Parallelization on Minimal Dominating Set with Game Theory

Hsin-En, Su 0616215
Po-Yu, Liu 0616223
Yu-Wei, Shih 0616213

1 INTRODUCTION

Minimal dominating set(MDS) is a NP-complete problem, however there are several way to approximate the solution. A game-theoratic approach has been proposed by Li-Hsing Yen [1] which allows the problem be solved in a distributed scenario. Also, they mentioned a synchronous daemon method but not implemented due to its complexity. Which inspired us to try and design a proper parallelized version of the algorithm.

2 STATEMENT OF THE PROBLEM

One special characteristic of the game theory is that it can have more than one solution known as Nash Equilibrium. The solution is depended on the sequence of the actions that players take in the game system. This resembles the characteristic that threads are also not determinisitc, but depends on the scheduling of the operating system. So we are interesting in benefits we might gain by mimicking players in the game system with threads.

3 PROPOSED APPROACHES

As mentioned above, we want to mimick the players in the game system with threads. We will create a thread for every node in the graph. To avoid simultaneos action between neighboring nodes, we give a backoff probability to each player to reduce the conflict rate. Note that having a conflict will not make the result wrong but only to slightly increase the computation time. Nodes that did not backoff will check the status of their neighbors and compute its utility, then make a decision to either set itself on or off. This process is repeated until the graph find a Nash Equilibirum, and no one continues to change their decision.

image1.png

4 RELATED WORK

Li-Hsin Yen's work[1] is highly related to our project, however our focus is mostly different. Their work is focusing on solving the problem with game theory, while our project is to solve the algorithm in parallel to boost the computation time on the NP-complete problem.

5 LANGUAGE SELECTION

In this project, we choose OpenMP and Pthread for multi-threading methods and implement the project in C++. OpenMP is a convenient option in C++ 11 or later versions. However, we might require some of the flexibility of pthread due to the idea to mimick the players with thread, so we might use both of them to implement our algorithm.

6 EXPECTED RESULT

We expect that the computation time will be relatively short due to players can now simultaneously make decisions.

7 TIMETABLE

Work	Deadline	Remarks
Reading paper	11/11	algorithm and paper
Finish sequential code	11/25	maybe in C++
Finish parallel code	12/9	need to discuss before
Optimize parallel	12/30	*
Finish fianl report	1/6	need work division
Finish research	1/6	yeah!

8 REFERENCES

[1] Li-Hsing Yen Member, IEEE and Zong-Long Chen. "Game-Theoretic Approach to Self-Stabilizing Distributed Formation of Minimal Multi-Dominating Sets"

[2] H. Kakugawa ; T. Masuzawa. "A self-stabilizing minimal dominating set algorithm with safe convergence"

[3] E. W. Dijkstra, âĂIJSelf-stabilizing systems in spite of distributed control"

[4] N. Guellati and H. Kheddouci, âĂIJA survey on self-stabilizing algorithms for independence, domination, coloring, and matching in graphs,"

[5] Z. Xu, S. T. Hedetniemi, W. Goddard, and P. K. Srimani, "A synchronous self-stabilizing minimal domination protocol in an arbitrary network graph,"

[6] N. Megiddo, "Applying Parallel Computation Algorithms in the Design of Serial Algorithms", 22nd Annual Symposium on Foundations of Computer Science, pp. 399-408, 1981-October.