PROJECT PROGRESS REPORT

Date: Mar-27-2013

Project Title: Analysis of iterative deepening minimax tree search using hashing based memoization for Abalone

Investigators:
1. Deshpande, Amol (amold1@umbc.edu)
2. Peshave, Akshay (peshave1@umbc.edu)
3. Prakash, Bharat (bhp1@umbc.edu)

Study of Abalone game-play, transposition tables, hashing techniques

There are 2^64 possible game states for the Abalone game tree. However, one of the improvement techniques that has been mentioned in the paper which makes use of transposition table maintains space for storing only 2^22 states which requires a second level of hashing for generating the 22 bits keys for representing game states. Due to the fact that 2^42 possible keys can be mapped to the same row of the transposition table, there is a need to store the original 64 bit key for identifying the state specifically along with the value of the evaluation function associated with that state. Moreover, if the state gets mapped to a row which has been already occupied by the data of other state then this has to be replaced by the new one. This approach has no scope for improving the access time for the frequently referred states and hence even if a particular state is being referred multiple times it can get replaced by the new state being mapped to the same row.

Optimizations to memoization being investigated

According to the current approach described in the paper, if a key representing some state gets mapped to a particular row of the transposition table which is occupied by the data (evaluation function value) associated with some other state, then the data of the old state gets replaced by that of the new one. We propose a new approach for storing the data associated with the states in a transposition table. The main goal of this approach is to improve the average case time required to access frequently referred states and reduce expected number of transposition table collisions for some hash keys. Instead of storing the data associated with only one state in a particular row of the transposition table, this approach stores a chain of dynamically varying length of nodes containing data associated with multiple states. For doing this we need to compute the following:

1. Expected number of marbles remaining on the board after *i* rounds of game-play.
2. Expected number of valid game states (in terms of number of marbles remaining) e

3. Existing in the transposition table after *i* rounds of game-play.
4. Expected number of collisions at a transposition table location
5. Expected number of references to a state hashed to a transposition table location

The average expansion rate of an Abalone game tree is known and is useful in computing these expected values.

Approach for Memoization Optimization

For any state, evaluation function returns the result to be stored along with the 64 bits key representing that state in the transposition table for the future reference. It is crucial to determine the validity of such memoized states specially when the transposition table size is less than that of the size of the total game states space. An important observation is the ***number of marbles*** on the board is monotonically non-increasing. We can use the number of marbles for determining the validity of memoized entries as game-play proceeds.

This can be achieved by storing extra information bits associated with each row in the transposition table. These information bits will represent the number of marbles which were present on the board when the game was in states represented by the nodes of the linked list associated with that row. There can be maximum of 28 marbles on the board of Abalone and at any time the total number of marbles on the board cannot be more than 28 and it goes on decreasing as the game progresses, hence we need to store 5 extra bits along with each row of the transposition table. This number can then be later used by comparing with the current number of marbles on the board for determining the validity of the states stored at any row of transposition table.

Once the second level hashing is done and the state is mapped to a particular row of the transposition table, we compare the current number of marbles on the board and the value o*f the number of marbles* field associated with that row to check whether or not the number of marbles has decreased after the states represented by that row were memoized. If yes, then we delete the states represented by that row and make space for new states. We can do this because, game state can never go back to these states as the number of marbles cannot increase again.

For the purpose of making above mentioned improvements we need to modify the structure of the transposition table as follows:

Structure of the head nodes in transposition table :

1. Number of times the states have been hashed to this transposition table location.
2. Threshold on the number of nodes that can be chained to this transposition table row. This threshold is recalculated every time a collision occurs and an increase or decrease is decided as a function of:
    a. the number of collisions already seen for the 22-bit key of this hash location.
    b. the expected value of future collisions.

   c. the utility value distribution of the chain v/s the utility value of the colliding state.
   d. the number of marbles remaining on the board.
3. Number of nodes chained to that transposition table row
4. Number of marbles associated with all states represented by that row
5. Number of times the node has been referenced for accessing the evaluation function value

Structure of the chained nodes for each transposition table row:

1. Zobrist 64 bit key representing the state of the game
2. Evaluation function output value for the state being represented by that node
3. Number of times the node has been referenced for accessing the evaluation function value

<u>**Pending Work Items**</u>

- Simulation of the transposition table to experimentally derive the chaining threshold function.
- Amortized analysis of the game state retrieval time for our approach.

<u>**Challenges We Foresee**</u>

- Simulation of the 4Mega transposition table entries based on average case parameters such as expansion rate of the game tree without simulating the actual game tree may lead to approximations in the analyses.
- We may not be able to incorporate all the parameters we seek to, in the chaining threshold evaluation function, even though they are intuitively critical to the approach we are pursuing. This may be seen as a consequence of the above mentioned challenge.

Project Schedule

TASK	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11
Groundwork & Literature Survey	✔										
Proposal	✔										
Study of : Abalone game-play, transposition tables, hashing techniques	✔	✔	✔								
Simulation of proposed transposition table structure					█	█	█				
Exploring refinements to memoization data structures and strategy			✔	✔	✔	█	█				
Formalization and analysis of proposed memoization changes					█	█	█	█			
Interim Progress Report					✔						
Draft Report							█	█			
Final Report									█	█	█

W=Week
Weeks highlighted in gray are milestones (W5: Interim Progress Report due; W8: Draft Report due; W11: Final Project Report due)
✔ =Done, █=Pending

Current state of report / Snapshot

Project Title : Analysis of iterative deepening minimax tree search using hashing based memoization for Abalone

Investigators: Deshpande Amol, Peshave Akshay, Prakash Bharat

Keywords: Memoization, Hashing, Transposition Table, Iterative Deepening Search, Abalone, Board Games

Introduction:
This project includes the analysis of iterative deepening minimax tree search using hashing and memoization for the board game Abalone. When there are game trees where frequent repetition and overlap of sub-trees are expected, memoization helps us to improve performance. We have observed that the data structures used for memoization and hashing is a major factor in the analysis of the time and space bounds for this particular problem. We also explore several other techniques to come up with mathematical analysis and modelling to show time and space bounds for iterative deepening minimax tree search and game state memoization.

Previous work:
In their research paper “Exploring Optimization Strategies in Board Game Abalone for AlphaBeta Search”, Papadopoulos A., Toumpas K. et. al. have used memoization coupled with iterative deepening search to achieve upto 25% speedup. The main aim was to to reduce the redundant searching of nodes. They have studied how an appropriate combination of different techniques is useful in achieving the same. To ensure viability of this approach,though, only a fraction of all the game states are memoized. Consequently, transposition table hash-collisions occur at a large rate.

Specific Aims
High number of hashing collisions may be a limiting factor for the speedup achievable through game state memoization. We investigate techniques for reducing collisions for hash keys with high expected value of future collisions while not compromising with availability of frequently repeating states in the transposition table. These include changes to the entries stored in the transposition table, proposing an algorithm to dynamically alter various parameters used and evaluation of the approach.

Methods:
Study of the structure and game-play of Abalone will help in understanding the importance of the techniques and data structures used and start our analysis. Next step is to try and find ways to improve the memoization and hashing techniques which can be useful in improving the time requirement. By calculating different probabilities by which collision occurs, we try to study how

a different way of storing data in the transposition table can help to improve the performance. Also, we will be attempting to mathematically model, and analyse thereafter, the random variations in the iterative deepening search time over multiple searches rooted at the same board configuration. This will help us formulate bounds on the search time as a function of game and transposition table variables.

References:

● Papadopoulos, A.; Toumpas, K.; Chrysopoulos, A.; Mitkas, P.A.; , "Exploring optimization strategies in board game Abalone for AlphaBeta search," Computational Intelligence and Games (CIG), 2012 IEEE Conference on , vol., no., pp.6370, 1114 Sept. 2012

● Albert Lindsey Zobrist, A New Hashing Method with Application for Game Playing, Tech. Rep. 88, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, (1969)

● Cuckoo Hashing for Undergraduates, 2006, R. Pagh, 2006