From Days to Seconds: A Scalable Parallel Algorithm for Motion Planning*

Sam Ade Jacobs and Nancy M. Amato
Parasol Lab., Dept. of Computer Science and Engineering, Texas A&M Univ
College Station, Texas, 77843-3112, USA
(sjacobs,amato)@cse.tamu.edu

ABSTRACT

This work presents a scalable framework for parallelizing sampling based motion planning algorithms. The framework subdivides the configuration space (C-space) into regions and uses (sequential) sampling-based planners to build roadmaps in each region. The regional roadmaps are later connected to form a roadmap of the entire free space. By subdividing the space, we reduce the work and inter-processor communication associated with nearest neighbor calculation, a critical drawback and bottleneck to scalability in existing parallel motion planning methods.

We show that our method is general enough to handle a variety of planning schemes, including the popular Probabilistic Roadmap (PRM) and Rapidly-exploring Random Trees (RRT) algorithms. We compare our approach to two other existing parallel algorithms and demonstrate that our approach achieves better and more scalable performance. Our approach achieves almost linear scalability to hundreds of processors on a 2400-core Linux cluster and over a thousand core on a Cray XE6 petascale machine.

Categories and Subject Descriptors

I.6.8 [**Computing Methodologies**]: SIMULATION AND MODELING—*Types of Simulation, Parallel*

General Terms

Algorithms, Performance

Keywords

Robotics, Motion Planning, High Performance Computing

*This research supported in part by NSF Grants EIA-0103742, ACR-0081510, ACR-0113971, CCR-0113974, ACI-0326350, CRI-0551685, CCF-0833199, CCF-0830753, by the DOE, Chevron, IBM, Intel, HP, and by King Abdullah University of Science and Technology (KAUST) Award KUS-C1-016-04.

SC'11 Companion, November 12–18, 2011, Seattle, Washington, USA.

1. INTRODUCTION

Motion planning is a field of study in robotics that deals with problems of finding a path for a robot or any movable object in a given environment filled with obstacles. Motion planning problem is to find a path, starting at the initial configuration of the robot and terminating at the goal configuration, while avoiding collision with obstacles [1]. This problem is no longer limited to robotics but plays a significant role in a number of other application areas including computer animation, medical surgery, computer-aided design (CAD) and computational biology [1].

Many variants of motion planning algorithms and methods have been developed over the years. Currently, the most prevalent are the sampling-based approaches. These approaches are efficient and can be applied to solving high dimensional problems.While not guaranteed to find a solution, they are probabilistically complete meaning that the probability of finding a solution if one exists, increases with the number of samples generated.

Even with this advance in algorithms, the efficiency of sampling-based motion planning algorithms is not sufficient for current needs. For example, the authors in [5] reported that it takes several hours on a desktop PC to compute a roadmap modeling the folding motion of a small protein. This time increases to several weeks if more accurate energy calculations are used. Thus, these applications are limited in the problem sizes they can handle and the level of detail they can afford to employ. For many application areas, parallel computing offers the advantage of not only reducing computation time, but also improving the solution quality as well. As such, there has been some research effort into parallelizing existing algorithms.

In this research, we present a strategy for parallelizing sampling-based motion planning algorithms. Our proposed strategy uses C-space subdivision to achieve scalability. We first divide the problem into separate pieces for each processor. Then, each processor independently applies a sampling-based planner in their portion of C-space and produce a regional roadmap. Finally, we connect the regional roadmaps to form one roadmap representing the full C-space. By subdividing the space,we reduce the amount of work and inter-processor communication that is incurred during nearest neighbor calculations required for roadmap connection.

2. RELATED WORK

For a detailed survey of early work in general parallel motion planning, please see [2]. Recently, research efforts focus on parallel sampling-based motion planning due to the success of sequential sampling-based motion planning in solving high dimensional problems[3]. We briefly present two previous parallel sampling-based motion planning algorithms.

Parallel PRM (pPRM) directly parallelizes sequential PRM [5]. Briefly, each processor generates an "equal" number of nodes in the entire C-space in parallel and adds them to the roadmap. Then, each processor attempts to connect its nodes with their k nearest neighbors in the entire roadmap. The major drawback of this approach is the all-to-all computation and communication involved in the $O(n^2)$ method used to find nearest neighbors.

In the Parallel Sampling-based Roadmap of Trees (pSRT) [4], the nodes of a pSRT roadmap are trees instead of individual configurations.The collections of these trees form the roadmap. Connections between trees are attempted between closest-pairs of configurations between the two trees. The authors adopted the master-slave architecture. Each slave processor computes a predefined number of trees in the entire C-space. The master is responsible for arbitration of tree ownership, nearest neighbor computations, and determination of which pairs of trees to attempt for connection. Edge validation is distributed to the slave processes.

In the two approaches mentioned above, and others not listed here, we identify inter-processor communication, redundant computation and load imbalance as the key bottlenecks to scalable performance. To address some of these drawbacks, we use a C-space subdivision approach in our work.

3. ALGORITHM OVERVIEW

The strategy of our approach is given in Algorithm 1. Initially, the problem's C-space is subdivided into regions. These regions are stored in a *region graph*, whose vertices represent regions and whose edges encode the adjacency information between regions. Individual region roadmaps are constructed in Step 3. The framework can handle a variety of planning schemes. In Step 4, the regional roadmaps are connected to obtain a roadmap over the entire C-space.

Algorithm 1 Parallel Sampling-based Motion Planning

Input: The C-Space C, A set of motion planners S, number of regions N
Output: A roadmap graph G
1: Decompose C into N regions
2: Make a region graph $R = (V, E)$ with V and E representing each region and adjacency information between regions, respectively
3: Indepedently construct roadmaps in each region using **any** desired planner $s \in S$
4: Connect regional roadmaps in adjacent regions to form a roadmap G for the entire problem

4. EXPERIMENTAL RESULTS

We implemented and tested the four different algorithms on a 2400-core, 8TB memory LINUX cluster. The first two implementations were based on our proposed approach but with two different strategies as the underlying sequential planner. These two implementations are referred to as pSBMP-RRT and pSBMP-PRM: parallel sampling-based motion planning with RRT and PRM as underlying sequential planners respectively. For the purpose of evaluation and comparison, we implemented pPRM[5] and pSRT[4]. Figure 1 shows the speedup for the four algorithms. From Figure 1, one will observe that our proposed method (pSBMP-PRM and pSBMP-RRT) achieves good scalability compared to the existing methods. To establish our claim for scalability and test the limit of our method, we carried out experiments on a Cray XE6 petascale machine for processor counts from 240 to 1200. The result is shown in Figure 2. We observe that scalability is still possible on a massively parallel machine such as Cray XE6.

For more details on our method, analysis of algorithms and full results, we refer the reader to our research webpage at `https://www.parasol.cse.tamu.edu/amatogroup/research/parallelmp`

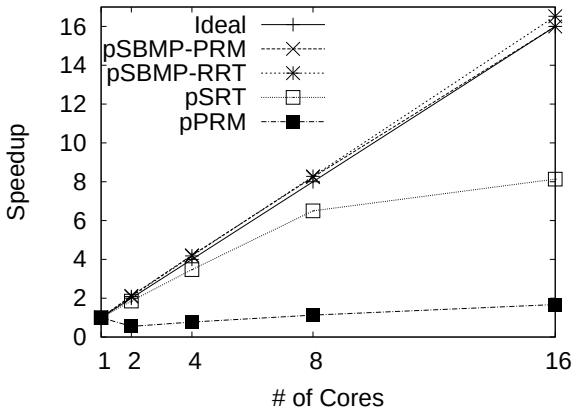

Figure 1: Comparison with other methods

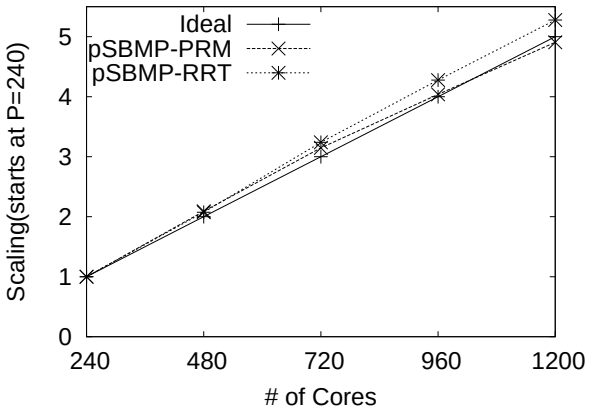

Figure 2: Performance on Cray XE6

5. REFERENCES

[1] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, June 2005.

[2] D. Henrich. Fast motion planning by parallel processing - a review. *Journal of Intelligent and Robotic Systems*, 20(1):45–69, 1997.

[3] D. Hsu, L. Kavraki, J.-C. Latombe, and R. Motwani. Capturing the connectivity of high-dimensional geometric spaces by parallelizable random sampling techniques. In *Proc. IEEE Workshop Randomized Parallel Computing(WRPC)*, 1998.

[4] E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd, and L. E. Kavraki. Sampling-based roadmap of trees for parallel motion planning. *IEEE Trans. Robot. Automat.*, 2005.

[5] S. Thomas, G. Tanase, L. K. Dale, J. M. Moreira, L. Rauchwerger, and N. M. Amato. Parallel protein folding with STAPL. *Concurrency and Computation: Practice and Experience*, 17(14):1643–1656, 2005.