A Survey on Game Tree Search approach on GPU

Ms. Rutuja U. Gosavi
Mrs Archana S. Vaidya
[1]Student,ComputerEngg.
Department, GES's R. H. SapatCOE Nashik
Savitribai Phule Pune University, India
[2]Professor,ComputerEngg.
Department, GES's R. H. SapatCOE Nashik
Pune University, India

Abstract-**Game tree search is a classical problem in artificial intelligence and game theory. GTS algorithm is generally used in real time applications like video games, chess, Go etc. Comparison can be done with CPU-based and GPU-based performance by applying GTS algorithm. This algorithm is used for finding player's next move in the game in minimum time. GPU works on SIMD approach and it performs much faster than regular PC. Here implantation can be done with the combination of CPU and GPU. CPU can be used for generating number of choices of players' moves as a tree structure and evaluation of these moves can be done by the GPU. Players' move is termed as nodes and these nodes can be evaluated parallely by GPU's thread. System is viewed as combination both BFS and DFS for three generation on CPU and evaluation on GPU respectively. Acquire algorithm to other games of much higher complexity to find best real time response, such as CHESS, SUDOKU, HEX to weigh up the effectiveness and efficiency of proposition.**

Index Terms— **CHESS, HEX, GPU, GTS, Parallel Computing,**

I. INTRODUCTION

Graphics Processing Unit is a programmable logic chips that renders images, animations and video for the computer's screen. GPUs are located on plug-in cards, in a chipset on the motherboard or in the same chip as the CPU.

GPU is worked as a powerful support for massive parallelism as it works as a parallel computing platform and programming model created by NVIDIA.

A GPU performs parallel operations on data to render images for the screen. Although it is used for 2D data as well as for zooming and panning the screen, a GPU is essential for smooth decoding and rendering of 3D animations and video. The more sophisticated the GPU, the higher the resolution and the faster and smoother the motion in games and movies. GPUs on stand-alone cards include their own memory, while GPUs in the chipset or CPU chip share main memory with the CPU.

GPU features include

- 2-D or 3-D graphics
- Digital output to flat panel display monitors
- Texture mapping
- Application support for high-intensity graphics software such as AutoCAD
- Rendering polygons
- Hardware overlays
- MPEG decoding

A GPU is not only used in a PC on a video card or motherboard; it is also used in mobile phones, display adapters, workstations and game consoles. Lots of applications have get benefits from the massive parallelism capability of GPU [2][3][4]. It is a way to solve compute intensive AI problems due to its SIMD architecture specialized for parallel computing. Single Instruction Multiple data architecture describes computers with multiple processing elements that perform the same operation on multiple data points simultaneously. GPU works on SIMD approach hence it is best method to find nest move in computer games.

A. CUDA

CUDA stands for Compute Unified Device Architecture, is a parallel computing platform and programming model created by NVIDIA and implemented by the graphics processing units.

CUDA gives developers direct access to the virtual instruction set memory of the parallel computational elements in CUDA GPUs.

CPUs like Intel Core 2 Duo and AMD Opteron are good at doing one or two tasks at a time, and doing those tasks very quickly. Graphics cards, on the other hand, are good at doing massive number tasks at the same time, and doing those tasks relatively quickly.

CUDA consist of following features:

1. CUDA is only well suited for highly parallel algorithms
2. CUDA is extremely well suited for number crunching
3. CUDA is suited for large datasets
4. Writing a kernel in CUDA
5. Writing programs with CUDA

B. Game Tree Search on GPU

According to game theory GTS is a graphical representation in which nodes are positions in a game and edges are moves. Game tree in a game represents initial position at the start and contains all possible moves from each node.

Game Tree Search is a combinatorial game theory problem, it is difficult to find an optimal solution for many computer games like Connect6 [6] and Chess [7] due to their exponential time complexity. Game tree search is responsible for finding better GTS algorithm to obtain near-optimal solution in various games e.g. online computer game, decision tree, expert system and etc.

C. Structure of Graphical Processing Unit

To satisfy such a requirement like to reduce computational time for games, parallel computing technologies introduced to improve the performance of GTS algorithms. CPU-based parallelism methods have been studied for many years [8] [9].

The parallel computation on the GPU is performed as a set of concurrently executing thread blocks, which are organized into a 1D or 2D grid. They can be 1D, 2D or 3D with each thread designated by a unique combination of indices. The hardware schedules the execution of blocks on the multiprocessors in units of 32 threads called warps.

Computing on graphics processing units is the utilization of a GPU, which typically handles computation only for computer graphics, to perform computation in applications traditionally handled by the CPU.

The use of multiple graphics cards in one computer, or large numbers of graphics chips, further parallelizes the already parallel nature of graphics processing.

GPU works on a SIMD approach and it is best method to find the best move in a computer game. Graphics processing Unit achieve in some specific tasks higher performance than conventional processors. These basic tasks are decomposed into higher number of smaller subtasks that can be processed con-currently by graphics card. It improves the computational time to find answer which is the feasible optimal solution.

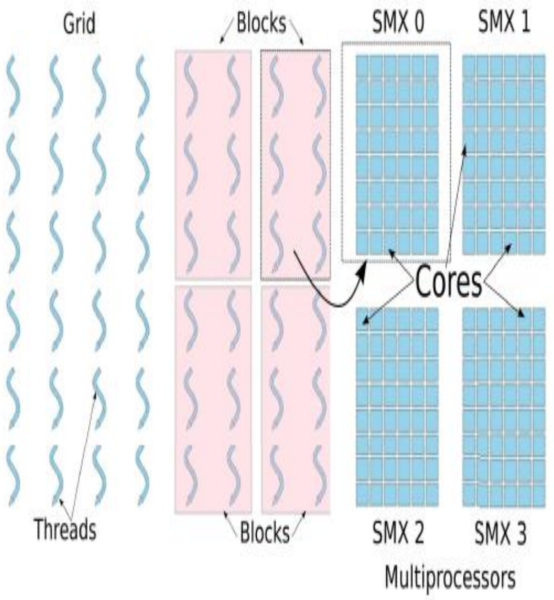

Fig 1: Structure of GPU

II. BACKGROUND AND MOTIVATION

Goal of GTS is finding player's nest move that maximizes his/her chances of winning. In game theory it is hard to find near optimal solution because of its time complexity. It is important thing to find near optimal solution and to accelerate the speed of GTS for real time applications [5].

Many applications are getting benefits from massive parallelism capability of GPUs.

GPUs have independently developed to perform data-parallel computation using multiple cores, the main fact to take advantage of this technology by performing some computation on GPUs that has traditionally been done on CPUs. A recent trend in computer architecture is the move from traditional, single core processors to multi-core processors and further to many-core or massively multi-core processors. The result of this trend is that computational problems which can take advantage of multiple threads with significant speed up. Hence GPU is a feasible way to improve the performance of GTS.

Thus it is essential to investigate if GTS can get benefit from GPU and compare with GPU-based approach with CPU-based approaches.

III. LITERATURE SURVEY

In previous study classical tree-based algorithm was used. Tree-based GTS forms a tree structure which includes number possible moves of player.

In this algorithm CPU traverse choices one after another in DFS manner. Algorithms are listed below:

Table I: Algorithms on CPU

Algorithms	Pros and cons
PVS- Principle Variation Splitting	1. There is no synchronization among processors. 2. Processor who finished its task needs to wait for other processors finish to start up another around of calculation 3. It increases time of calculation because of serial approach.
EPVS- Enhanced Principle Variation Splitting	1. Subtree is allocated to ideal processor 2. The performance and efficiency is improved 3. The enhanced method will bring extra communication overhead.
DTS- Dynamic Tree Splitting	1. It uses a peer-to-peer model on multi-processor systems. 2. Compared with PVS and EPVS algorithms, the advantage of DTS algorithm is its usability and scalability. 3. DTS does not split the tree by specific nodes. 4. DTS algorithm can search nodes concurrently and efficiently because addition of processor is oversimplified as it uses point-to-point protocol, which results in high scalability in the algorithm. 5. Still these three algorithms are tree-based.

IV. GTS ALGORITHM ON GPU

According to previous studies some challenges arise while working with GPU:
1. Complexity of algorithm design on SIMD architecture.
2. Low performance of divergence on GPU for rule-based computer games.
3. Low pruning efficiency of parallel GTS algorithms.

Solution of these problems depends on exploiting parallelism potential of GPU [1].

To solve the above challenges, a node based parallel method to utilize the potential of GPU can be adopted.

The idea is to use node-based game tree search algorithm on GPU. GPU consist of multiple thread structure hence it will process game tree nodes simultaneously.

A. NODE-BASED APPROACH

To solve the challenging problems in Section III following ideas to solve GTS problems on GPU:

1. . To adopt node-based parallel computing for game tree search.
   The tree-based approach is not suite for GPU architecture. The approach is assigning a set of nodes from one or multiple subtrees to processors, while the tree-based approach is while the tree-based approach is assigned to processors. The benefit of method is not only taking advantages of the high concurrency of GPU, but also avoiding the complexity of tree splitting.

2. The combination of depth-first and breadth-first search.
   Commonly, there are two methods to search the tree, the depth-first and the breadth-first search.
   In BFS method all threads evaluates node simultaneously. DFS method is responsible for traversing tree structure.

3. Hybrid programming on both CPUs and GPU.
   As we are using BFS-DFS method, hybrid programming is achieved through GPU-CPU combination respectively. CPU is responsible for maintaining tree structure and depth first search of generated tree. GPU is responsible for evaluating all nodes of the tree simultaneously i.e. breadth first search. Therefore, third method is to use both CPU and GPU architecture in GTS algorithm [1].

B. APPROACHES

Using above three ideas CPU will generate tree structure and GPU will evaluate number of nodes.

A representative methods can be used in computer game includes minimax algorithm, negamax algorithm with alpha-beta pruning etc. [10]. This can be applied many games like SUDOKU, GO, Connect4/6, Chess etc.

Node-based GTS algorithm can be tested with this approach and results are seen whether algorithms can get benefit from it or not.

IV.CONCLUSION

This review paper focuses on a Parallelization of Node based Game Tree Search Algorithms on GPU. Parallel GTS algorithm presented three approaches for obtaining speedy optimal solution of real time computer games on GPU.

By use of both CPU and GPU architecture, the approach can take advantage of the capability of GPU to compute massive nodes in parallel and GPUs flexibility to accelerate tree search and pruning. This approach can be tested by implementing it for SUDOKU, HEX and CHESS games. Results of implementation can be compared with serial implementation of game tree search.

ACKNOWLEDGMENT

We are glad to express our sentiments of gratitude to all who rendered their valuable guidance to us. We would like to express our appreciation and thanks to Prof. Dr. P. C. Kulkarni, Principal, G. E. S. R. H. Sapat College of Engg., Nashik. We are also thankful to Prof. N. V. Alone, Head of Department, Computer Engg., G. E. S. R. H. Sapat College of Engg., Nashik. We thank the anonymous reviewers for their comments.

REFERENCES

[1]Liang Li, Hong Liu, HaoWang, Taoying Liu, Wei Li, “A Parallel Algorithm for Game Tree Search using GPGPU”, IEEE Transaction on Parallel and Distributed Systems,31 July,2014

[2]X. Huo, V. T. Ravi, W. Ma, and G. Agrawal, “Approaches for parallelizing reductions on modern GPU”, International Conference on High Performance Computing (HiPC), 2010, pp. 1-10.

[3]W. Ma and G. Agrawal, "An integer programming framework for optimizing shared memory use on GPU", International Conference on High Performance Computing (HiPC), 2010, pp. 1-10.

[4]J. Soman, M. K. Kumar, K. Kothapalli, and P. J. Narayanan, "Efficient Discrete Range Searching primitives on the GPU with applications", 2010 International Conference on High Performance Computing (HiPC), 2010, pp. 1-10.

[5]A. Bleiweiss, "GPU accelerated pathfinding", The 23rd ACM Symposium on Graphics Hardware, 2008, pp. 65-74.

[6]I.C. Wu and D.Y. Huang, "A New Family of k-in-a-Row Games", Advances in Computer Games, H. van den Herik, S.-C. Hsu, T. Hsu, and H. Donkers, Eds. Springer Berlin /Heidelberg, 4250:180-194, 2006

[7]C. E. Shannon, "Programming a computer for playing chess", Philosophical Magazine Series 7, 41(314):256-275, 1950.

[8]M. G. Brockington, A Taxonomy of Parallel Game-Tree Search Algorithms, 1996.

[9] R. M. Karp and Y. Zhang, "On parallel evaluation of game trees", The first annual ACM symposium on Parallel algorithms and architectures, 1989, pp. 409-420.

[10]P. Borovska and M. Lazarova, "Efficiency of parallel minimax algorithm for GTS", The 2007 international conference on Computer systems and technologies, 2007, pp. 14:1-1 4:6.