

Towards efficient and scalable parallel static mapping

Jun-Ho Her

INRIA Bordeaux Sud-Ouest

François Pellegrini

ENSEIRB, LaBRI and INRIA Bordeaux Sud-Ouest

351, cours de la Libération, 33405 TALENCE, FRANCE

{jun-ho.her|francois.pellegrini}@inria.fr

I. CONTEXT OF OUR WORK

Graph partitioning is an ubiquitous technique which has applications in many fields of computer science and engineering. It is mostly used to help solving domain-dependent optimization problems modeled in terms of weighted or unweighted graphs, where finding good solutions amounts to computing, eventually recursively in a divide-and-conquer framework, small vertex or edge cuts that balance evenly the weights of the graph parts.

Because there always exists large problem graphs which cannot fit in the memory of sequential computers and cost too much to partition, parallel graph partitioning tools have been developed. PT-SCOTCH is another attempt to provide a simple and efficient library for parallel graph partitioning and ordering.

Previous work from our team resulted in the design of several efficient parallel algorithms for the parallel ordering of distributed graphs by nested dissection [1]. These building blocks comprise a parallel multi-level framework taking advantage both of a probabilistic matching algorithm [2], [3] as well as of a banded diffusive refinement algorithm [4] to smooth the computed partitions during its uncoarsening phase. Then, the recursive vertex separation process used to compute nested dissection orderings has been adapted to edge bipartitioning, enabling the parallel computation of k -way graph partitions by means of recursive weighted bipartitioning [5].

II. LIMITATIONS OF RECURSIVE BIPARTITIONING

The parallel computation of graph partitions by means of recursive bipartitioning has been made available since release 5.1 of the PT-SCOTCH software package [6]. Its current revision, 5.1.6, provides partitions of good quality irrespective of the number of

processes, while competing software PARMETIS [7] can suffer a loss in partition quality (increased cut size) when the number of processes increases (see Fig. 1).

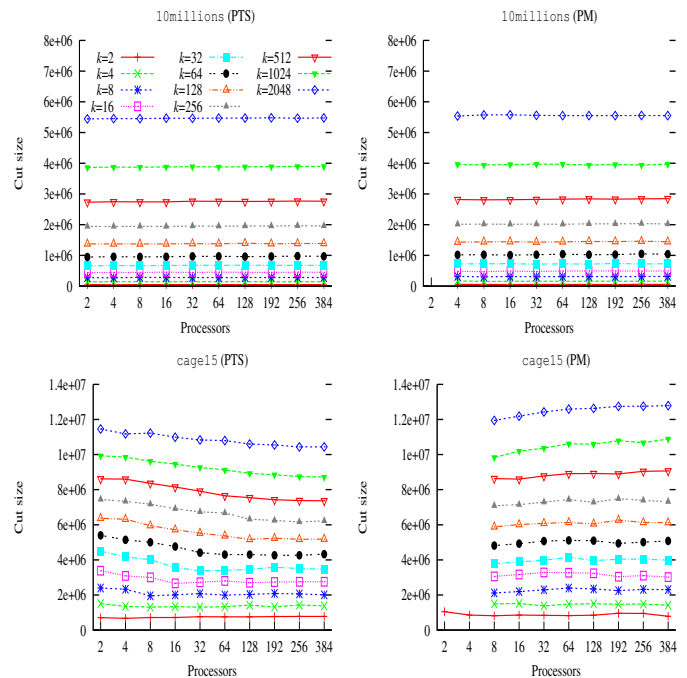


Fig. 1. Cut sizes of PT-SCOTCH (PTS, left) and PARMETIS (PM, right) for graphs 10MILLIONS and CAGE15.

On most of our experiments, the partitions computed by PT-SCOTCH compare favorably to the ones produced by PARMETIS. For instance, this gain can be as high as 20 % when bipartitioning, on 384 processors, a mesh graph of more than 82 million vertices.

However, our current recursive bipartitioning scheme has some major drawbacks. While PT-SCOTCH can be more than three times faster than PARMETIS in the bipartitioning case, such as for

the aforementioned mesh graph, when the number of parts increases, its execution time suffers a penalty factor which tends to a constant proportional to the inverse of the coarsening ratio [5], as can be seen in Fig. 2. Also, its scalability in time, which is excellent up to 128 processes, suffers from the amount of data to exchange so as to fold the separated subgraphs onto half of the processes at each stage of the recursive bipartitioning process.

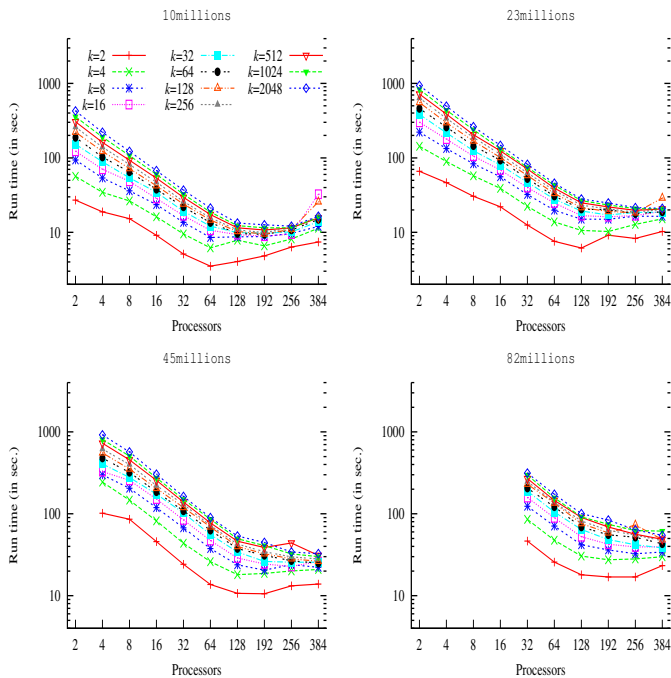


Fig. 2. Execution times of PT-SCOTCH for graphs 10MILLIONS, 23MILLIONS, 45MILLIONS and 82MILLIONS.

Finally, recursive bipartitioning may eventually result in poorer partition quality, due to the greedy nature of the algorithm, compared to direct k -way partitioning [8]. In [5], we evidenced that, for the test graphs we considered, the overhead of recursive bipartitioning starts to overwhelm the gain of our refinement algorithm when considering more than a thousand of parts.

III. PARALLEL DIRECT k -WAY STATIC MAPPING

For all of the aforementioned reasons, we are currently developing a parallel direct k -way graph partitioning method, basing on our parallel multi-level framework, by extending our diffusion refinement method to the k -way case. This extension has already been done in the sequential case by [9], basing on our findings, and its parallelization should be as straightforward as it has been for the 2-way case. This should give us the best of both worlds, in term of speed as

well as in term of quality.

Moreover, as recent parallel computing architectures are characterized by ever increasing numbers of processors and heavily heterogeneous communication subsystems, taking into account the underlying topology of the target machine is essential to effective minimization of running time. Static mapping is the corresponding combinatorial problem, which aims at assigning statically parallel tasks onto physical processors so as to reduce some relevant message congestion cost function. Preliminary results, using an iterative sparse linear system solver as test program, showed that significant gains in runtime can be achieved when using static mapping, instead of plain graph partitioning, to compute domain decompositions for multi-core architectures.

Our talk will describe the new parallel graph partitioning and static mapping features of PT-SCOTCH, and present some experimental results.

REFERENCES

- [1] J. A. George and J. W.-H. Liu, *Computer solution of large sparse positive definite systems*, Prentice Hall, 1981.
- [2] C. Chevalier, *Conception et mise en œuvre d'outils efficaces pour le partitionnement et la distribution parallèles de problèmes numériques de très grande taille*, Thèse de Doctorat, LaBRI, Université Bordeaux I, Sept. 2007.
- [3] C. Chevalier and F. Pellegrini, "PT-SCOTCH: A tool for efficient parallel graph ordering," *Parallel Computing*, vol. 34, pp. 318–331, 2008.
- [4] F. Pellegrini, "A parallelisable multi-level banded diffusion scheme for computing balanced partitions with smooth boundaries," in *Proc. Euro-Par'07, Rennes*. Aug. 2007, vol. 4641 of *LNCIS*, pp. 191–200, Springer.
- [5] J.-H. Her and F. Pellegrini, "Efficient and scalable parallel graph partitioning," *Parallel Computing*, 2009, Submitted.
- [6] "SCOTCH: Static mapping, graph partitioning, and sparse matrix block ordering package," <http://www.labri.fr/~pelegrin/scotch/>.
- [7] "METIS: Family of multilevel partitioning algorithms," <http://glaros.dtc.umn.edu/gkhome/views/metis>.
- [8] H. D. Simon and S.-H. Teng, "How good is recursive bipartitioning," *SIAM J. Scientific Computing*, vol. 18, no. 5, pp. 1436–1445, sep 1997.
- [9] H. Meyerhenke, B. Monien, and T. Sauerwald, "A new diffusion-based multilevel algorithm for computing graph partitions of very high quality," in *Proc. 22nd IPDPS*, 2008.