IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 11, November 2015

A Review on Hybrid Approach for Game Tree Search on GPU and CPU

Dipali V. Patil[1], Kishor N. Shedge[2]

M.E. Student, Department of Computer Engineering, SVIT, Nasik, India[1]

Assistant Professor, Department of Computer Engineering, SVIT, Nasik, India[2]

Abstract: In the field of game theories and artificial intelligence Game tree search is a classical problem. The general use of GTS algorithm is in real time applications having much higher complexity like video games, chess, connect6, Go etc. Different algorithms for game tree are used to find the player's next best move on the game tree in minimum time. Main focus of the system is on increasing massive parallelism capabilities of GPUs to accelerate the speed of game tree algorithms and propose a general parallel game tree algorithm on GPUs. In game tree search, GPU surpasses a single CPU if high level of parallelism is achieved because of its searching is in BFS manner and CPU is in DFS manner so that CPU did not produce improvement. Here combination of DFS and BFS is main focus and selection will be the depth-first search on CPU and use breadth-first search on GPU. CPU can be responsible for generating number of choices of players' moves as a tree structure and parallel evaluation of these moves can be perform using GPU. It intends to look into a hybrid CPU-GPU solutions.

Keywords: SIMD, GPU, GTS, SUDOKU, Parallel Computing.

I. INTRODUCTION

Many applications [4] [5] [6] have get advantage from the parallelism capability of GPU. Some AI problems can be easily solved by GPU because of its SIMD architecture special for parallelism. GPU is stands for Graphical Processing unit. Single Instruction Multiple Data (SIMD) architecture of computer having many processing elements (PE) which perform the same operation on multiple data points simultaneously and it exploits the data level parallelism. On the SIMD, single instruction computations are performed at a time. CUDA development toolkit supports the parallel work and implemented on GPU. In AI, Game tree search is important approach and GTS is used to find the best move for computer games. Parallel computation on GPU is performed as a concurrently executing thread blocks set. These are organized into a 1D grid or 2D grid.1D, 2D or 3D grid with each thread designated by unique combination of indices. The hardware schedules the execution of blocks on the multiprocessors as units of 32 threads called as warps. Computing on graphics processing units handles computation only for computer graphics and handled by GPU, but computation in applications traditionally handled by the CPU.

A. Game Tree Search

Game tree is a directed graph whose nodes are positions and edges are moves. Complete game tree of game is the game tree Starting at the initial position and having all possible moves from each position. The figure 1 shows the first two levels, in the game tree for the game tic-tac-toe. Three choices of move has available for First player: in the center, at the edge, or in the corner and the second player has four choices for the reply if first player played in the center, otherwise two choices and game is continue. GTS is combinatorial problem therefore hard to find an optimal solution for many games like Chess and Connect6; hence focus is find better GTS algorithms to obtain close-optimal solutions.

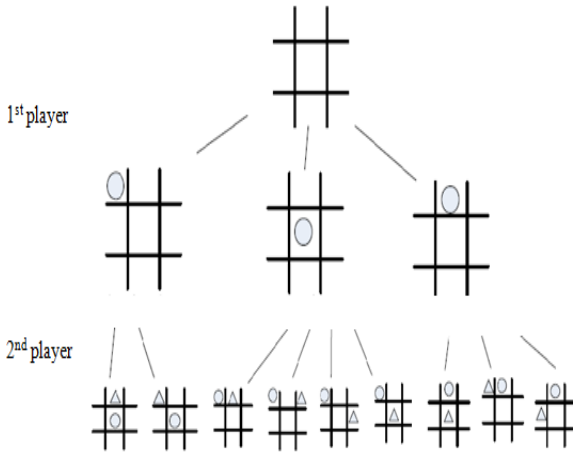

Fig1. Game tree for tic-tac-toe

B. CUDA

For highly parallel algorithms CUDA is best platform. Compute Unified Device Architecture (CUDA) mostly used for parallel computing. NVIDIA created the programming model of CUDA and is implemented over the GPUs. CUDA provides direct access to the virtual instruction set and memory of the parallel computational elements in GPUs.

In CUDA, code executed on the GPU in the form of functions called as kernels. On the GPU for the efficient implementation of kernels, must consider the limited amount of resources like on-chip memory and registers. The properties of the GPU are presented by its compute capability and are queried at run-time. It is used to adjust

the kernel parameters. Usual template of operation in the CUDA kernels is to copy the data from global memory to shared memory, process and copy the results back. All these steps are executed in parallel.

II. LITERATURE SURVEY

In the previous studies different game tree search algorithm have been proposed. Parallel GTS algorithm is the best option to increase the speedup of computer games. Depend on the granularity of parallelism, parallelism on the GTS algorithms can be divided into two types, tree based parallelism and node base parallelism. For the Tree-based Game Tree Search tree structure is form and which includes number possible moves of player. Node based GTS methods compared with existing GPU based methods used nodes instead of tree as a basic searching unit. Description of the different GTS algorithms is given below.

George karypis and vipin Kumar [8] was consider a tree based Game tree search for the single instruction multiple data (SIMD) machines. Tree search on SIMD machines consist of two pats. First is triggering mechanism and second is redistribution mechanism. Parallel searching on the unstructured tree provide better efficiency on the SIMD computers. Also, cost of building large scale parallel computer is high on MIMD so is better choice.

For the multiple processor platforms APHID: Asynchronous parallel game tree search method provided by Brockington and Schaeffer [9]. Which consider the tree based parallelism for game tree. As compare to synchronous methods for determining the minimax value Asynchronous game-tree search algorithms can be efficient or better.APHID makes the algorithm easy to integrate into the sequential game-tree searching program. APHID having better speedup as compare to synchronous searching methods. Integrating the APHID into the existing alpha-beta searching program is easy.

P. Borovska and M. Lazarova[10] was proposed minimax algorithm. Minimax is a game tree search algorithm divided into two logically stages, the first one for the first player which is the computer and the second one for the second player that
is the human. The algorithm tries to find the best legal move for the computer even if the human plays the best move. It means, it maximizes the computer score when it chooses the computer move, while minimizing that score by choosing the best legal move for the human when chooses the human move.

M. S. Campbell and T. A. Marsland[11] was proposed negamax algorithm. Negamax is a similar algorithm for MiniMax with only one small difference that is, it uses only the maximization function instead of using both maximization and minimization functions. This can be done by negating the value that is returned from children from the opponent's point of view rather than searching for minimum score.

D. E. Knuth and R. W. Moore [12] was presented Alpha-Beta algorithm. Alpha-Beta Algorithm is an intelligent modification that can be applied in MiniMax or NegaMax algorithms. Kunth and Moore proved that many branches could be pruned away of the game tree which decrease the time needed to finish the tree, it will give the same result as MiniMax or NegaMax. The main idea of the algorithm is cutting the uninteresting branches over the game tree.

V. Manohararajah [13] presented the principle variation splitting algorithm. PVS is a tree based parallel GTS algorithm using multiple processor. In this PVS algorithm, the initial branch is marked by 1 as a principle node [24]. In game tree, nodes should be serially searched by first processor P0 before beginning of parallel search of other nodes. Other processor has to wait, for finishing the searching of previous one. One's all processor finished their task, best move to player return by PVS. Drawback of PVS, processor who has completed their task needs to wait for another processor.

V. Manohararajah [13] presented the Enhanced principle variation splitting algorithm. EPVS avoid limitation of PVS algorithm and use the multiprocessor platform. In the EPVS algorithm, subtrees are assigned to idle processor from other busy processor So that efficiency and performance is increased. Extra communication overhead will be comes along with EPVS method.

R. M. Hyatt [14] was proposed Dynamic Tree Splitting algorithm for parallel GTS. Peer-to-peer model for multi-processor systems is used for DTS. In this split-points list (SP-LIST) were maintained by which all processors find uncalculated nodes to process. List is empty at the initial state. One processor takes the root node of the game tree and other nodes are in the idle state. If no split points remain in SP-LIST, HELP message will be broadcast to all processors with the help of the idle processor. Busy processor who receives that message will split and copy state of the subtree at SP-LIST. The idle processor goes through SP-LIST again to obtain a split point and search the subtree from that point. If no split point left and all processors stop in an idle state DTS algorithm completes. DTS algorithm is usable and scalable compared with PVS and EPVS

III.MOTIVATIONS

Major Goal of GTS is that finding best move of the player's that maximizes his/her chances of winning. For many computer games, hard to find an optimal solution because GTS is a combinatorial problem in the field of game theory and it also having an exponential time complexity. Hence, finding out near optimal solution is important thing to accelerate the speed of GTS for real time applications such as real time games on computer. Main motivation to use the GPU is that, it processes the thousands of game tree nodes in parallel and many applications gets benefits from its parallelism capacity.

Game tree In the game theory is directed graph in which nodes indicates the positions in a game and whose edges are moves. Game tree starts at the initial position and containing all the possible moves from each position. For the many application areas such as artificial intelligence game trees are the most important because

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 11, November 2015

searching the game tree using the minimax algorithm and its variants is the one way to find the best move for the game.
GPU having more computing power, low power consumption and large memory bandwidth; these factors make them more applicable. CPU have few cores with lots of cache and it can handle only few software threads at one time but On the other side GPU having hundreds of core so it can handle thousands of threads in parallel. Hence it is important to investigate that GTS can get benefit from GPU and compare with GPU-based approach with CPU-based approaches.

IV. PROPOSED SYSTEM

Some challenging problems arise while working with GPU and according to previous studies i.e.Low pruning efficiency of the parallel GTS algorithms, Complexity of algorithm design for SIMD architecture, Low performance of divergence on GPU for the rule-based computer games. To solve these GTS challenges, following node based parallel method to utilize the potential of GPU can be adopted.

A. Node-Based Approach

1. Adopt node-based parallel computing for game tree search.
The tree-based approach is not suite for GPU architecture. The node based approach is assigning a set of nodes from one or multiple subtrees to processors, on other side the tree-based approach is assigned to processors. The use of method is not only taking advantages of the high concurrency of GPU similarly avoiding the complexity of tree splitting.

2 The combination of depth-first search and breadth-first search.
There are two methods to search the tree, the depth-first search and the breadth-first search. For GPU based GTS algorithm, selection is the depth-first search on CPU because of memory limit and use breadth-first search on GPU. In BFS method all threads evaluates node in parallel and for DFS traversing tree structure.

3. Hybrid programming on both CPU and GPU.
Hybrid programming is achieved through GPU-CPU combination respectively using BFS and DFS methods. CPU is maintaining game tree structure and perform depth first search on generated tree and also interacting with GPU. GPU takes tree nodes from CPU is responsible for evaluating all nodes in parallel i.e. breadth first search is performed. Therefore, method is to use both CPU and GPU architecture in GTS algorithm.

B. Architecture
The most common goal of Game Tree Search is finding of the players move so that maximizes his chance of winning. In Game tree, game is spitted into much number of possible choices these are considered as the possible moves which is next move for the player. Many of the choices of the games are computed as sequentially by processor in Depth First Search manner Using tree-based approach. Tree based approach can't be easily used in GPU because of the SIMD technique on GPU. Node-based approach is advantageous over tree-based approach because in which CPU generating the number of possible trees contains the nodes and leaf. On the CPU, creates the number of possible moves in the form of tree. CPU is responsible for execution control and is responsible for maintaining a gametree structure. On the GPU unit by number of threads evaluation of all nodes, leafs takes place. Using this hybrid approach takes an advantage of computation on the CPU in the DFS manner as well as evaluation of nodes by the GPU in a BFS manner.

Using such a combination of CPU and GPU the system architecture is form as shown in the fig 2. Input in the form of Problem data set known as matrix which is provided as an input to CPU. When the matrix is provided as an input to the system, CPU generates number of possible trees contains the nodes as well as leaf. CPU is performs operation like maintaining the tree structure, process data, generation of the all nodes, and the tree pruning, also performs checking of leaf nodes, and in the end solution returned to the root node. Calculation of many tree nodes is done in the same depth in the current gametree, which is the breadth-first search (BFS). In addition, each cycle in the searching process will take in deepest nodes of the current game tree, which is the depth-first search (DFS). That means on DFS approach CPU works to calculate the nodes, since CPU will execute faster as compare to GPU in this situation. And on BFS approach, GPU used for calculating the branch and the leaf nodes in the parallel.

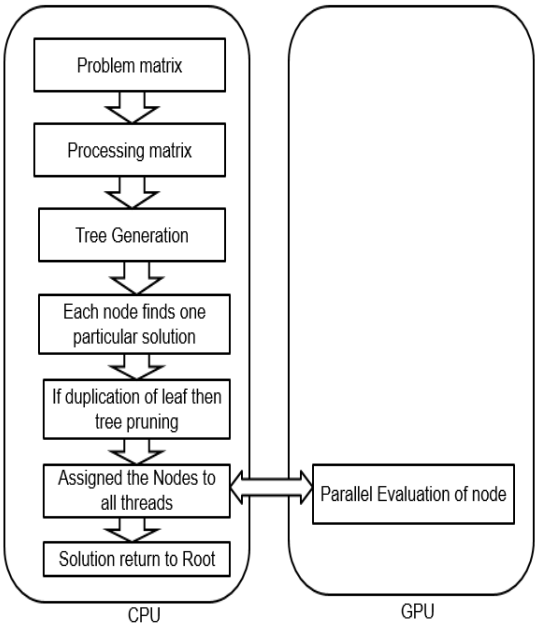

Fig2. System Architecture

V. CONCLUSION

Main focus of this review paper on a Parallelization of Node based Game Tree Search Algorithms on GPU and CPU. Parallel GTS algorithm presented three different approaches for obtaining fast optimal solution of real time

DOI 10.17148/IJARCCE.2015.41108

computer games on GPU. By use of hybrid approach on CPU and GPU architecture, the approach can take advantage of the capability of GPU to compute nodes in parallel and GPUs flexibility to accelerate tree search and also pruning. This approach can be tested by implementing it for SUDOKU, CHESS and connect6 games. Implemented results can be compared with the serial implementation of the game tree search.

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, aug, 2015.
[2] K. Rocki and R. Suda, "Parallel minimax tree searching on GPU," in Parallel Processing and Applied Mathematics, vol. 6067, R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski, Eds., Berlin, Germany: Springer, pp. 449–456, 2010
[3] D. Strnad and N. Guid, "Parallel alpha-beta algorithm on the GPU," Journal in Computing and Information Technology, vol. 19, no. 4, pp. 269–274, 2011.
[4] X. Huo, V. T. Ravi, W. Ma, and G. Agrawal, "Approaches for parallelizing reductions on modern GPU," International Conference on High Performance Computing (HIPC), pp. 1–10, 2010.
[5] W. Ma and G. Agrawal, "An integer programming framework for optimizing shared memory use on GPU," International Conference on High Performance Computing, pp. 1–10, 2010.
[6] J. Soman, M. K. Kumar, K. Kothapalli, and P. J. Narayanan, "Efficient Discrete Range Searching primitives on the GPU with applications", International Conference on High Performance Computing (HiPC), pp. 1-10, 2010.
[7] C. E. Shannon, "Programming a computer for playing chess", Philosophical Magazine Series 7, 41(314):256-275, 1950.
[8] G. Karypis and V. Kumar, "Unstructured tree search on SIMD parallel computers," IEEE Transaction on Parallel and Distributed Systems, vol. 5, no. 10, pp. 1057–1072, 1994.
[9] M. G. Brockington and J. Schaeffer, "APHID: Asynchronous parallel game-tree search," Journal of Parallel and Distributed Computing, vol. 60, no. 2, pp. 247–273, 2000.
[10] P. Borovska and M. Lazarova, "Efficiency of parallel minimax algorithm for GTS," in Processing International Conference on Computer Systems and Technologies, pp. 14:1–14:6, 2007.
[11] M. S. Campbell and T. A. Marsland, "A comparison of minimax tree search algorithms," report on Artificial Intelligence, University of Alberta, Canada, vol. 20, no. 4, pp. 347–367, 1983.
[12] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," Artificial Intelligence, vol. 6, no. 4, pp. 293–326, 1975.
[13] V. Manohararajah, "Parallel alpha-beta search on shared memory multiprocessors," Master's thesis, Graduate Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada, 2001.
[14] R. M. Hyatt, "The dynamic tree-splitting parallel search algorithm," ICCA Journal, vol. 20, no. 1, pp. 3–19, 1997.

DOI 10.17148/IJARCCE.2015.41108