing proper names. As for now, the tool cannot correctly process input sequences including first name + patronym collocations (e. g. *сын Анны Ивановны — Anna Ivanovna's son*) or abbreviated name forms (e. g. *сын Вл. Набокова — V. Nabokov's son*).
- Broadening the range of sequence types. For example, the tool cannot correctly process the sequence below because it does not fit any of the sequence type schemas.

  a. *шурина* *моего* *сын*
  wife's brother my son
  'my wife's brother's son'

- Adding context analysis features for better differentiation between the sequence types.
- Updating the template approach. At the relations analysis stage, complex kinship terms can be replaced with their simpler explanations, e. g. turning *тёща* (*mother-in-law*) into *мать жены* (*a wife's mother*), allowing to only store templates for the basic kinship terms, namely parents, children, siblings and spouses.
- Identifying coreference. At this point, the program does not register several words referring to the same character as in *сын моего отца* (*my dad's son*) and is unable to depict that in the graph.
- Making the tool adjustable for other languages by eradicating the language dependencies in the code.

V. Conclusion

This paper has presented kinship term possessive sequences as a field for natural language processing development, presenting the authors' original tool for the sequences' human-readable visualization. The program appears quite efficient and performs well on a broad range of input data; however, suggested paths for further development might push its limits significantly.

The project source code is available on github. The evaluation corpus, as well as the list of kinship terms processed by the program, can also be viewed there.

The tool is also available for public use as a Python package. As for now, the users are able to:

- extract sequences from a given piece of text;
- build a graph upon a given sequence;
- visualize an already-built graph or sequences from a given piece of unprocessed text.

References

[1] https://ruscorpora.ru/new/

[2] Dahl, Östen & Koptjevskaja-Tamm, Maria. (2001). 11. Kinship in grammar. 10.1075/tsl.47.12dah.

[3] Paykin, K., van Peteghem, M. External vs. Internal Possessor Structures and Inalienability in Russian. Russian Linguistics 27, 329–348 (2003).

[4] Jones, Doug. (2010). Human kinship, from conceptual structure to grammar. The Behavioral and brain sciences. 33. 367-404

[5] Rutter L, VanderPlas S, Cook D, Graham MA (2019). "ggenealogy: An R Package for Visualizing Genealogical Data." Journal of Statistical Software, 89(13), 1–31.

[6] "Graphviz and Dynagraph – Static and Dynamic Graph Drawing Tools", by John Ellson, Emden R. Gansner, Eleftherios Koutsofios, Stephen C. North, and Gordon Woodhull, in Jünger & Mutzel (2004).

[7] Plotly Technologies Inc. Collaborative data science. Montréal, QC, 2015. https://plot.ly

[8] https://toytree.readthedocs.io/en/latest/

[9] Borges, J. (2019). A contextual family tree visualization design. Information Visualization.

[10] https://www.toptenreviews.com/software/home/best-genealogy-software/

[11] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

[12] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.

[13] Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.

[14] Korobov M.: Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts, pp 320-332 (2015)