## Application form: Compute component of the Flemish Tier-1 supercomputing platform

|  |
| --- |
| Title of the application:  EasyChair code in case of resubmission: EasyChair code in case of continuation:  Applicant name, first name: Institution: Research group / department: E-mail address: OECD FoS code (see regulations):  VSC id of all mandated persons, separated by commas:  Core-hours applied for: GPU-hours applied for: Largest amount of scratch disk required (in TiB) on Tier-1 at any given time (so *not* the grand total amount):  Largest associated number of files on Tier-1 at any given time:  List of simulation codes and their version numbers:  |

*This application should not exceed 18 pages, excluding possible appendices (confirmation letter of financing institution, software license, etc.) which may be considered by the Tier-1 Allocation Board.*

1. Research project within the framework of which computing time is applied for.
	* Title
	* Supervisor(s) and their e-mail address
	* If available, IWETO or FRIS link
	* Financing institution or channel (FWO, BOF, VLAIO, EU, etc.).

Attach the confirmation letter as enclosure. Attach a letter of approval of your own institution in case the project has not gone through a scientific approval process.

1. Include a short description of your research project, in layman’s terms wherever possible, with a view to dissemination. Explicitly mention the scientific questions that you are planning to address and the overall scientific goals of the project. (max. 1 A4 in Arial 12)

1. Persons mandated by the Applicant to compute on the Tier-1 within the framework of the present project. Please provide for every person:
	* Name, first name
	* VSC id
	* Institution
	* Research group / department:
	* Experience with using particular HPC resources (i.e. Tier-0/Tier-1/Tier-2 infrastructure) in Belgium and abroad. Specify both the name of infrastructure and number of years it was used.
	* List of computing time allocations received during the past two years, on the Flemish Tier-1 systems, as well as other Tier-1 and Tier-0 systems.

1. Why does this project need to run on a Tier-1 system? Select appropriate checkbox(es):

[ ] It requires resources (#nodes, #cores, #GPUs, memory, interconnect,
 storage) that are not available on Tier-2 systems.

[ ]  The total runtime for the project (e.g. due to number of jobs, job turnaround
 time) would be prohibitively large on Tier-2 systems.

[ ]  Other reason, please specify.

1. Provide information for each software package that will be used.
	* If centrally installed on Tier-1 compute or a Tier-2 system within VSC, state the module name and system name.
	* If not open source software, state that the associated license can be validly used by all mandated users on Hortense. Add a copy of the signed license to this application.
2. Provide the results of parallel efficiency tests for each software package that will be used.
	* Perform these benchmark tests on Hortense (using, e.g., a Starting Grant).
	* Use system/problem sizes that closely reflect those of the intended computational tasks (e.g., same mesh size, actual molecular system, similar I/O pattern, same communications patterns, etc.). If a different system/problem size is used in the tests, describe how it relates to the problem size in the application. Characteristic I/O must be included in the tests. For example, simply run your application tasks for a limited number of iterations.
	* List the results in a table and plot efficiency versus number of cores or number of GPUs using a log scale x-axis (see example Table 1 and Plot 1).
	* Start the scaling tests of your code using the *smallest* number of cores or GPUs possible. If possible, the baseline is using 1 core or 1 GPU on a dedicated node. If not possible, explicitly state why (e.g. lack of memory, impossible to finish within wall clock time of 72 hours, …).
	* Mention on which partition the tests were run: cpu\_rome, cpu\_milan, gpu\_rome\_a100\_40 or gpu\_rome\_a100\_80.
	* Wall clock times are preferably obtained by averaging the timing results of several similar simulations for each node/core/GPU configuration. This is required when task farming jobs that vary significantly in run time. In that case, give an indication of variation (e.g. standard deviation).
	* When benchmarking on GPUs, the timing for a run on one full CPU node should also be reported (when a CPU version of the code is available) to assess the speedup obtained by computing on GPUs. Report the timings when using the CPU node exclusively using only the optimal number of cores.
	* Task loads that don’t use the maximum number of cores/GPUs per node are preferentially packed together, using the worker framework, atools, …. If the maximum number of cores cannot be used because of bandwidth issues, this should be mentioned explicitly.
	* Explain anomalies in plot and table.
	* Clarify, based on the parallel efficiency plot and table, which number of nodes and cores/GPUs you plan to use for your computational tasks (cf. Section 7) and explain why, since production hours must be explicitly derived from scaling results. *Parallel efficiency should be at least 50% in competitive calls.*
	* In case of VASP, explicitly state the values of NPAR, KPAR, NCORE, NBANDS investigated to determine the optimal combination and the combination(s) of those parameters chosen for your computations.
	* In case of computing different systems (be it in chemical composition or size), provide benchmarks for all relevant system sizes, e.g. a benchmark for small, medium and large size and/or for the different compositions.
	* The default memory per core is 1970 MB on the 256 GB Rome and Milan nodes, and 3970 MB on the 512 GB Rome nodes. If you use more memory per core than the default, this should be taken into account in computing the “Total core-hours per task”. The factor is (estimate-of-memory-usage-per-core / default-memory-per-core). Examples:
		+ A job requesting 32 cores and 32 GB of memory, requires 1 GB per core, so less than the default. Factor = 1.
		+ A job requesting just 1 core but all the memory of a 256 GB Rome node, effectively uses all cores of that node. Factor = 128.
		+ A job requesting 32 cores and 96 GB of memory, requires 3 GB per core. This is more than the default of 1970 MB on the 256 GB nodes but fits within the default of 3970 MB on the 512 GB nodes of cpu\_rome\_512, so factor = 1. If the jobs needs to run on the cpu\_milan partition, the factor becomes 3072 / 1970 = 1.64.
		+ A job requesting 32 cores and 160 GB of memory, requires 5 GB per core, so more than the default of 3970 MB on the 512 GB nodes. Factor = 5120 / 3970 = 1.29. If the job would run on the 256 GB nodes, the factor becomes 5120 / 1970 = 2.65, so you’re advised to choose the most optimal partition for your job.

If you don’t specify memory requirements in your job script, the factor will be 1.

Example Table 1 (CPU)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of nodes  | Total number of cores  | Wall clock time (s)  | Speed-up (w.r.t. baseline)  | Efficiency  |
| *Abaseline*  | *Bbaseline*  | *Cbaseline*  | *1.00*  | *1.00*  |
| *A1*  | *B1*  | *C1*  | *Cbaseline/C1*  | *(Bbaseline\*Cbaseline)/(B1\*C1)*  |
| *A2*  | *B2*  | *C2*  | *Cbaseline/C2*  | *(Bbaseline\*Cbaseline)/(B2\*C2)*  |
| *Baseline = minimal configuration with which your computational task can be carried out on Tier-1.*  |
| *Wall clock time is difference between start/end of the computational task, including any I/O operations as part of that task.*  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of nodes  | Total number of cores  | Wall clock time (s)  | Speed-up (w.r.t. baseline)  | Efficiency  |
| 1 | 1 | 200000 | 1,00 | 1,00 |
| 1 | 32 | 6300 | 31,75 | 0,99 |
| 1 | 64 | 3161 | 63,27 | 0,99 |
| 1 | 128 | 1597 | 125,24 | 0,98 |
| 2 | 256 | 850 | 235,29 | 0,92 |
| 4 | 512 | 460 | 434,78 | 0,85 |
| 8 | 1024 | 250 | 800,00 | 0,78 |
| 12 | 1536 | 180 | 1111,11 | 0,72 |
| 16 | 2048 | 150 | 1333,33 | 0,65 |
| 32 | 4096 | 90 | 2222,22 | 0,54 |
| 64 | 8192 | 55 | 3636,36 | 0,44 |

Example Plot 1 (CPU)

**

*The optimal number of cores in this example is 1536, as parallel efficiency quickly drops below 70% when more cores are used.*

Example Table (GPU)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Number of nodes  | Total number of CPUs  | Total number of GPUs  | Wall clock time (s)  | Speed-up (w.r.t. baseline)  | Efficiency  |
| 1 | 12 | 1 | 4153 | 1,00 | 1,00 |
| 1 | 24 | 2 | 2530 | 1,64 | 0,82 |
| 1 | 48 | 4 | 1256 | 3,31 | 0,83 |
| 2 | 24 | 2 | 2510 | 1,65 | 0,83 |
| 2 | 48 | 4 | 1140 | 3,64 | 0,91 |
| 2 | 96 | 8 | 745 | 5,57 | 0,69 |

If available: timing on one full CPU node (using the most optimal number of cores) of the CPU version of the code used on the GPUs.

|  |  |  |
| --- | --- | --- |
| Number of nodes  | Total number of cores  | Wall clock time (s) |
| 1 | 128 | 4000 |

1. Justify the number of core-hours and GPU-hours, and storage volume applied for.

Describe your planned computational tasks and the sequence in which these tasks will be performed. Start from the examples in Table 2 and Table 3 and adjust them to your project.

Note that per requested GPU-hour on Hortense, you will automatically receive 12 core-hours on the CPU cores of the node containing that GPU unit. These core-hours do not need to be specified explicitly on page 1 and in Table 3.

Provide additional descriptions for the computational tasks listed in the table. Resource estimates (wall clock time, number of nodes/cores/GPUs, estimate of memory requirement (not the target node memory), storage) should be based on the results of actual calculations on Hortense (via, e.g., a Starting Grant) for system/problem sizes that match closely those of the intended computing tasks (e.g., same mesh sizes, actual molecular system, same I/O pattern, same amount of communications, etc.). If you plan to run the tasks concurrently, mention this in the description, so you can specify the correct total amount of scratch space required at any given time.

*NB: After 3 months of the allocation time have passed, you will lose 20% of the initially granted core-hours and GPU-hours, if that 20% has not been used. (cf.*

*regulations, § 9)*

Example Table 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|   | Core-hour calculation   |  |  |  |  |   |   | Storage volume estimate  |
| Computational task  | Number of such jobs  | Wall clock time (in hours) per job  | Number of Tier-1 nodes per job  | Number of Tier-1 cores per node per job  | Memory factor (memory-per-core (MB) / default-memory-per-core\*) | Total core-hours per task\*  | Estimate of memory usage (GiB) per node per job  | OpenMP / MPI / OpenMP + MPI (hybrid) / worker framework / atools / etc.  | Tier-2 DATA/HOME volume (TiB) + number of files  | Tier-1 SCRATCH volume (TiB) number of files  | +  |
| Task * software X
* parameters/conditions
* system/mesh size
* …
 | A  | B  | C  | D  | E | = A x B x C x ceil(D x E)  | F  |   |   |   |  |
| Task example CP2K * CP2K – MD
* 100 ns runs
* PBE functional
* 1 -> 5 water molecules
 | 5  | 48  | 12  | 128  | 1 | 368640  | 64  | MPI  | 0 TiB 0 files  | 0.1 TiB 5000 files  |  |
| Task example worker * MDTraj postprocessing
* 5000 files
 | 10000  | 0.5  | 1  | 1  | 1 | 5000  | 1.5(192 GiB for 128 jobs in one node)  | These single-core jobs will be packed within 1 node using worker framework  | 1 TiB 10000 files  | 0.1 TiB 5000 files  |
|   |   |   |   |   |  | Sum of core-hours applied for = …  |   |   |   | Largest amount of scratch disk required + number of associated files *at any given time*= …  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Important information:*  |   | *3 days is the* *maximum* *wall clock time for any job.*  |   |   |   | *Memory limits (GiB/CPU node) Hortense:* *256/512*   |   |   |   |

*Regarding the storage volume estimate, we do not ask the total number of files or volume generated during the complete project, but the maximum number at any given time. This will be (a lot) smaller than the total number or volume.*

*\* Default memory per core: 1970 MB on the 256 GB Rome or Milan nodes, 3970 MB on the 512 GB Rome nodes*

The example CP2K task needs to run 5 times, for a molecular system containing 1 to 5 water molecules. Based on timing runs on Hortense, we found that one such job runs for 48 hours on 12 nodes, using all the cores (128) in the node. The job needs 64 GiB RAM in each node and produces 20 GiB of SCRATCH storage (1000 files). Since the 5 jobs (for the 5 listed molecular systems) will be run concurrently, 5 x 20 GiB = 100 GiB of scratch disk space is required (and 5 x 1000 = 5000 files) for the entire task.

File postprocessing with the MDTraj tool is done in the example worker task, where 10000 jobs need to run on 5000 files on the SCRATCH volume to generate 10000 files on the Tier-2 DATA volume. Each job runs on 1 core. Based on 5 timing runs on Hortense, we found that the job duration varies between 25 and 28 minutes, and memory usage is 1.5 GiB at most. To be on the safe side, we foresee 30 minutes per job (0.5 hours). Wherever possible, 128 jobs will be packed on a single node using the worker framework, so 128 jobs require 1 full node for 30 minutes. For the entire task, 5000 core-hours are required.

Example Table 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|   | GPU-hour calculation   |  |  |  |   |   | Storage volume estimate  |
| Computational task  | Number of such jobs  | Wall clock time (in hours) per job  | Number of Tier-1 nodes per job  | Number of Tier-1 GPUs per nodeper job  | Total GPU-hours per task  | Estimate of memory requirement (GiB) per node per job  | OpenMP / MPI / OpenMP + MPI (hybrid) / worker framework / atools / etc.  | Tier-2 DATA/HOME volume (TiB) + number of files  | Tier-1 SCRATCH volume (TiB) + number of files  |
| Task * software X
* parameters/conditions
* system/mesh size
* …
 | A  | B  | C  | D  | = A x B x C x D  |   |   |   |   |
| Task example QE * Quantum Espresso
* 1,500 compounds
* SCF calculation
 | 1500  | 8  | 1  | 2  | 24000  | 106  | MPI & OpenMP  | 0.4 TiB 2500 files  | 1.2 TiB 7500 files  |
|   |   |   |   |   | Sum of GPU-hours applied for = …  |   |   |   | Largest amount of scratch disk required + number of associated files *at any given time*= …  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Important information:*  |   | *3 days is the* *maximum* *wall clock time for any job.*  |   |   | *Per requested GPU-hour, you automatically receive 12 core-hours on the CPU cores of the GPU node. Please do not specify these core-hours in the table.* | *Memory limits (GiB/GPU node) Hortense:* *256*   |   |   |   |

The example QuantumEspresso task needs to run 1500 times, to perform an SCF calculation on 1500 different compounds. All tasks can be executed independently of each other. Based on timing runs on the GPU nodes of Hortense, we found that one such job runs for 8 hours on 1 node, using 2 GPUs along with 24 CPU cores in the GPU node. Each job requires 106 GiB of RAM, therefore two jobs can run simultaneously on a Hortense GPU node (256 GiB). The worker framework will be used to pack 2 tasks in one job that will make sure both end up on one GPU node, optimally using all GPUs of that node. Each job generates 5 files that total 0.8 GiB. For all tasks, this amounts to 1500 x 0.8 GiB = 1.2 TiB of SCRATCH storage (7500 files). These will be regularly offloaded to the Tier-2 DATA storage in a compressed format.

1. Describe how you will manage the workflow and the resources requested in the period during which the task is to be performed.

In case you will launch a large number of computational tasks, describe how you will manage your jobs and provide details regarding job management, automation and dataflow. Just submitting “manually” is not advised. Will you make use of a task/workflow manager, such as the worker framework, atools or something similar? On which infrastructure or node will this manager run? The VSC Cloud can help if you want to run dedicated infrastructure for your workflow ([https://www.vscentrum.be/cloud).](https://www.vscentrum.be/cloud)

Please present how you will manage your data. Since there is no backup of the scratch file system, this is important. Describe how the transfer of files to/from Hortense will be managed and automated, and whether data reduction and/or compression of files will be performed. If available, provide information about IOPS.

1. In case you requested GPU compute time, are you interested in getting a preparatory access project on LUMI), a European pre-exascale computer (<https://lumi-supercomputer.eu/>, taking into account the “detailed instructions for application” on <https://www.enccb.be/GettingAccess>? No separate application would be required. If you answer “Yes” below, the LUMI team will contact you with further information.

[ ]  Yes

[ ]  No

*Applicants allow FWO/VSC to make this proposal in its entirety public e.g. as an example or inspiration for other researchers.*

*Applicants commit to collaborate with VSC, upon its request, in the preparation of a success story (see* [*https://www.vscentrum.be/stories*](https://www.vscentrum.be/stories)*).*

|  |
| --- |
| Don’t hesitate to consult the Tier-1 Compute support (compute@vscentrum.be) or your local support ([www.vscentrum.be/getintouch](http://www.vscentrum.be/getintouch)) when you are preparing your application. |