Main activities

- Study state of the art on data replication and load balancing
- Analyze needs for target applications of exa2pro
- Propose extensions to existing replication and load balancing strategies to leverage task parallelism
- Improving dynamic distributed load-balancing strategies to cope with the loss of computation nodes, while minimizing the amount of replications by determining pieces of data that can be recovered from data that must be preserved
- Conceiving StarPU runtime system replication strategies so that the loss of a computation node does not entail loss of data and the computation can be efficiently resumed without impacting all other computation nodes, while minimizing the amount of replications by determining pieces of data that can be recomputed from data that must be preserved
- Extending the task-based programming model and its execution model to allow for failure detection, and for seamless restart strategies, so that the runtime uses the task graph information to recover the data required by remaining tasks from the data which was duplicated, and restart the tasks whose instantiation was lost
- Improving dynamic distributed load-balancing strategies to cope with the loss of computation nodes, while limit the volume of data redistribution

Going further, by exploring possible relationships and cooperation between scheduling algorithms and the replication/restart mechanisms to increase the effectiveness of replication, while limiting the impact of restart steps.

Assignment

The Exa2pro European project gathers different research institutes and industrial partners to aim at enhancing programmability of the future exascale computing systems.

At these scales, hardware failures are very common, leading to the sporadic loss of whole computation nodes, and to the need of fault tolerance techniques. However, at these scales, the classical fault tolerance techniques such as checkpoint-restarts even show their limitations themselves. Task-based programming models however provide rich information about the flow of computations and their relationship with respect to data, which can be used to revamp these techniques with fine-grain, selective replications and restarts according to the task graph. Moreover, this should allow to let execution continue while the lost tasks and data are being recovered in an optimized way from data checkpoints.

The unexpected loss of a computation node entails a disruption in the balancing of the computation load, due to the loss of the node itself, but also due to the computation time lost in recovering from the data loss. This thus imposes careful dynamic redistribution of the workload of the failed node and recovery expenses over the system to avoid imbalances while limiting data redistributions.

The Inria STORM team has a long experience on runtime systems in general, and on task-based runtime systems in particular, notably through research related to the StarPU runtime system. StarPU currently supports scalable distributed execution through the MPI communication standard, and work is currently conducted to add adaptive dynamic load balancing according to application load fluctuations.

The keys to success

Affinities both with abstracting problems and actually implementing the proposed solutions.

The unexpected loss of a computation node entails a disruption in the balancing of the computation load, due to the loss of the node itself, but also due to the computation time lost in recovering from the data loss. This thus imposes careful dynamic redistribution of the workload of the failed node and recovery expenses over the system to avoid imbalances while limiting data redistributions.

The unexpected loss of a computation node entails a disruption in the balancing of the computation load, due to the loss of the node itself, but also due to the computation time lost in recovering from the data loss. This thus imposes careful dynamic redistribution of the workload of the failed node and recovery expenses over the system to avoid imbalances while limiting data redistributions.

The unexpected loss of a computation node entails a disruption in the balancing of the computation load, due to the loss of the node itself, but also due to the computation time lost in recovering from the data loss. This thus imposes careful dynamic redistribution of the workload of the failed node and recovery expenses over the system to avoid imbalances while limiting data redistributions.

Warning: you must enter your e-mail address in order to save your application to Inria. Applications must be submitted online on the Inria website. Processing of applications sent from other channels is not guaranteed.
graphs

Complementary activities:

- Write documentation for the proposed application interfaces
- Write research papers
- Present scientific results in conferences

Skills

Technical skills:

- Mastering software development under UNIX-like operating systems
- Good level in C/C++ language programming, system programming and parallel programming

Language:

- Mastering technical and scientific English
- Good writing skills

Additional skill:

- Knowledge of MPI, task-based programming

Benefits package

- Subsidised catering service
- Partially-reimbursed public transport

Remuneration

1982€ / month (before taxes) during the first 2 years, 2085€ / month (before taxes) during the third year.