The Prague Bulletin of Mathematical Linguistics
NUMBER 107 APRIL 2017 57–66

Česílko Goes Open-source

Jernej Vičič,[a][b] Vladislav Kuboň,[c] Petr Homola[d]

[a] University of Primorska, The Andrej Marušič Institute
[b] Research Centre of the Slovenian Academy of Sciences and Arts, The Fran Ramovš Institute
[c] Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Charles University
[d] Semantek Ltd.

Abstract

The Machine Translation system Česílko has been developed as an answer to a growing need of translation and localization from one source language to many target languages. The system belongs to the shallow parse, shallow transfer RBMT paradigm and it is designed primarily for translation of related languages. The paper presents the architecture, the development design and the basic installation instructions of the translation system.

1. Introduction

The system Česílko (language data and software tools) was first developed as an answer to a growing need of translation and localization from one source language to many target languages. The starting system belonged to the Shallow Parse, Shallow Transfer Rule-Based Machine Translation – (RBMT) paradigm and it was designed primarily for translation of related languages. The latest implementation of the system uses a stochastic ranker; so technically it belongs to the hybrid machine translation paradigm, using stochastic methods combined with the traditional Shallow Transfer RBMT methods. The source code that is now published as open-source under the MIT license (The MIT License (n.d.), 2016) is almost the same as that which is presented in (Homola and Kuboň, 2008) with some slight modifications made to compile the code on GNU/Linux.

This article presents the architecture, the development design and the basic installation instructions of the translation system and is organised as follows. The state

Corresponding author: jernej.vicic@upr.si
Cite as: Jernej Vičič, Vladislav Kuboň, Petr Homola. Česílko Goes Open-source. The Prague Bulletin of Mathematical Linguistics No. 107, 2017, pp. 57–66. doi: 10.1515/pralin-2017-0004.

of the art is presented in Section 2, followed by the description of the history of the translation system in Section 3. Section 4 presents the outline of software architecture, followed by the description of the installation process in Section 5. The article concludes with a list of tested environments in Section 6 and a discussion and further work in Section 7.

2. State of the Art

The framework presented in this paper can be attributed to the paradigm of Fully Automatic Machine Translation (FAMT), which comprises every automatic translation of natural languages with no user intervention (”EAMT”, 2010). More specifically, the framework focuses on the translation of related languages, one of the most suitable paradigms for this domain is the Shallow Transfer Rule-Based Machine Translation. It has a long tradition and has been successfully used in a number of MT systems, some of which are listed in Section 2.1. Shallow-transfer systems usually use a relatively linear and straightforward architecture, where the analysis of a source language is usually limited to the morphemic level.

The latest version of Česílko uses a stochastic ranker, so technically it is a hybrid machine translation system framework, using stochastic methods combined with the traditional Shallow Transfer RBMT.

2.1. Existing MT Systems for Related Languages

A number of experiments in the domain of machine translation for related languages have led to the construction of more or less functional translation systems. The systems are ordered alphabetically:

- Altinas (Altintas and Cicekli, 2002) for Turkic languages.
- Apertium (Corbi-Bellot et al., 2005) for Romance languages.
- Dyvik, Bick and Ahrenberg (Dyvik, 1995; Bick and Nygaard, 2007; Ahrenberg and Holmqvist, 2004) for Scandinavian languages.
- Česílko (Hajič et al., 2000a), for Slavic languages with rich inflectional morphology, mostly language pairs with Czech language as a source.
- Ruslan (Oliva, 1989) full-fledged transfer based RBMT system from Czech to Russian.
- Scannell (Scannell, 2006) for Gaelic languages; between Irish (Gaeilge) and Scottish Gaelic (G‘aidhlig).
- Tyers (Tyers et al., 2009) for the North Sámi to Lule Sámi language pair.
- Guat (Vičič et al., 2016) for Slavic languages with rich inflectional morphology, mostly language pairs with Slovenian language.

3. The history of the MT system Česílko

The idea to develop an MT system for very closely related languages has actually been inspired by the request of the company SAP to support the localization of their products into Slavic languages. The original idea was relatively simple – the texts were supposed to be translated by human translators from the original languages (English and German) into Czech and then automatically translated into a number of related languages. The translations served as a support for human translators from the original languages into the target Slavic languages. The automatically translated texts were added to the translation memories, from which they were retrieved only in the event that no better translations already existed in the translation memory (this can easily be achieved by setting a penalty for machine translated texts in the translation memory). The details of this setup can be found for example in (Hajič et al., 2000a).

The actual architecture of the system called Česílko has been developed between the years 1998 and 2000, it was for the first time described in (Hajič et al., 2000b), more detailed description can be found in (Hajič et al., 2000a).

Figure 1 shows the architecture of the most popular translation system for related languages Apertium (Corbi-Bellot et al., 2005) and its predecessor, Česílko (Hajič et al., 2003), designed primarily for the translation between Slavic languages.

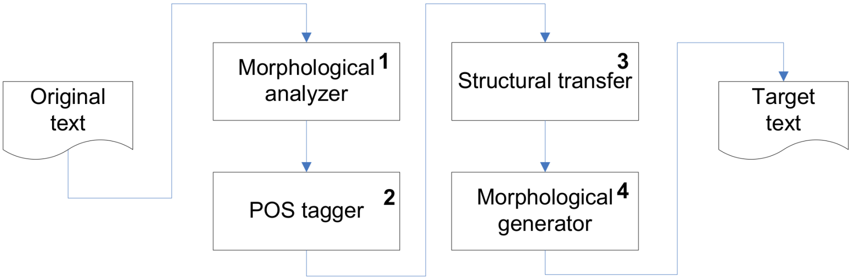

Figure 1. The modules of a typical Shallow Transfer RBMT system, this architecture was adopted in the first version of Česílko.

.

The system exploited the work from the previous MT system RUSLAN cf. (Oliva, 1989) and (Hajič, 1987), also aiming at the translation between related Slavic languages. Its development started in mid eighties, the system aimed at the automatic translation of texts from a limited domain (manuals of operating systems of mainframe computers). The system RUSLAN system was designed as a traditional transfer-based system with full morphological and syntactic analysis of Czech as a source language and the syntactic and morphological synthesis of Russian as the target language. The system was designed with the assumption that the close relatedness of both languages would primarily be reflected in the transfer phase and in the dictionary of the system. This assumption turned out to be wrong.

The transfer phase, originally very simple, had to be substantially enlarged in the process of testing the system. The simplicity of the transfer phase was achieved also due to the fact that the lexical transfer was not handled by an independent transfer module it was actually performed in the dictionary lookup phase, and the transfer module actually covered only structural transfer and primarily dealt with syntactic differences only. The subtle syntactic differences between Czech and Russian were still differences after all, and as such they had to be handled by specific transfer rules. The number of those specific transfer rules grew together with the amount of the text used for testing the system.

The lessons learned in this project clearly indicated that the strategy chosen for RUSLAN did not exploit the similarity of both languages to the desired extent and that the closeness of both related languages actually did not have a major positive effect on the quality of the results achieved. It was negatively influenced by the complexity of the system and the close relatedness of languages actually called for much simpler architecture.

Instead of the morphological similarity, the simplified architecture of the Česílko exploited the syntactic similarity of the related languages and also chose a more similar target language – Slovak. The syntactic analysis of the source language (Czech) was completely removed due to the assumption that both languages have in fact identical syntax (the existing differences being only marginal). A stochastic morphological tagger performed the disambiguating role of the syntactic analysis. It took the ambiguous information provided the morphological analysis module of Czech and provided a single morphological tag with the highest probability in the given context. The translation module then translated both the lemma provided by the morphological analysis and the tag (the target language morphology uses slightly different tagset and therefore the translation of the source language tag was necessary). This information was then exploited by the morphological synthesis module of the target language. No syntactic information was used for the synthesis of the target language, the system strongly relied on the syntactic similarity of both languages.

The results of the Czech-to-Slovak translations were good enough to justify further experiments (the translated text required less than 10% post-editing operations in order to obtain a high quality translation.). The next two target languages added were Polish and Lithuanian. The results of these experiments have been described in (Hajič et al., 2003). The experiments clearly showed that the most important phenomenon, which makes the automatic translation of related languages easier, is their syntactic similarity. From the lexical point of view, Lithuanian is much less similar to Czech than it is to Polish. Lithuanian also belongs to a different language family (Baltic languages), while Czech and Polish are both Western Slavic languages. On the other hand, the syntactic differences between Czech and Polish or Czech and Lithuanian are of a similar nature, and thus the fact that the translation results of these two language pairs are of a similar quality strongly supports the hypothesis that the syntactic similarity is the decisive factor in the MT of closely related languages.

Several other experiments with other target languages (e.g., Lower Sorbian, Macedonian, Russian etc.) have been performed in subsequent years. All these experiments, described for example in (Homola and Kuboň, 2004) and (Dvořák et al., 2006), showed that the more similar is the syntax of the source and the target language, the better are the translation results. It became clear that the better translation quality can be possible by changing the architecture through hybridization of the original architecture by the involvement of a stochastic ranker instead of the tagger. This substantial improvement of the architecture has been described both in the Ph.D. thesis of Petr Homola and in several articles such as (Homola and Kuboň, 2008) and (Homola and Kuboň, 2010). The change of the architecture actually improved the translation quality for all target languages, but, unfortunately, for less related ones, the improvement was only relatively small.

4. The architecture

The Česílko system has a very simple architecture. It exploits the close similarity of both languages at all linguistic levels. There is no full-fledged analysis of the source text, the system adopts a simplistic approach of ignoring syntactic differences and focusing on morphology and lexica. A partial (shallow) parser is implemented mainly to cope with possible high degree of morphological ambiguity present in a morphologically rich languages such as Czech or Slovenian. Figure 2 shows the architecture of the latest version of Česílko that is being published as open-source.

The translation system is organized as a pipeline of four programs each using the output of the preceding program. The morphological analyzer *morph* searches for all possible applications of the surface forms in the source morphological dictionary. The output of this module is fed to the shallow syntactical analyzer *syntan* implemented as a bottom-up chart parser. The formalism of Q-systems has turned out to suit the requirements although there are already plans to change the setting. The *transfer* module searches for the source – target lemma pairs in the bilingual dictionary, applies the changes in charts and later uses the target morphological dictionary to prepare the paths in the chart in the target language. The traversal of all possible paths through the chart gives a set of translation candidates. The output of the *transfer* ranked by a target language model – *ranker*, which is a simple trigram language model although the architecture allows a transparent change of the latest element. *Ranker* selects the best translation from the list of candidates according to the language model. The shallow parser produces highly ambiguous results because it is generally impossible to

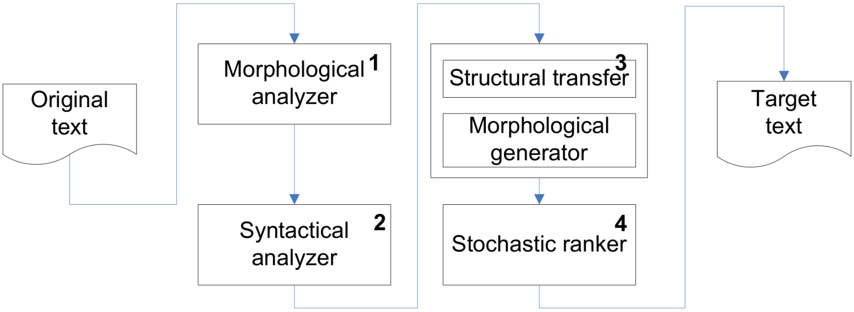

Figure 2. The modules of new Česílko system.

.

fully morphologically disambiguate the input sentences on the basis of local context only. The task of selecting the best result is left till the end of the processing chain, to a stochastic ranker of generated target language sentences. A simple trigram-based language model (trained on word forms without any morphological annotation) sorts out "wrong" target sentences. An extended description of the architecture and all modules can be found in (Homola and Kuboň, 2008).

4.1. Česílko strengths

One of the weaknesses of the shallow-transfer RBMT system is how they deal with the ambiguities introduced by the morphological analysis. The ambiguities can be eliminated with a set of rules using in a form of CG grammar (Karlsson et al., 1995) or using statistical POS taggers such as (Brants, 2000). Such architectures are presented in Figure 1. The errors introduced at the early phases of the translation pipeline have a big effect on the translation quality. Eliminating the modules leads to an explosion of the number of translation candidates (morphologically rich languages produces millions of candidates) (Vičič et al., 2009).

4.2. Translation quality evaluation

The evaluation of the translation quality was done previously in (Homola and Kuboň, 2008) and (Homola and Kuboň, 2010) on language data that is not part of the open-source distribution. The best results were obtained with the translation pair Czech – Slovak (only this direction), the results using the HTER (Snover et al., 2006) metric were 3.15 %.

5. Installation process

The latest version of Česílko was coded for OS X in Objective C, the reason for this was solely pragmatic (the main developer was using OS X). This experiment was mainly focused on making Česílko available as open-source and on porting Česílko to GNU/Linux and Windows, through Cygwin (Steinhauser, 2013) or MinGW (Shpigor, 2013) (Minimalist GNU for Windows). The port is based on GNUstep library/runtime (Chisnall, 2012). The GNU/Linux port comprised on adjusting the libraries and parts of the source code to accustom small changes in code (mainly just changing header files).

The source code and a test dataset is available on GitHub.[1] The test dataset supports the language pair Czech – Slovak, it is a small subset of the data used in (Homola and Kuboň, 2008). Following tools and libraries need to be installed on a fresh installation of Ubuntu in order to successfully compile and start Česílko:

- *git* – fast, scalable, distributed revision control system,
- *clang* – C, C++ and Objective-C compiler (LLVM based),
- *GNUstep Development Environment* – development tools,
- the latest version of the libobjC2 from GNUstep (not available in repository).

Following tools and libraries need to be installed on a fresh installation of mac OS in order to successfully compile and start Česílko:

- *Xcode* – Xcode is an integrated development environment (IDE).

A quick cheat-sheet of the installation on the Ubuntu operating system is presented in Figure 3. A script that installs the development environment (and many other things) enables easy install.

When all parts of the development environment are prepared, simply go to the code directory and start the make process: *make; make install*. A test translation pipeline is prepared in the Makefile. Start test target by typing: *make test*; a successful installation will present a translation of the test example (in Czech) into the Slovak language.

6. Tested environments

The code was successfully compiled and started (used) on these platforms:

- latest LTS editions of Ubuntu (Ubuntu 16.04 and 14.04). It was compiled with Ubuntu clang version 3.8.02ubuntu4 (tags/RELEASE_380/final).
- latest editions of the OS X El capitan and macOS Sierra. It was compiled with Apple LLVM version 7.0.2 (clang700.1.81).

[1] GitHub: `https://github.com/cesilko/cesilko`

```
sudo apt-get install git
# Copy an Objective-C installation script from
# http://wiki.gnustep.org/index.php/GNUstep_under_Ubuntu_Linux
# and start the script using root privileges.
git clone https://github.com/cesilko/cesilko
cd cesilko
make
make test
```

Figure 3. The installation steps for the build environment and Česílko. In the last command, the test target shows a typical usage with the toy dataset. For the OS X, install xcode and ignore the first three lines.

.

7. Discussion and further work

The paper has presented the history of the Shallow Parse/Transfer RBMT system Česílko which was transformed to a hybrid MT paradigm with the change in the architecture by the addition of a stochastic ranker. The system has been made available to the research community by open-sourcing the source code under MIT license (The MIT License (n.d.), 2016). At the moment the supported environment is GNU/Linux (tested on Ubuntu 16.04 amd 14.04 platform), although the code was developed on MacOS and compiles well on that operating system. The Windows platform is supported only using Cygwin or MinGW, so this is one of the first steps that need to be performed in the near future.

While the architecture of Česílko is really simple, it is modular and flexible so one can easily add new modules. One possible addition is a fully-fledged parser based on unification and a broad-coverage valency lexicon, which would allow for more distant language pairs. Another module being worked on is pragmatic interpretation of the source text, particularly the translation-by-abduction approach (Hobbs and Kameyama, 1990), which is planned for future versions instead of the statistical ranker to evaluate translation candidates on logical grounds.

Bibliography

Ahrenberg, Lars and Maria Holmqvist. Back to the Future? The Case for English-Swedish Direct Machine Translation. In *Proceedings of The Conference on Recent Advances in Scandinavian Machine Translation*. University of Uppsala, 2004.

Altintas, Kemal and Ilyas Cicekli. A Machine Translation System between a Pair of Closely Related Languages. In *Proceedings of the 17th International Symposium on Computer and Information Sciences (ISCIS 2002)*, page 5. CRC Press, 2002.

Bick, Eckhard and Lars Nygaard. Using Danish as a CG Interlingua: A Wide-Coverage Norwegian-English Machine Translation System. In *Proceedings of NODALIDA, Tartu*. University of Tartu, 2007.

Brants, Thorsten. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference*, pages 224–231. Seattle, WA, 2000.

Chisnall, David. A New Objective-C Runtime: From Research to Production. *Queue*, 10(7): 20:20—-20:24, 2012. ISSN 1542-7730. doi: 10.1145/2330087.2331170. URL `http://doi.acm.org/10.1145/2330087.2331170`.

Corbi-Bellot, Antonio M, Mikel L Forcada, and Sergio Ortiz-Rojas. An open-source shallow-transfer machine translation engine for the Romance languages of Spain. In *Proceedings of the EAMT conference*, pages 79–86. HITEC e.V., 2005.

Dvořák, Boštjan, Petr Homola, and Vladislav Kuboň. Exploiting Similarity in the {MT} into a Minority Language. In *Proceedings of the 5th {SALTMIL} Workshop on Minority Languages*, pages 59–64, Paris, France, 2006. European Language Resources Association. ISBN 2-9517408-2-4.

Dyvik, Helge. Exploiting Structural Similarities in Machine Translation. *Computers and Humanities*, 28:225–245, 1995.

"EAMT". European Association for Machine Translation, 2010. URL `http://www.eamt.org/`.

Hajič, Jan. RUSLAN: an MT System Between Closely Related Languages. In *Proceedings of the third conference of the European Chapter of the Association for Computational Linguistics*, pages 113–117. Association for Computational Linguistics, 1987.

Hajič, Jan, Jan Hric, and Vladislav Kuboň. Česílko – an MT system for closely related languages. In *Proceedings of ACL 2000*, pages 7–8, 2000a.

Hajič, Jan, Jan Hric, and Vladislav Kuboň. Machine translation of very close languages. In *Proceedings of the 6th Applied Natural Language Processing Conference*, pages 7–12. Association for Computational Linguistics, 2000b.

Hajič, Jan, Petr Homola, and Vladislav Kuboň. A simple multilingual machine translation system. In Hovy, Eduard and Elliott Macklovitch, editors, *Proceedings of the MT Summit IX*, pages 157–164, New Orleans, USA, 2003. AMTA.

Hobbs, Jerry R and Megumi Kameyama. Translation by Abduction. In *Proceedings of the 13th Conference on Computational Linguistics - Volume 3*, COLING '90, pages 155–161, Stroudsburg, PA, USA, 1990. Association for Computational Linguistics. ISBN 952-90-2028-7. doi: 10.3115/991146.991174. URL `http://dx.doi.org/10.3115/991146.991174`.

Homola, Petr and Vladislav Kuboň. A translation model for languages of acceding countries. In *Proceedings of the {EAMT} Workshop*, 2004.

Homola, Petr and Vladislav Kuboň. A method of hybrid MT for related languages. In *Proceedings of the IIS*, pages 269–278. Academic Publishing House EXIT, 2008.

Homola, Petr and Vladislav Kuboň. A Method of Hybrid MT for Related Languages. *Control and Cybernetics*, 39(2):421–438, 2010. ISSN 0324-8569.

Karlsson, Fred, Atro Voutilainen, Juha Heikkila, and Arto Anttila, editors. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, 1995. ISBN 3110141795. URL `http://portal.acm.org/citation.cfm?id=546590`.

Oliva, Karel. *A Parser for Czech Implemented in Systems Q*. MFF UK Prague, 1989.

Scannell, Kevin P. Machine translation for closely related language pairs. In *Proceedings of the Workshop Strategies for developing machine translation for minority languages*, pages 103–109. Genoa, Italy, 2006.

Shpigor, Ilya. *Instant MinGW Starter*. Packt Publishing, 2013.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231. Citeseer, AMTA, 2006. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.4369&amp;rep=rep1&amp;type=pdf`.

Steinhauser, Martin Oliver. Installation guide to Cygwin. In *Computer Simulation in Physics and Engineering*, chapter Appendix B, pages 445–447. Walter de Gruyter GmbH & Co. KG, 2013.

The MIT License (n.d.). The MIT License, 2016. URL `http://www.opensource.org/licenses/MIT`.

Tyers, Francis M, Linda Wiechetek, and Trond Trosterud. Developing prototypes for machine translation between two Sámi languages. In *Proceedings of EAMT*. HITEC e.V., 2009.

Vičič, Jernej, Petr Homola, and Vladislav Kuboň. A method to restrict the blow-up of hypotheses of a non-disambiguated shallow machine translation system. In *Proceedings of the RANLP*, pages 460–464, Borovec, Bulgaria, 2009. Association for Computational Linguistics.

Vičič, Jernej, Petr Homola, and Vladislav Kuboň. Automated implementation process of machine translation system for related languages. *Computing & Informatics*, 35(2):441 – 469, 2016.

Address for correspondence:
Jernej Vičič
`jernej.vicic@upr.si`
University of Primorska, UP IAM
Muzejski trg 2, 6000 Koper, Slovenia