Proposition de recherche doctorale

Memory management optimisation in distributed systems

Mots clés :

- **Directeur de thèse** : Eric RENAUDT
- **Co-encadrant(s)** :
- **Unité de recherche** : Services répartis Architectures MOdélisation Validation Administration des Réseaux
- **Ecole doctorale** : École Doctorale Informatique, Télécommunications, Électronique de Paris
- **Domaine scientifique principal** : Divers

Résumé du projet de recherche (Langue 1)

In order to explore further the capabilities of parallel computing architectures such as grid, cluster, multi-processors and more recent, cloud and multi-cores, an easy-to-use parallel language is an important challenging issue. From the programmer's point of view, OpenMP is very easy to use with its ability to support incremental parallelization, features for dynamically setting the numbers of threads and scheduling strategies. However, as initially designed for shared memory systems, OpenMP is usually limited to intranodes' computations on distributed memory systems. Many attempts have tried to port OpenMP on distributed systems. The most emerged approaches focus on exploitation capacities of a special network architecture and so they can not provide an open solution. Some others use another support technique such as DMS, MPI, Global Array and as a consequence, they meet the difficulties to become a fully-compliant and high performance implementation of OpenMP. As yet another attempt to built an OpenMP compliant implementation for distributed memory systems, CAPE --that stands for Checkpointing Aide Parallel Execution-- was built which main idea is as follows: when reaching a parallel section, the master thread is dumped and its image is sent to slaves; then, each slave executes a different thread; at the end of the parallel section, slave threads extract and return to the master thread a list of modifications that has been locally performed; the master includes these modifications and resumes its execution. For proving the feasibility of this paradigm, the first version of CAPE was implemented using complete checkpoints. However, some analysis showed that the large amount of data transferred between threads and the extraction of the list of modifications from complete checkpoints lead to a weak performance. Furthermore, this version was restricted in parallel problems verifying Bernstein's conditions, i.e, it did not solve the requirements of shared data. This thesis present the approaches to improve CAPE's performance and to overcome the restricts on the shared data. First, we have developed the DICKPT (which stands for Discontinuous Incremental Checkpointing) technique, an incremental checkpointing technique that supports the ability to save incremental checkpoints discontinuously during the execution of a process. Based on the DICKPT, the execution speed of the new version of CAPE has significantly increased. For example, the time to compute a large matrix-matrix product on a desktop cluster is very similar to the execution time of the same optimized MPI program. Moreover, the speedup associated with this new version for various number of threads is quite linear for different problem sizes. In the side of shared data, we propose the UHLRC, that stands for Updated Home-based Lazy Release Consistency, a modified version of the Home-based Lazy Release Consistency (HLRC) memory model, in order to make it more appropriate with the characteristic of CAPE. The prototypes and algorithms to implement the synchronization and the OpenMP data-sharing clauses and directives were also specified. These two works have ensured the ability of CAPE in aspect of shared-data.