PortageII 2.0
Performance Information and Required Configuration

Updated Feb 2013

Caveat: this document is based on benchmarks run in September 2011 using Portage 1.4.3. The system requirements of PortageII are the same as those of Portage 1.4.3 when using the configuration described here. When using the configuration we recommend for PortageII, you can expect training and translation times to be 20 to 50% longer, while memory requirements remain comparable. Either way, the recommended hardware configuration is the same.

This document contains information about the performance of PortageII and the required IT configuration to make it work. The measurements provided here are approximate as a number of factors can have an impact on PortageII performance in any given situation; however, this will be sufficient to establish recommendations as to the configuration necessary for a system that will run PortageII.

We have measured performance in three scenarios:
1) a corpus comprised of about half a million aligned sentence pairs from the Canadian Hansard, with over 10 million tokens in each language – we will call this the "small" corpus;
2) a corpus comprised of about 1.7 million aligned sentence pairs for the European Parliament proceedings (europarl corpus), with around 50 million tokens in each language – we will call this the "medium" corpus;
3) a corpus comprised of 5.2 million aligned sentence pairs from the committee proceedings of the Canadian parliament, with around 90 million tokens in each language – we will call this the "large" corpus.

PortageII is trained using the framework provided with Portage 1.4.3, going from English to French, using MITLM to build language models, performing rescoring, confidence estimation and truecasing, but not using lexicalized distortion models (LDMs). (Note: with Portage 1.4.3, using LDMs was not recommended, but with PortageII, using their hierarchical variant, HLDMs, is recommended.)

Training PortageII involves the following steps:
1) LM – training of language models / NOTE: this is done with software not provided by NRC;
2) TM – training of translation models;
3) TC – training of truecasing models;
4) TUNE – optimizing the decoder weight;
5) CE – training the confidence estimation model (optional).
6) RESCORE – optimizing the weights for rescoring (optional, often not worthwhile);

	Small corpus			Medium corpus			Large corpus		
Step	**Wall time**	**CPU time**	**Peak RAM**	**Wall time**	**CPU time**	**Peak RAM**	**Wall time**	**CPU time**	**Peak RAM**
LM	0.1 h	0.1 h	1 GB	0.6 h	0.6 h	4 GB	0.9 h	1.0 h	6 GB
TM	1.1 h	12.5 h	3 GB	5.0 h	76.9 h	15 GB	7.9 h	87.3 h	18 GB
TC	0.4 h	0.4 h	1 GB	1.9 h	1.9 h	4 GB	3.0 h	3.1 h	6 GB
TUNE	0.8 h	3.3 h	1 GB	2.4 h	20.5 h	4 GB	1.2 h	8.1 h	3 GB
CE	0.1 h	1.4 h	1 GB	0.5 h	11.1 h	2 GB	0.4 h	11.3 h	3 GB
RESCORE	0.5 h	4.5 h	2 GB	0.5 h	5.8 h	2 GB	0.3 h	2.2 h	2 GB

Table 1: Time and memory requirements for training 16-ways parallel; Wall time = elapsed time; CPU time = total CPU time over all parallel processes; Peak RAM = the most RAM needed by any one process in each step

Translation can be done using decoding only, decoding plus rescoring, or decoding plus confidence estimation.

	Small corpus		Medium corpus		Large corpus	
	Speed	**RAM**	**Speed**	**RAM**	**Speed**	**RAM**
Decoding only	0.1 s / sent	1 GB	1.5 s / sent	3 GB	0.3 s / sent	3 GB
With CE	0.1 s / sent	1 GB	1.5 s / sent	3 GB	0.3 s / sent	3 GB
With rescoring	2.7 s / sent	1 GB	6.8 s / sent	3 GB	2.3 s / sent	3 GB

Table 2: Speed and memory use for translating

You'll notice in Table 2 that the medium corpus is the slowest to translate. Also, in Tables 2 and 3, one sees that rescoring is faster for the large corpus than for the small one. You should expect this kind of variability, which can be caused by many factors, such as characteristics of the training corpus, test data, etc. Thus, our benchmarks can only give you a rough idea of the performance you might expect from PortageII.

	Small corpus	Medium corpus	Large corpus
Training	3.8 GB	15.8 GB	26.7 GB
Final models	1.3 GB	5.4 GB	8.8 GB

Table 3: Disk usage

Experiments carried out on these corpora show that rescoring has a high cost while adding little to the results; therefore, our recommendation is not use rescoring. Confidence estimation can be used to give an interactive user an idea of the expected quality of output, and costs little extra. The decoding only and decoding plus CE systems are thus the relevant references, even though we have also measured and analyzed using rescoring for the sake of completeness of our analysis.

We performed these experiments on a Dell R710 machine with Intel Xeon X5560 CPUs, with a total of 8 cores and 48GB of RAM, with hyperthreading enabled, giving us 16 virtual CPUs. All parallelizable steps in PortageII were run 16-ways parallel. Such a large machine is not required, but it is worth having. It cost around $10K in 2010. The CPU time columns in Table 1 give reasonable estimates of how long things would take if no parallelization was used.

Recommended System Configuration

For training PortageII 2.0

RAM: a strict minimum of 16 GB is required, 32 to 64 GB is recommended, and any quantity up to or even over 128GB continues to be useful. For multi-core machines, you should have at least 4GB of RAM per core (or virtual core, if you use hyperthreading), so you can maximize parallelism.
Number of CPUs and cores: it's worth having many cores (4 to 16), although there is no minimum requirement. PortageII can take advantage of all available cores, and can also take advantage of clustered environments with multiple machines sharing the same file system. We use a cluster of machines with 4 to 16 cores each.
Disk space: this depends on how many systems you plan to train, but you should plan no less than 1 TB, while several to many TBs will likely be useful. This storage does not need to be local; it can be on an NFS server or on a SAN.
Swap space: in your disk purchases, allow for twice as much local disk space for swap as you have RAM on each machine. This is on top of the disk space you need for your work (discussed above), and on top of other local disk space required for the OS and /tmp. The 2:1 swap:RAM ratio is important to respect : having less swap space than that will cause problems.
OS: PortageII only runs on Linux 64 bits. We have used Scientific Linux, Red Hat, OpenSUSE, Ubuntu; PortageII is compatible with most Linux distros.
Virtualization: not recommended for training PortageII.

For translating using PortageII 2.0

RAM: you should plan for enough RAM to hold all the models in memory, plus a bit more, even though comparing Tables 2 and 3 indicate that's not strictly necessary. If you plan to allow parallel access to your translation server, allow some more RAM, but not linearly: concurrent requests that use the same models will share the memory required for these models, so only the memory needed for the computations increases; however, concurrent requests using different models means adding the memory requirements for both models and computation space.
Number of CPUs and cores: again, there is neither minimum nor maximum here, PortageII can exploit what you have.
Disk space: here the requirements are quite small, in the 10's of GBs, or more if you plan to have many systems on the same server. Periodic clean up of the translation server is recommended, however, as temporary files are not automatically removed.
Swap space: follow the 2:1 swap:RAM ratio, as discussed above.
OS: Linux 64 bits (see above for details).
Virtualization: PortageLive allows for easily deploying and running translation servers running Linux in a virtual computing environment (e.g., using VMWare on a PC or a centralized server), although a multi-core physical server will allow you to better exploit the benefits of memory-mapped IO and obtain faster performance.