Research Review – IBM’s Deep Blue [link]

Daniel Meechan

Udacity Artificial Intelligence Nanodegree

Term 1, Project 2 (October 2017)

Goals

The paper describes IBM’s Deep Blue system, which beat Chess World Champion Garry Kasparov in 1997 after many years of development. It explores ‘the rationale that went into the design decisions behind Deep Blue’ and the various factors which contributed to the system’s success. The authors suggest that large searching through parallel processing, non-uniform search, and the complex evaluation function were the more significant factors; while other factors like endgame databases, the extended book, and evaluation function tuning also contributed.

Techniques

The first AI to beat a Grandmaster was Carnegie Mellon’s Deep Thought in 1988. IBM improved on this design with Deep Thought 2, which used an improved evaluation function, parallel search, and an extended book of moves. Building on Deep Thought 2, Deep Blue featured a grandmaster game database and improvements to its evaluation function, parallel computing, and search. To perform parallel searches, one processor was designated as the master, with the remainder as workers. The master searches the top levels of the game tree, then distributes the leaf nodes to workers for further searching. To maximise parallelism and minimise master overload and communication overhead, workers would already have their next job ready to execute before completing their current job.

Deep Blue’s search system used a hybrid of software and hardware search. Hardware search was very fast because of the custom 8x8 silicon chips doing the processing. However, hardware search’s fixed-form nature meant it could not be customised. Additionally, its searches were relatively simple, so it only went 4 or 5 levels deep. Hence, software search typically was used for the upper levels of the game tree, while hardware search was used for the deepest few levels. Software search used a depth-limited minimax search with alpha-beta pruning. Although the hardware search didn’t use pruning, the system still averaged a search speed of 126 million positions per second during the 1997 match against Kasparov and averaged a search depth of 12.2 levels after a 3-minute search. This search used a combination of quiescence search, iterative deepening, non-uniform searches, transposition tables (which stores the results of previous searches in a hash table to avoid repetition), and NegaScout (an alternative to alpha-beta search).

Most of the time between Deep Blue’s loss against Kasparov in 1996 and the victory in 1997 was spent improving the system’s scoring heuristic, which was made up of 8000 components. The algorithm looked at 4 values: material, position, Kings safety, and tempo. Material is the points value assigned to different pieces (like a pawn being valued at 1 and the queen being valued at 9). Position is the number of safe squares your pieces can attack.

King safety evaluates the King’s position and determines its safety. Tempo rates how productive a player’s moves are to developing control of the board. Deep Blue used both a fast and a slow heuristic function: the fast function scored a board state using the pieces on the board and their positions; the slow function scanned column-by-column for certain chess concepts like square control and king safety.

Deep Blue used an opening book made by grandmasters which was comprised of 4000 positions. These positions were chosen because they ‘[emphasized] positions that Deep Blue played well’. The system also used an extended book, made up of summaries of 700 00 grandmaster matches. The extended book guided Deep Blue in situations where the opening book wasn’t applicable. Moves found in the extended book would be given a bonus in the search tree, making it more likely that moves played by grandmasters would be chosen. While Deep Blue did have an endgame database which includes all chess positions using 5 or fewer pieces on the board, it was only used once in the matches against Kasparov.

Results

IBM’s Deep Blue team used a combination of massively parallel computing, complex evaluation functions, and both an opening book and extended book of moves to achieve their victory against Kasparov. The system’s game search used a combination of quiescence search, iterative deepening, transposition tables, NegaScout, and a hybrid of hardware and software to achieve its average of 126 million positions per second. Meanwhile, the use of both fast and slow heuristic functions and non-uniform search allowed the system to adapt its processing to different phases of the game.