Some Ideas on Using Grid Infrastructure for Natural Language Processing *

Radovan Garabík[1], Jan Jona Javoršek[2], and Tomaž Erjavec[2]

[1] Ľ. Štúr Institute of Linguistics, Slovak Academy of Sciences, Bratislava, Slovakia
garabik@kassiopeia.juls.savba.sk
[2] Jožef Stefan Institute, Ljubljana, Slovenia
jan.javorsek@ijs.si tomaz.erjavec@ijs.si

Abstract. The article discusses possibilities of using the Grid platform for Natural Language Processing tasks. Legal problems concerning distribution of copyrighted texts are described and possible solutions including encryption of data are outlined.

1 Introduction

Increasing computing requirements for acquiring and processing large data-sets and working with big corpora in Natural Language Processing (NLP) and related disciplines allow us to work on NLP algorithms and tasks that were impractical just a few years ago. The Grid network provides a good distributed environment for heavy-duty computational tasks, oriented towards the use by scientific community, and the possibility of using it for NLP-related tasks is obvious. Typical NLP tasks that could benefit from the Grid environment include language modelling (including algorithm and language model training, rather computationally demanding tasks), data conversion and indexing (especially when dealing with huge text corpora), and various kinds of language analysis. Some of these jobs can be easily parallelized or need to be run simultaneously many times with different parameters. Conversely, querying corpora is an interactive task that depends on low latency response and is best carried out by dedicated servers outside of the Grid environment.

2 Legal issues

The actual deployment of Grid computing in the natural language processing area faces specific legal issues – the data being processed are in majority of cases copyrighted, and the research institutions either have very strict legal agreements governing the use of the data, or are operating entirely on copyright law sections allowing scientific and research use of the data. The situation is somewhat similar to the problems the users of Grid computing in health care systems – though in that case, metadata are the most sensitive and protected part of the data-set, while in corpus linguistics the data (i. e. texts) are restricted, but the metadata is usually freely accessible.

* Parts of these article have been published in [GJE09]

In the extreme (but very frequent) case, the research institution using the data for research does not have the right to distribute the data at all. However, it might be still advantageous to use the Grid infrastructure for computing clusters of the institution itself, and use middleware functions to restrict data-replication to those processing nodes and data storage elements physically located in the organization. This way, the whole Grid can still be used for less sensitive tasks, or for post-processing the results of operations on sensitive data. It is nevertheless desirable to protect the data leaving the organization premises from casual snooping. Also, a measure of additional protection seems to be necessary – to avoid data leaking in case the computer hosting the Grid node is compromised, unbeknown to the administrators.

3 General requirements of NLP related tasks

Contemporary NLP tasks are rather varied; some of them require a lot of "pure" computing power, but many tasks, especially in the area of corpus linguistics, merely process large data files. From the software point of view, the tools used are very diverse – they are often programmed in typical computer languages, like C or C++, but a lot of data processing is done in scripting languages, such as Perl or Python, and Java is increasingly popular, and more often than not, one specific task uses several different tools bound by short programs written in a shell script. From this follows than the tools are often fragile and require a specific environment, which sometimes means that even using a different GNU/Linux distribution that the one the software has been developed on can be a major problem.

From Grid point of view, the best way to use the specific software is to install it inside a runtime environment which is made available to the jobs when submitted to the Grid. This is directly supported by the Grid infrastructure and requires no additional steps or privileges. However, at this time this requires a significant effort, since all the tools and their dependencies have to be compiled (or installed in a non-standard location inside the runtime environment) on the standard SLC distribution, which can be problematic if the software has many external dependencies.

4 Userspace and Full Virtualization

There are two possible solutions: to run under a chroot environment or to use virtualization, and several different approaches that lie somewhere in between those two extremes, ranging from paravirtualization, which requires cooperation from the guest operating system kernel, used e.g. by the XEN virtualization solution; to compartmentalization (e.g. Linux virtual servers and OpenVZ), which divides the host operating system into different compartments with completely separated processes, network access and file systems but sharing the same kernel; to vanilla kernel namespace support, which only separates user and process management (slightly extending chroot separation).

Many of the commonly used distributions already have support for (at least partial) installation inside a chroot environment built in. But in the context of Grid infrastructure this solution has a significant disadvantage, since it requires support from the cluster administrator because chroot environments are not a standard feature of the

Grid environment. Use of the chroot environment also does not support transparent encryption of the filesystem in such a way that the files are not directly accessible from the administrator of the host environment.

However, installing and using virtual machines requires not just administrator cooperation, but often also nonstandard host operating system extensions (such as special kernel modules). One of the more interesting virtualization systems in this context is User Mode Linux, which does not require any special host support, runs as an ordinary user process and provides a complete guest Linux kernel environment. Unfortunately, guest environment in this case suffers from a big I/O performance degradation, which can be a noticeable problem when dealing with very large corpus data.

The virtualization techniques mentioned differ on performance impact [PZW$^+$07] – ranging from none at all in case of a simple chroot or chroot with namespaces, over very little for `OpenVZ`-like compartmentalization to a more significant one for full virtualization. The specific areas of impact vary, too – while the raw CPU performance rarely decreases by more than a few percent (with the exception of complete software emulation of the guest architecture), I/O penalties are sometimes severe.

Using a complete virtual machine allows us to run a complete GNU/Linux distribution, with completely separate networking support and user management, including the ability to run processes with superuser privileges, and the ability to use filesystems otherwise not supported by the host system[3]. But the main advantage is the possibility to run completely different operating system. However, installing and using virtual machines requires not just administrator cooperation, but often also nonstandard host operating system extensions (such as special kernel modules). One of the more interesting virtualization systems in this context is User Mode Linux, which does *not* require any special host support, runs as an ordinary user process and provides a complete guest Linux kernel environment. Unfortunately, guest environment in this case suffers from a big I/O performance degradation, which can be a noticeable problem when dealing with very large corpus data.

While there is significant research in the use of different kinds of virtualization in the context of Grid technologies, unfortunately this is not a wide spread feature at this time. We have been able to use clusters with full support for chroot environemnts, but we realize that for quick adoption and widespread use of Grid computing in NLP, porting of tools to the most often supported environment, i.e. SLC, is currently necessary. There is ongoing research into different levels of virtualization with plans for support in the European grid infrastructure (EGI) in the future.

5 Data Access and Encryption

In practice, members of a research project or a discipline can set up a Virtual Organization (VO) and decide on its modes of operations and access to resources quite independently. They have to decide what kind of tools the VO members will be using in the Grid, define the data formats, prepare data repositories, develop execution environments with the tools installed and set up a Virtual Organization Membership Service server (VOMS server) to store authorization credentials.

[3] Such as encrypted filesystems.

Then some resources have to be made available to the community of VO members. In practice, that means obtaining support of a number of Grid sites (organizations owning computing clusters partaking in the Grid) that have to configure their Grid middleware installations to include the new VOMS server in its authorization procedures and to either install the execution environment (or, more realistically, environments) for the VO or give access to some members of the VO so that they can perform the installation and maintenance if the execution environment on the site themselves. Additionally, a number of Grid storage elements (SE) has to be configured to allow the VO members to access and store the data on their disk space.

The data has to be suitably protected where it is permanently stored. Therefore we propose to store the data in encrypted form in a dedicated storage element and set up the access authorization in such a way that access is restricted to VO users who belong in a VO group of users who signed the necessary legal agreements to access the data. Furthermore, we propose that the data is transferred to the untrusted environment of Grid worker nodes, where jobs perform their computations, in the encrypted form and that the decryption keys are issued to the jobs protected with asymmetric encryption decryptable only by the job's Grid proxy keys so that only the jobs can access the keys and decrypt the data.

In this manner, access and decryption is regulated with the authorization of embedded VOMS attributes in the proxy certificate without any additional authorization steps, while the data is never shipped or stored in unencrypted form. If the tools used by the job have to store temporary files on disk, these are protected from other processes (with the exception of system administrators, who are already bound by strong agreements pertaining to data security on the Grid) and are in addition of short-lived nature. The simplest implementation of the system described involves the use of a decryption filter in the job script and is rather simple to deploy. A more flexible solution, based on CryptoSRM (cryptographic storage resource manager) and Hydra Key Storage (a distributed fragmented encryption key storage system) is described in [SK08].

6 Description of NLP Tasks at the Jožef Stefan Institute

This section reports on a number of tasks carried out at the Jožef Stefan Institute, Ljubljana, Slovenia. We have performed these task first as a proof of concept and then as an analysis of expected use-cases. The groundwork covered has been used since in day-to-day NLP tasks by various users.

6.1 FidaPLUS Corpus Re-Taging

The first use-case considered was the automated annotation or morpho-syntactic tagging of FidaPLUS, a corpus of modern Slovene (621 million words). Our experiment was based on ToTaLe, an automated multilingual annotator [EIPS05]. While FidaPLUS is, naturally, fully marked-up and annotated, we have used ToTaLe to re-tag the corpus with a newly developed tag-set JOS [EK08].

The tagger is implemented partially in perl in partially in C. We encountered no problems with porting to the target environment (Scientific Linux 5 was used). We had to develop a number of scripts for conversion of the existing corpus files from annotated XML to plain text input as expected by the tagger, and decided that the transformation would not be done beforehand but would be carried out in the context of each grid job.

The next step was the organization of our jobs. Each job spends a certain amount of time waiting to be scheduled, waiting for data downloads etc. In addition, the software might require some time to start processing, which is true with the ToTaLe parser, which spends some time processing and loading its models, a procedure repeated for each job start. It is therefore appropriate to make the jobs long enough and as few as possible to avoid loosing too much of the time elapsed to these delays. On the other hand, every job has a certain probability of error at the instantiation, and there is a certain probability of hardware or software difficulties occurring during the run-time of the job, which usually forces the system to terminate and reschedule the job on another worker node.

These limitations mean that one should avoid overly short jobs (to avoid the scheduling overhead), to many jobs (to avoid too many job-start failures) and to long jobs (to lessen the probability of job restarts). Experience has shown that jobs that run between 2 and 10 hours, with 4 as an optimum, are the most efficient, but this measure depends partly on the local setup of the cluster.

Since most files in the corpus are too short to warrant an entire job, we decided to group the files into small jobs. We developed a script that created a task for us: a number of job description files and scripts to run the converter and annotator on groups of files and collect the results. In this manner, a parallel solution for corpus annotation with ToTale was easily implemented with two scripts and some set-up files, and then used with the entire FidaPLUS corpus.

In our case, we found that dividing the $\sim$44 000 files of FidaPLUS into 630 jobs, 70 files each, gave us jobs with manageable durations, albeit slightly on the long side. The mean execution time for jobs had been around 8 hours, and approximately one additional hour had been spent waiting for a free worker node in the scheduling queue. The whole process had been completed in under 12 hours. Over 6500 hours of computing time were used in the process, and 70 GB of morpho-syntactically annotated corpus produced.

Practical applications of this service, particularly since ToTaLe already supports several MULTEX-East languages and tag-sets and continues to support more languages (cf. [Erj04]) are obvious.

6.2 *n*-gram Processing

The second use case we undertook was an example of *n*-gram statistics. In this case, we collected the frequencies for 2-grams for the whole FidaPLUS corpus separately for words, lemmas and MSDs (tags), representing in total a corpus of 1863 million tokens.

In this use-case, the computational load per file was much lower, and the start-up time for the jobs was much shorter (we used Ted Pedersen's *n*-gram statistics package for Perl [BP03]). On the other hand, since now a single job could process a much larger amount of data, data transfer penalty was slightly more noticeable. But we encountered a different problem: the process of merging the resulting data into the final counts proved represent a rather more time-consuming task than first expected. To facilitate this problem, we have made each task perform the merging of its constituents' *n*-gram counts as part of its run, with a number of final jobs for the merging of the counts produced from the first batch.

We have divided the task in 90 jobs, each processing and combining 500 files. The individual jobs had a mean run time of 2.5 hours. A final 2 hour job has been needed to combine the final results. The whole task had been accomplished under 6 hours, consuming less than 100 hours of computing time.

Clearly, there is room for much improvement here: since we could not perform more interesting *n*-gram counts with a higher *n*, since the *n*-gram statistics package simply lacks the facility for combining such counts efficiently. The problem is being resolved, however.

The *n*-gram statistics produced during the experiment were later used in the development of a statistical chunker for Slovene.

7 Description of NLP Tasks in the Slovak National Corpus

This section analyzes some of the tasks commonly carried out at the Slovak National Corpus of the Ľ. Štúr Institute of Linguistics, with an emphasis on the deployability of Grid computing. Only those activities where the utilization of the Grid platform could be beneficial are described.

7.1 Slovak National Corpus Conversion and Tagging

Slovak National Corpus construction is a multilevel process. First there is document conversion, where source documents are converted into a common text format. This is usually being done continuously while adding new documents into the corpus (often, the conversion process has to be tuned to particularities of the given document). Then comes the morphosyntactic tagging, which is done while preparing a new major revision of the corpus (usually once per year). Since the tagger has usually improved in the meantime, we use this opportunity to re-tag the whole corpus (the version being prepared at the time of writing contains over 700 million tokens).

Morphosyntactic tagging of the Slovak National Corpus consists of two steps. The first performs morphosyntactic analysis, where each word in the input texts is assigned a set of possible morphosyntactic tags. This step essentially consists of looking up the possibilities of lemma/tag combinations in a constant database table using the wordform as a key, with an additional step for unknown words, where the list of possible tags is derived from the similarities of word endings to the ones present in the database tables. The software is implemented in the Python programming language and is actually quite fast, since the core of the task consists simply of a look-up in the possibilities in the tables, and most of the CPU work is spent on I/O operations, parsing the input file and assembling the output. On a modest hardware (Intel Xeon 2.33 GHz CPU) it is able to process over 10 000 words per second. It can also parallelize easily, since the words can be analyzed independently of each other.

The second step is disambiguation, where each word is assigned a unique lemma and a morphosyntactic tag out of the possibilities assigned in the first step. For disambiguation, we use *morče*, an averaged perceptron model originally used for the Czech language tagging [SHRS09], re-trained on the Slovak manually annotated corpus. Disambiguation is much slower that the morphology analysis, its average speed reaches only about 300 words per second. Parallelization at the application level is also not possible without some redesign of the *morče* itself, but the nature of tagging makes it easy to split the input data into as many chunks as we want and run *morče* instantiations in parallel.

Given the speed differences between morphology analysis and disambiguation, we can safely consider the morphology analysis execution time negligible and design the whole tagging to be done in one step, without the need to parallelize the morphology analysis process while the disambiguation is to be run in parallel. The whole re-tagging takes about a month of CPU time, and would greatly benefit from being run in parallel in the Grid environment. Unfortunately, the data in the corpus are the most restricted in the sense of copyright access – until there is a reliable way of encryption, we cannot even contemplate the possibility of letting the data get into the Grid network.

The source archives are 46 GB, however, a substantial percentage of this are original scan images (when converted into raw XML text, the size is about 6 GB uncompressed). We could benefit from storing the archive in the Grid data storage, however the license conditions do not allow this possibility.

Where the Grid could be used without any of the above mentioned problems is the planned project of Slovak Web corpus, i.e. corpus from Slovak language texts collected from the internet (predominantly from the WWW). Even if – technically – we cannot distribute copies of the texts, the sheer fact of their free availability takes away rather strict necessity of data protection, since the texts will neither be interesting enough to warrant effort in breaking the protection schemes deployed, nor will their copying result in disclosing data that was not publicly available in the first place. Since we expect the Web corpus to be several times of the size of current Slovak National Corpus, its morphosyntactic tagging will benefit greatly from the computing power of the Grid.

7.2 Corpus of Spoken Slovak

Corpus of Spoken Slovak[GR07] is a project to record reasonable amount of sound samples of contemporary Slovak, together with their phonemic transcription, lemmatization and morphosyntactic analysis. At the time of writing, the corpus contains about 160 hours of sound recordings, corresponding to 1.2 million tokens. Since the transcription is done manually (no reasonably accurate transcription software exists), the remaining task of morphosyntactic analysis is exactly the same as with the Slovak National Corpus texts.

The archive is kept in FLAC format, and we convert the whole recordings into Ogg Vorbis and Speex formats (for easier handling and transcription) and for the final linking through the corpus web interface we split the files into small chunks corresponding to dialogue turns. The conversion process is easily parallelized (the files are independent and the conversions can be run concurrently – in fact, we do the conversions when adding new files into the archive, and usually there is no need to re-run the whole conversion, unless we change encoding parameters, such as sound quality/bitrate). Since the archive size is currently over 200 GB, data storage capabilities and network throughput are more important than raw CPU power.

Since we ourselves recorded most of the content of the corpus, we took great care to ensure that there are no copyright problems with the corpus and the corpus could be made available without any limiting conditions. Of course, with sound recordings we have to take care not only of copyright law, but also the law on protection of personal data [Ná05]. We do this by censoring any sensitive information (e.g. persons' names) before including the recordings in the archive, and by including only those recordings where each of the parties present agrees on including the recording in the corpus. Therefore the spoken corpus could be stored, analyzed and made available on the Grid platform, without further legal complications, and Grid could be useful for the whole process.

8 Conclusion

In order to truly exploit the Grid potential, we envisage a scheme where the linguistic data (especially text corpora) are stored on the Grid infrastructure as well and the existing Grid access control infrastructure is extended in order to provide secure access to the data to third parties interested in accessing the data in a way that ensures that all the limitations and conditions arising from the copyright law and other binding agreements are met. For this, it might be necessary to devise a standardized system for efficient storage of sensitive data and controlled access to the data.

Our vision for ergonomic use of the Grid infrastructure is to be able to create independent virtual environments, at least based on chroot-environments, but preferably closer to full virtual machines with an arbitrary GNU/Linux distribution inside (or even with other operating systems), optionally with encrypted base file system image, with transparent access to encrypted data residing at the Grid storage (in some limited and hackish manner, this is possible to achieve with existing tools – using Usermode linux as a virtual machine not requiring any elevated privileges on the host system, and combination of hostfs and encryptfs to access the encrypted data on the host file system). Taking the idea one step further, the virtual machines can present to the guest operating systems a massively parallel multiprocessing environment that the guest applications can take advantage of and that are distributed across many Grid nodes in the real hardware.

References

[BP03] Satanjeev Banerjee and Ted Pedersen, *The Design, Implementation, and Use of the Ngram Statistics Package*, vol. 2588, January 2003.

[EIPS05] Tomaž Erjavec, Camelia Ignat, Bruno Pouliquen, and Ralf Steinberger, *Massive multi-lingual corpus compilation:Acquis Communautaire and totale*, Archives of Control Sciences **15** (2005), 529–540.

[EK08] Tomaž Erjavec and Simon Krek, *The JOS morphosyntactically tagged corpus of Slovene*, The 6th International Conference on Language Resources and Evaluation, ELRA, May 2008.

[Erj04] Tomaž Erjavec, *MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora*, Fourth International Conference on

Language Resources and Evaluation, LREC'04, ELRA, 2004, pp. 1535–1538.

[GJE09] Radovan Garabík, Jan Jona Javoršek, and Tomaž Erjavec, *Evaluating Grid Infrastructure for Natural Language Processing.*, NLP, Corpus Linguistics, Corpus Based Research. (Brno) (Jana Levická and Radovan Garabík, eds.), Tribun, 2009.

[GR07] Radovan Garabík and Milan Rusko, *Corpus of Spoken Slovak Language*, Computer Treatment of Slavic and East European Languages (Brno) (Jana Levická and Radovan Garabík, eds.), Tribun, 2007.

[Ná05] Národná rada Slovenskej republiky, *Zákon č. 428/2002 Z. z. o ochrane osobných údajov Z. z. v znení zákona č. 602/2003 Z. z., zákona č. 576/2004 Z. z. a zákona č. 90/2005 Z. z.*, *Zbierka zákonov Slovenskej republiky*, Bratislava, Slovakia, 2002, 2004, 2005.

[PZW$^+$07] Pradeep Padala, Xiaoyun Zhu, Zhikui Wang, Sharad Singhal, and Kang G. Shin, *Performance evaluation of virtualization technologies for server consolidation*, Tech. report, HP Laboratories, 2007.

[SHRS09] Drahomíra Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta, *Semi-supervised training for the averaged perceptron POS tagger*, EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 2009, pp. 763–771.

[SK08] Nuno Santos and Birger Koblitz, *Security in distributed metadata catalogues*, Concurrency and Computation: Practice and Experience **20** (2008), no. 17, 1995–2007.