# History

Mid 1990s – IBM wanted to build a world-class chess engine

1988 Deep Thought a.k.a. Deep Blue I

* About 700k nodes/s
* Uses 2 processors

1990 Deep Thought 2

* Deep Thought + improvements
  + Medium-scale multiprocessing (about 24 engines)
  + Enhanced evaluation hardware, using “band-aid” mechanisms to correct Deep Thought’s weaknesses in endgames
  + Improved search software
  + Extended opening book

1995 Deep Blue I

* Single-chip engine
* About 1.6 – 2M nodes/s per chip
* Using 216 chips 🡪 about 50 – 100M nodes/s
* Lost to Kasparov badly (4 – 2) in 1996

1997 Deep Blue II

* Rematch won to Kasparov (3.5 – 2.5)
* Using 8000 features instead of 6400 for evaluation
* Adapted from the loss of 1996 to Kasparov
* About 2 – 2.5M nodes/s per chip
* 480 chips

# System

3 layers:

* One of the processors is the master
  + Does the search work and distributes workload
* The other processors are the workers
  + Do a bit of searching
  + Examine and evaluate positions

Tactical positions: ~100M positions/sec as long sequences of forcing moves exist

Quiet positions: ~200M position/sec

In the Kasparov match: observed about 126M positions/sec. Max 330M positions/sec

Uses typical techniques in game-theory AI:

* quiescence search
* iterative deepening
* transposition tables
* NegaScout

Challenges:

* Large search capacity
  + How to best make use of the search capacity? This was novel
    - Search should be highly non-uniform
    - Search should provide insurance against simple errors (meaning that pruning had to be done very carefully to not miss unexpectedly important moves)
* Hardware evaluation
  + Evaluation function is implemented in the hardware (simplifies the programming part)
    - Time to evaluate is constant
    - Adding features doesn’t cost anything in time
    - But it is not possible to add new features to the hardware evaluation, and software patches are painful
* Hybrid software/hardware search
  + Same problem as for the evaluation, not too flexible because of the hardware
  + Horizon effect
* Massively parallel search
  + Integration is complex

# Chess Chip

3 parts:

1. Move generator
2. Evaluation function
3. Search control

## Move generator

8x8 hardwired finite state machine – generates all moves simultaneously, reduces latency

## Evaluation function

Fast evaluation vs slow evaluation, to skip computing an expensive full evaluation when an approximation is good enough.

* Fast evaluation. Done in a single clock cycle. Computes easy stuff like piece placement.
* Slow evaluation. Scans board for things like square control, pins, king safety etc.

## Search Control

Implements null-window alpha-beta search

Null-window eliminates the need for a value stack. But disadvantage is that it is only an approximation.

Hardware search: no transposition table. Not too bad as this is handled by software nicely.

Move stack: in Deep Blue, repetition detection makes it more precise.

# Software Search

Selective search, called “dual credit with delayed extensions”:

* extend forcing/forced pairs of moves – helps for tactics and avoiding horizon effect
* forced moves are expectation dependent – to determine what is a forced move
* fractional extensions – avoid extending all forced moves pairs
* delayed extensions – only extend a ffp when it has enough importance (has been detected multiple times)
* dual credit
* preserve the search envelope – to avoid oscillating search

Based on depth-limited alpha-beta search using negamax.

## Credit generation mechanisms

1. singular, binary, trinary, etc.  
   a move significantly better than all alternatives
2. Absolute singular  
   there is only one legal move a large credit can be given
3. Threat, mate threat  
   using a null move search to detect threats
4. Influence  
   move which are enabled by previous moves. Meaning this explores new branches
5. Domain dependent  
   check evasion, passed pawn pushes

Positions near the root generally received more credit than positions far from the root. This allows the search to not explode.

## Pruning

A new idea is “No progress”: if a move is good for a given side, it is best to play it earlier rather than later.

Only had noticeable results In blocked positions with few pieces.

# Hardware search

On the chess chip.

Fixed-depth null-window search, with quiescence search.

Fast, but relatively simple. Quite shallow, usually 4 to 5 ply.

# Parallel search

480 chess chips, 16 chips per node, 30 nodes. They communicate via a high-speed switch, thanks to a micro channel bus.

* One master nodes, all other slaves.
* Control distribution
* Parallelism
* Synchronization

1. First few iterations: on the master node.
2. As the search gets deeper, jobs get allocated on other nodes. There were big challenges with:
   1. Load balancing
   2. Mater overload – major bottleneck
   3. Sharing between nodes

Deep Blue parallel search is non-deterministic!!

Performance:

* About 8% efficiency in tactical positions
* About 12% in quiet positions

No idea what these mean

# Evaluation function

* A sum of about 8000 different feature values.
* Features can be static or dynamic (scaled over the search).
  + Examples of dynamic features that are dependent on material present on the board:
    - King safety
    - Passed pawns
    - Pawn structure defects
* Features are initialized by the **evaluation function generator** (on the master node). This is expensive!
* The evaluation function generator also links features that are related
* Features like king safety or rooks on files are created and tuned “by hand”
* Some things were done thanks to automated analysis, e.g. detection of noisy features. Hill-climbing approach was used to determine noisiness, and suspicious features were then examined by hand.

**Extended book**

Using a grandmaster games database (700k games) to influence chances of playing a given move, if it was played by grandmasters.

Factors influencing extended book evaluation:

* number of times a move was played
* relative number of times it was played
* strength of the players playing the moves
* recentness of the move
* results of the move
* commentary on the move

**Endgame databases**

Databases with all positions with 5 or fewer pieces, and their expected result. (Helps a lot for Alpha-Beta!)