Experimental Deployment of a Grid Virtual Organization for Human Language Technologies

Jan Jona Javoršek, Tomaž Erjavec

Jožef Stefan Institute
Jamova ulica 39,
SI-1000 Ljubljana, Slovenia
jan.javorsek@ijs.si, tomaz.erjavec@ijs.si

Abstract

After a brief overview of the elements of modern grid computing, a number of common use-cases of natural language processing tasks running on the grid are presented, notably corpus annotation with morpho-syntactic tagging (600+ million-word corpus in one day), n-gram statistics processing of a corpus and web-accessible services with annotation and term-extraction as examples. Implementation considerations and common problems of using grid for this type of tasks are laid out. Finally, a simple action plan for evolving the infrastructure created for these experiments into a fully functional Human Language Technology grid Virtual Organization is given with the goal to make the power of European grid infrastructure available to the linguistic community.

1. Introduction

We have set to port a number of natural language processing tasks to use a modern grid infrastructure and tested a number of assumptions regarding the benefits of the use of grid infrastructure in the context of human language technologies. This allowed us to present a concise overview of architectures and solutions needed to adapt linguistic resources and tools to be used inside grid environment and to assure that the grid environment can provide the services needed to securely access these resources and carry out computationally intensive or resource-intensive linguistic tasks in a secure and timely manner.

The motivation for the present work stems from the endless increases in computing and storage requirements for compilation and processing of large textual data-sets and working with larger and larger annotated corpora in Human Language Technologies (HLT) and related disciplines. This increases go beyond the increases of computing power in individual machines, but map well to the current development in wide-scale distributed computing architectures. We decided to test how we could use grid infrastructure, a widely deployed parallel computing infrastructure, to enable us to better deal with the increasing demands. More so, we believe that the availability of computing and storage resources that are an order of magnitude more powerful than what is currently available to most linguistic researches not only permits researches to increase the size of the data-sets used in their linguistic work, but also to develop and use new methods and better statistical models that could take advantage of the new computing resources. In addition, since the infrastructure permits much faster processing of existing data-sets, it also permits faster testing and experimental cycles for developers and researches, thus leading to new tools, models and hypotheses, enabling researches to delve into areas previously not practical to research.

The fields where we expect the most immediate benefits are, in our opinion, corpus compilation, annotation and processing, parsing and parser development, textual analysis, term extraction and tools development.

2. Previous Work

A number of existing papers have treated the possibilities of using grid technology for natural language processing. Tamburini (2004) presents an overview of the area, Carroll et al. (2005) discuss key challenges of a 'grid-enabled' NLP platform, there is considerable work in Portugal (Luís et al., 2008; de Matos et al., 2008; Martins de Matos et al., 2008; Martins de Matos and Ribeiro, 2008; Marujo et al., 2008). Also, there are several related on-going projects, notably within Clarin and in the German Text Grid project (Neuroth et al., 2009), a part of D-Grid Project (Neuroth et al., 2007).

The common ground of the existing work is the evaluation of the existing infrastructure and testing of the infrastructure in the context of natural language processing and human language technologies. However, in general there is a lack of concrete proposals and examples for creation of the needed infrastructural components to enable the use of grid technology in the field of computational linguistics.

We hope our experimental virtual organization of human language technologies could become such a proposal.

3. Grid, Tasks and Jobs

Grid computing implements a distributed network of computing resources accessible through a special software API layer: the grid middleware. While originally it was to be used with existing computing resources, it has become mostly used to support large-scale parallel computing using dedicated computing centres. A grid network can be construed as a form of distributed parallel computing, where a "virtual supercluster" is composed of loosely-coupled computing clusters just as a computing cluster is composed of individual networked machines. The parallel goes deep, in fact, since the features needed to orchestrate operations of a computing cluster are present on the grid level as well: grid managers schedule jobs and metadata servers take care of data storage just as job schedulers and storage elements perform these tasks on a regular computing cluster. In this manner, the system is used like a large computing cluster, where the grid middleware layer takes care of the imple-

mentation details: the user submits jobs to the system and the middleware uses information on available resources to find the needed data and computing resources and submit jobs to the actual centres where they are handed over to worker nodes, the computers that perform the work for the user. The user then uses the middleware to monitor the status of the jobs and access the results—either downloading the data from the system or storing it in the system for further processing and analysis.

Compared to a single computing cluster, there is an important additional circumstance: clusters of a computing grid (usually referred to as "grid sites") are owned by different organizations, possibly working in different research fields, and located in different regions or states: grid infrastructure is unified only by its middleware layer that connects computing and storage resources to central resource mangers that keep track of the use and availability of the infrastructure. The middleware connects a computation request of a user (instructions on what program and data to use, usually called a 'job') to the central resource managers, and reports on success or failure and location of results.

3.1. Authentication: Certificates

To enable this kind of resource sharing, a scalable method of authentication and authorization is needed. Computing grids use a combination of public key infrastructure (PKI) digital certificates for authentication and virtual organizations (VO) for authorization.

The mechanics of PKI as an authentication mechanism are well known and will not be discussed in this paper beyond a short recapitulation: a digital certificate is issued to a concrete person based on strict personal identification procedures and then paired with a private key known only to that individual. The certificate is then signed by the secret key of the issuing organization. A signature by a specific secret key can be verified by its matching public key which is made part of the certificate. In this way, a service only needs to trust the issuers certificate to verify the authenticity of a certificate, while the certificate itself contains the information on the individual. Hosts and services have similar certificates, issued for the entity in question to a previously identified administrator. This set-up is very effective in large-scale distributed networks such as the grid infrastructure since a partial failure of the network does not impair the authentication mechanism.

3.2. Auhorization: VOs and Certificate Proxies

The second element of grid security mechanism is authorization—the process where an authenticated user or service is granted or refused certain specific rights to data or resources. The authorization mechanism is the central task of grid Virtual Organizations (VOs), provided by VO Management Servers (VOMS). VOMS servers have databases of users, their permitted roles and associated rights. The information can be downloaded to a server and used as a user database (usually in the form of a grid-map file which also defines mappings between grid users and local UNIX user names), but usually it is used in a more scalable manner: the user connects to the VOMS server with a special middleware command and asks the server for a proxy certificate with a specific role. The server issues the certificate, signing the VOMS attribute extension element, and the tool signs the certificate with the users certificate, establishing a chain of trust. Proxy certificates are short-lived and usually used for submitting jobs and running short-lived services since they permit a user to delegate his identity and his role to the job or service, thus enabling it to access the data and resources in accordance with the user's role in the VO.

This mechanism is, again, fully scalable in the distributed environment, while providing the means for fine-grained control of access to data and resources and full audit logging of said access. In the context of linguistic computation, the mechanism can be used to protect sensitive data, such as copy-protected corpora, from non-authorized access even when processing at foreign grid sites.

3.3. Role of Virtual Organizations

In most research disciplines and international research projects the Virtual Organization usually grows beyond mere means of authorization control: it becomes the technical organizational nucleus around which the gridification of tools and development of new methods, techniques and resources for the discipline or project is centered. The VO becomes the point of development of a project- or discipline-oriented distributed platform and in addition manages support agreements with grid site administrators that are prepared to allow the VO and its users to access their computing resources or research data, such as corpora.

A VO usually also presents a number of tools to its community to help manage data and jobs. One of the most important VO tools is to allow users to define 'tasks', recipes that generate suitable job commands to process a certain data set, that can invoke thousands of jobs and let the users use the infrastructure on a much larger scale and in an organized way. VO tools take task descriptions and split them up into a number of jobs according to predefined rules or specifications, freeing the user from intimate knowledge of grid performance and implementation issues, greatly simplifying the process parallelization of the workload.

4. HLT VO: An Experimental Set-Up

To enable us to test the grid infrastructure as a platform for large-scale human language technology experiments, we have created a virtual organization: HLT VO. In the hope of developing the VO into a full-blown virtual organization, we have set up a VOMS server and some of the basic parts of support infrastructure needed for the experiments.

To enable grid processing, one must first ensure that on all grid sites (i.e. clusters) that support a VO, a suitable software platform is available at each worker node (computer running the job). Such a platform is called an execution environment. We have developed a testing execution environment running inside Scientific Linux CERN v5 distribution of GNU/Linux (SLC5, the upcoming standard for European grid infrastructure) and made sure that suitable software packages for our experiments were installed and functional inside it.

To provide flexibility, we have used an advanced solution for managing execution environments, enabling us to run

the environment in a semi-virtualized mode under a chroot. In this way, a central VO-hosted environment could be exported to the worker nodes and used for the jobs without any modification to the worker nodes themselves. This environment is hosted by the VO and installed at the grid sites that support the VO, in the case of our experiments on our local grid site with over 1000 cores and considerable storage capacity. Since the set-up does not require any modification to the worker nodes, the installation is very light-weight and easily sustainable. We have tested it under NorduGrid ARC grid middleware, but and adaptation to the widely used gLite middleware is straight forward.

Secondly, a remote storage facility has to be available for jobs to retrieve the data to be processed and to store the results. There is a well-defined standard for remote grid storage, namely Storage Resource Manager protocol (SRM). We used dCache, one of the more flexible implementations of the SRM protocol, for our site, since the infrastructure was already available.

Usually, a VO also provides a meta-data catalogue to enable its users to locate the data using logical names regardless of the storage location in the grid. There are several different implementations with different APIs, but they are all used to manage multiple distributed storage systems, possibly with replication of data. Since only a single storage element was used in our experiments, we have omitted this step.

Finally, a way to divide up the workload to parallelize ti had to be provided. Since the data in most linguistic use cases is not interdependent, the usual approach can be used where the data is split up in approximately equal chunks and each chunk is submitted with one job. Such a collection of grid jobs related by their purpose and data source is called a task. Usually, VOs develop suitable tools for creation and management of tasks and jobs. As it has been the purpose of these experiments, among other things, to find most suitable ways of managing corpus data to structure jobs and tasks, we have developed a number of scripts to handle this work automatically. A more capable system using the meta-data server and live grid resource information will have to be developed for the use of HLT VO community. Usually, a web interface for task creation, job submission, job status tracking and data retrieval is also provided by the VO.

This set-up was sufficient to use the VO and our grid site to run a number of use-cases with typical natural language processing and corpus compilation jobs that might benefit from a distributed parallel processing environment.

5. Use-Cases

We have treated three common use-cases that were considered to have a large user-base and were considered suitable for the parallel grid infrastructure.

There are, of course, many use-cases and tool implementations that do not lend easily to parallelization. Some problems can be solved with tool re-engineering, but often such use-cases simply require sequential processing or a different processing architecture. Solving such problems is a complex task and is beyond the scope of this paper, where we will concentrate on some cases where the existing solutions could be put to good use.

5.1. Morphosyntactic Annotation (Tagging)

The first use-case considered was an automated annotation or morpho-syntactic tagging of a corpus. Tagging is a time-consuming and computationally intensive task that can be performed in parallel and as such is a good match from the distributed architecture of the grid infrastructure.

Our experiment was based on ToTaLe, an automated multilingual annotator (Erjavec et al., 2005). We have used ToTaLe to re-tag FidaPLUS, a corpus of modern Slovene (621 million words), with a newly developed tag-set JOS (Erjavec and Krek, 2008). The tagger is implemented partially in perl in partially in C, but presented no problems with porting to the target environment. We had to develop a number of scripts for conversion of the existing corpus files from annotated XML to plain text input for the tagger, which enabled us to perform test runs of the process. Most files were too short to warrant an entire job, so we decided to group then into small jobs. We proceeded to develop a scripted solution to create a number of job description files and scripts to run the converter and annotator on several files. In this manner we implemented a parallel solution for corpus annotation with ToTale and used it with the entire FidaPLUS corpus.

In our case, we found that dividing the 44.000 files of FidaPLUS into 630 jobs processing 70 files each was the most efficient considering the number of available worker nodes at our grid site. The mean execution time for jobs had been around 8 hours, and approximately one hour per job has been spent waiting for a free worker node in the scheduling queue due to the high load at the grid site at the time of the experiments. The whole process has been completed in under 12 hours, and in over 6500 hours of computing time 70 GB of morpho-syntactically annotated corpus data has been produced.

While deciding on the way to split up the data and divide the task into multiple jobs, we had to consider the time it takes to get a job running, which depends on the current usage pattern of the grid sites used and has been, in our case, considerable. Due to this metric, it pays to combine a large amount of data into a longer running job, but in fact 8 hours is rather long: experience shows that it is best to keep job times at around 4 to 5 hours to minimize loss of time due to problems with failing nodes or jobs which can be noticeable with large tasks with several thousands of jobs, but can produce a considerable delay in our small-scale set-up, too.

Practical applications of this service, particularly having in mind that ToTaLe already supports several MULTEX-East languages and tag-sets and continues to support more languages (Erjavec, 2004)), are obvious to most linguistic users.

5.2. n-gram Processing

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

In this use-case, the computational load was much lower for the n-gram counting itself, and the start-up time for

the jobs was much shorter (we used Ted Pedersen's n-gram statistics package for Perl, (Banerjee and Pedersen, 2003). On the other hand, since now a single job could process a much larger amount of data, data transfer penalty was slightly more noticeable. In addition, the task of merging the resulting data from different files into the final counts proved to outmatch the supplied scripts since they had to rescan the counts file for every input file, store it in memory and combine with the next file, so that the jobs quickly ran out of RAM. We solved the problem by changing the file format in a sorted one so that we could combine the results while incrementally reading the file. Interestingly, in a new release of Ted Pedersen's package in March 2010, the count-merging scripts have been replaced with a more complete implementation of a similar algorithm that we used for our final runs.

We have divided the task in 90 jobs, processing and combining 500 files per job. 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 (the remaining run-time having been lost to waiting in the scheduling queue).

Clearly there is room for much improvement here. We could not perform more interesting n-gram counts with a higher n, since the n-gram statistics package simply lacks a facility for combining such counts efficiently. In addition, the process could be optimized if the merging process would be organized in the form of a directed graph to use the available resources optimally. But even in this crude form, usability improvement from the parallelization speed-up was impressive.

5.3. APIs and Web Interfaces

Our third use case experiment was based on the idea that users would want to combine existing grid-enabled tools and offer them as services. Such slightly more complex set-ups web interfaces and external APIs to task managing servers would allow jobs and tasks to talk to each other, forming job chains as a response to a request from a web servers. This technique enables us to make some tools easier to use, without exposing the end-users to the complexity of parallelization and the grid middleware hidden behind the web page.

The most demanding problem for implementation of such services is the authorization requirement, since the service has to be entrusted with a grid-certificate to be able to submit jobs. (Rules for the creation of such services are quite strict but well defined by the work of the International Grid Trust Federation, so this technical detail is not covert further here.)

We have used two web interface examples to explore the possibilities, one based on the work done for the ToTaLe web interface, and the second for a term-extraction service that used existing n-gram statistics tools and ToTaLe to extract terms from user-supplied documents.

In our experimental set-ups, the main problem turned out to be the job submission delay, since it can take a lot of time for significant numbers of jobs to get scheduled and to start running due to existing load or I/O traffic at available grid sites. The submission delay makes normal job submission impractical for interactive interfaces since the user is faced with a long period of inactivity immediately after a new request is submitted to the service.

There are existing solutions, however, that make it possible to eliminate or diminish the effect of submission delay and make grid infrastructure much more responsive, albeit at the cost of minor additional resource consumption. To avoid the lag, we need to run a special job (so-called 'pilot job') at each grid site that we want to use with our application. This job is the application's controller at the grid site: it monitors current demands on the application and submits a number of seed jobs to the site. The seed jobs in fact reserve some worker nodes where they wait instructions from the application that will start the actual computation. This set-up cuts out the submission lag and permits almost instant response from the grid.

This kind of solutions have been extensively tested in medical applications and have also recently become used in the High Energy Physics community. They are also flexible and scalable, so we feel confident this technique can be used to provide highly responsive externally accessible services for human language technologies.

6. HLT VO: Requirements

In order to establish a sustainable HLT grid infrastructure in the framework of a Virtual Organization, the work performed at our grid centre is only a beginning, although the virtual organization is online and availble at `http://www.htlvo.org/`.

6.1. Gridification of Tools and Resources

Technically, we need to establish a central repository for development of grid-avare tools and helpers for parallelization of existing tasks. In order to do this effectively, we first need to establish contacts and collect use-cases within a considerable part of our community to find the best practices that will make the infrastructure as accessible as possible to a large number of researchers.

We also need to enable organizations, both technically and organizationally, to make their linguistic resources (such as corpora) available to the community and the HLT Virtual Organization. Only when we have standardized access to each-other's data sets will duplication of results become practical and productive.

6.2. Membership and Site Support

We also need to get a sufficient number of grid sites to support the VO (currently only SiGNET and Slovenian National Grid Initiative project, although there are foreign users) and register the VO at the international level with the European Grid Initiative. This is only possible if sufficient interest within the community of potential users is expressed to insure support of their institutions, their national grid initiatives and the owners of major corpora and other linguistic resources for their language.

6.3. Central Services and Data Management

A web interface for task managing and resource supervision/accounting services for the VO will need to be established, possibly based on existing solutions. This is crucial

to enable users to track the progress of their tasks and access data without the need for considerable knowledge of grid middleware and command-line interfaces.

The most important element of the VO infrastructure that is still missing is data storage and meta-data services. A number of different solutions is available, but in the case of HLT VO, we have very strict requirements regarding access control enforcement for non-free resources, such as commercial corpora and corpora containing non-free texts. The solution should be based on existing PKI infrastructure with an encryption layer for data protection. We have considered existing work in the medical field where due to privacy concerns the issue of access control has been well studied, for example with the combination of dCache and AMGA, the Arda Metadata Catalogue Project, for meta-storage (Santos and Koblitz, 2008), modified with additional access controls, as decribed by Piparo (2008).

We are considering the details of such an implementation but find it necessary to finish the design with a wider community cooperation to satisfy the different requirements due to different access restrictions to corpora. Some experimental work in this department has been described in a previous paper (Garabík et al., 2009).

7. Conclusions

We believe that the case for use of the grid infrastructure as a platform for human language technology research has been well supported with the presented use-cases. With the wide availability of the infrastructure in Europe, the necessary technical elements for wide adoption of grid infrastructure in human language technologies are present. We are therefore convinced that a grid HLT VO has very high usability potential, above all because of the fact that it would enable considerable shortening of development and experimental cycles, thus opening the way for new methods, approaches and data-sets.

In conclusion, we would like to point out additional benefits of virtual organizations, as demonstrated by the success of the common grid infrastructure in Europe. Joint management of tools, resources and assets, enabled by the concept of virtual organizations, has been shown to facilitate international collaboration, management of joint projects and thus better research. In this spirit, we hope that collaboration inside a Human Language Technologies Virtual Organization could lead to easier cooperation, development of better tools and better accessibility of linguistic resources for researchers.

8. References

Satanjeev Banerjee and Ted Pedersen. 2003. *The Design, Implementation, and Use of the N-Gram Statistics Package*, volume 2588. January.

John Carroll, Roger Evans, and Ewan Klein. 2005. Supporting text mining for e-science: the challenges for grid-enabled natural language processing. In *Proceedings of the UK e-Science All Hands Meeting*.

David M. de Matos, Tiago Luís, and Ricardo D. Ribeiro. 2008. Natural language engineering on a computational grid (NLE-GRID) T1 - architectural model. Technical Report 38 / 2008, INESC-ID, Lisboa.

Tomaž Erjavec and Jan J. Javoršek. 2008. Grid infrastructure requirements for supporting research activities in digital lexicography. In *Mondilex: Lexicographic Tools and Techniques*, pages 5–13. IITP RAS.

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

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

Tomaž Erjavec. 2004. MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *Fourth International Conference on Language Resources and Evaluation, LREC'04*, pages 1535–1538. ELRA.

Radovan Garabík, Jan J. Javoršek, and Tomaž Erjavec. 2009. Evaluating grid infrastructure for natural language processing. In *NLP, Corpus Linguistics, Corpus Based Grammar Research: Proceedings of the 5th International Conference Slovko*, pages 93–105. Tribun EU.

Tiago Luís, David Martins de Matos, Sérgio Paulo, and Ricardo D. Ribeiro. 2008. Natural language engineering on a computational grid (NLE-GRID) T5 - performance experiments. Technical Report 35 / 2008, INESC-ID, Lisboa.

David Martins de Matos and Ricardo D. Ribeiro. 2008. Natural language engineering on a computational grid (NLE-GRID) T2h - encapsulation of reusable components: Lexicon repository and server, january 2008. Technical Report 32 / 2008, INESC-ID, Lisboa.

David Martins de Matos, Ricardo D. Ribeiro, Sérgio Paulo, Fernando Batista, Luísa Coheur, and Joana P. Pardal. 2008. Natural language engineering on a computational grid (NLE-GRID) T2 - encapsulation of reusable components. Technical Report 31 / 2008, INESC-ID, Lisboa.

Luis Marujo, Wang Lin, and David Martins de Matos. 2008. Natural language engineering on a computational grid (NLE-GRID) T3 - multi-component application builder. Technical Report 33 / 2008, INESC-ID, Lisboa.

Heike Neuroth, Martina Kerzel, and Wolfgang Gentzsch, editors. 2007. *German Grid Initiative D-Grid*.

Heike Neuroth, Fotis Jannidis, Andrea Rapp, and Felix Lohmeier. 2009. Virtuelle Forschungsumgebungen für e-Humanities. Massnahmen zur optimalen Unterstützung von Forschungsprozessen in den Geisteswissenschaften. In *Bibliothek. Forschung und Praxis, 2/2009*.

Nuno Santos and Birger Koblitz. 2008. Security in distributed metadata catalogues. *Concurrency and Computation: Practice and Experience*, 20(17):1995–2007.

Fabio Tamburini. 2004. Building distributed language resources by grid computing. In *Proc. of the 4th International Language Resources and Evaluation Conference*, pages 1217–1220.