READ

RECOGNITION & ENRICHMENT
OF ARCHIVAL DOCUMENTS

D4.16
HPC Integration and Maintenance

Philip Kahle, Sebastian Colutto, Günter Hackl, Günter Mühlberger
UIBK

Distribution: http://read.transkribus.eu/

READ
H2020 Project 674943

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 674943

Project ref no.	H2020 674943
Project acronym	READ
Project full title	Recognition and Enrichment of Archival Documents
Instrument	H2020-EINFRA-2015-1
Thematic priority	EINFRA-9-2015 - e-Infrastructures for virtual research environments (VRE)
Start date/duration	01 January 2016 / 42 Months

Distribution	Public
Contract. date of delivery	31.12.2016
Actual date of delivery	28.12.2016
Date of last update	21.12.2016
Deliverable number	D4.16
Deliverable title	HPC Integration and Maintenance
Type	Report
Status & version	Final
Contributing WP(s)	WP4
Responsible beneficiary	UIBK
Other contributors	URO
Internal reviewers	Gundram Leifert, Hervé Dejean
Author(s)	Philip Kahle, Sebastian Colutto, Günter Hackl, Günter Mühlberger
EC project officer	Martin Majek
Keywords	High Performance Computing, HPC, Transkribus

Contents

1 Executive Summary

This deliverable outlines the progress of task 4.6, HPC integration and maintenance, which consists of two subtasks: tools and software libraries have to be adapted in order to be able to utilize an HPC cluster for massive parallelisation. Second, HPC infrastructure has to be included in the READ platform ecosystem, i.e. Transkribus. Both, UIBK and URO have access to an HPC cluster and thus using this resource pool is an obvious undertaking.
The following is divided into three sections: the beginning describes the different HPC usage strategies in Transkribus, while the second part summarizes the progress on tool adaptation for this scenario. In the end, the requirements for HPC integration in the READ platform and the current stage of affairs is specified.

2 HPC Usage in Transkribus

There are two types of tools that are candidates for HPC usage within Transkribus: the first group of tools has to be adapted in order to exploit HPC resources, i.e. the software itself breaks down the task as there is the need for resources, shared amongst each thread. This is for instance the case with HTR training, where several pages serve as input data but those have to be handled as a whole.
The second group of tools works on small units of input data, that can be produced by Transkribus, and the parallelisation can be accomplished on that level, i.e. a process is divided into to the finest granularity possible and then several instances of the tool are executed in parallel. An example for this use case is layout analysis on whole document image sets, where the respective software can concurrently work on all pages at once. Thereby, one instance of the tool does not need to be informed about the outcome of other instances and thus the parallelisation is not done in the tool itself but by Transkribus and is merely a topic to process management.

3 Tool Adaptation

As already mentioned, the most obvious candidate for adaptation is the training of new HTR models, such as recurrent neural networks (RNN) and hidden markov models (HMM). The technical partners providing this technology are UPVLC and URO whose software shall be discussed here with respect to HPC compatibility.

3.1 UPVLC HMM HTR

The HTR engine by UPVLC, based on hidden markov models, is generally already compliant with HPC systems, this counts for the training as well as the recognition. During the TranScriptorium project efforts have been made to port the respective start scripts to the Son of Grid Engine[1], which is available on the HPC cluster LEO3 at the Univer-

[1] https://arc.liv.ac.uk/trac/SGE

sity of Innsbruck[2].
However, the READ partners have agreed upon a new paradigm regarding tool integration (see D4.2) and thus a new implementation by UPVLC will be provided in 2017 and the integration process has to be reevaluated once the new version becomes available.

3.2 URO RNN HTR

The adaptation of the HTR engine by URO, based on Recurrent Neural Networks (RNN), is still ongoing. The implementation is based on Open MPI[3] and, as the engine is programmed in Java, the respecive Java Native Interface (JNI) is employed in order to share HPC resources.
Due to two bugs in the experimental JNI facilities of Open MPI, there is not yet a running version. The bugs have been confirmed on the HPC cluster of the University of Rostock and could be reproduced on the cluster of the University of Innsbruck. While the bugfix for this is still pending, efforts have been halted on this task.

4 Process Management

For integrating HPC infrastructure with Transkribus, four things have to be taken into account:

- Transkribus uses network attached storage (NAS) devices for exchanging files between its subsystems and access to this storage is mandatory
- All subsystems are written in Java 8, thus this software must be available
- Database access is needed for exchanging persisted object data
- Workflow implementations must be ported to the specific HPC system

While the first three requirements could be negotiated with the computing centre of the University of Innsbruck to be fulfilled on the HPC cluster, the last requirement is not trivial to be accomplished. The Son of Grid Engine, mentioned in section 3.1, contains an own scheduler and jobs are submitted via special start scripts. Interfacing with this engine with Java is cumbersome and other options, such as Open MPI, have to be investigated thoroughly before going in that direction. On the other hand, if a tool itself utilizes HPC facilities, the task can be accomplished rather easily. However, the only tools fitting this requirement (see section 3.2 and 3.1) are not yet available on this environment.
Due to those circumstances, UIBK has decided to shift this task to 2017.

[2]https://www.uibk.ac.at/zid/systeme/hpc-systeme/leo3/
[3]https://www.open-mpi.org/

5 Conclusion and Outlook

High performance computing in the field of handwritten text recognition is undoubtedly an important topic. Especially the task of training the engines on new script types already became a crucial bottleneck although the feature is not yet enabled for all users. Due to new server machines, which UIBK acquired (see D4.1), the criticality of this could be mitigated. However, in 2017 more resources will be put in this task, particularly when the natively HPC-compliant tools become available.