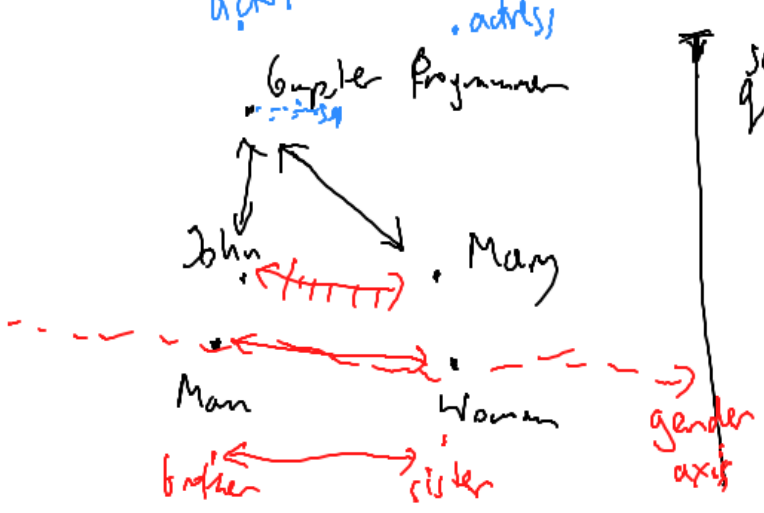
Word embeddings

$\{words\} \longrightarrow \mathbb{R}^N$ e.g. for GloVe, $N \in \{50, 100, 200, 300\}$

Paper

"Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings"



search query = "Common computer science phd student"

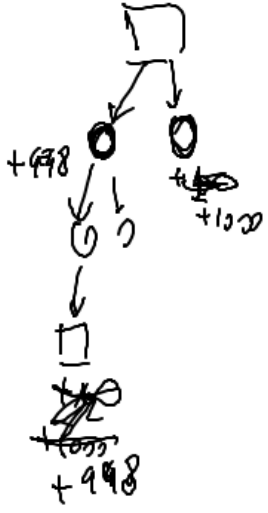
the pages were the same apart from the student's Name

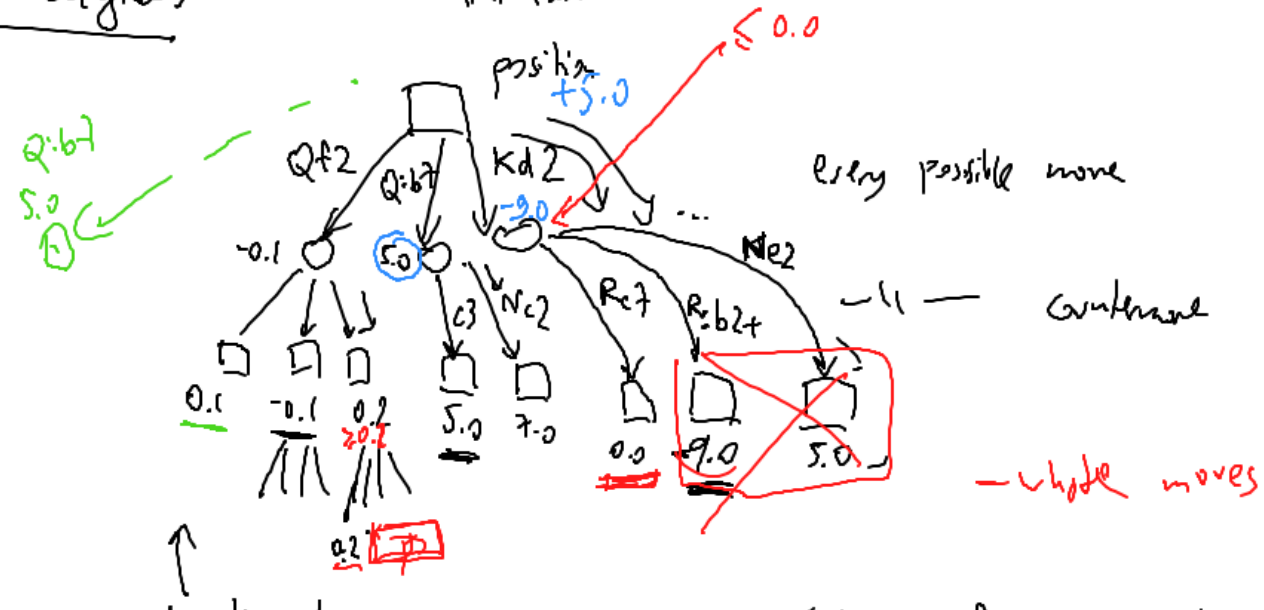
John

Mary

Chess, Go, ... engines

minimax





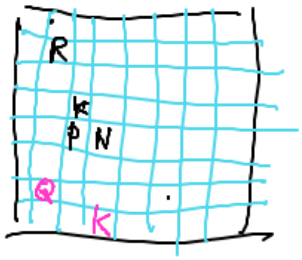
every possible move

— " — continuation

— white moves

in the leaves, evaluate the position from White's perspective

(we need an evaluation function!)



starting position, white to play

α-β pruning ☑ works best if the best move is analysed as first

sorting is ~~done~~ may be obtained by running this algorithm iteratively with depth of the tree increasing

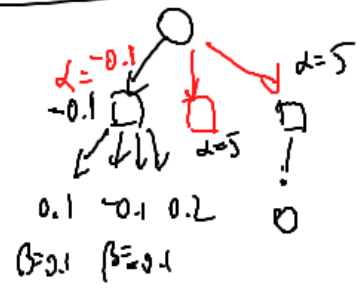
```
def αβ-max(α, β, depthleft):
    if depthleft == 0: return evaluate()
    for (all moves):
        score = αβ-min(α, β, depthleft-1)
        if (score ≥ β):
            return β
        if (score > α):
            α = score
    return α
```

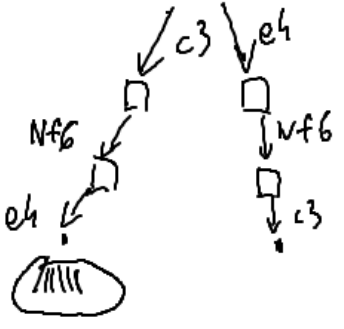
```
def αβ-min(α, β, depthleft):
    if depthleft == 0: return evaluate()
    for (all moves):
        score = αβ-max(α, β, depthleft-1)
        if (score ≤ α):
            return α
        if score < β
            β = score
    return β
```

αβ-max(-∞, ∞, depth)

α

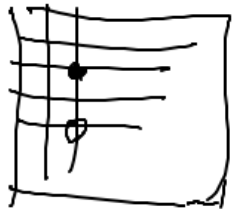
www.chessprogramming.org





Go

α-β pruning & minimax does not work well due to a large branching factor



19x19 = 361

Monte Carlo Tree Search

- Petr Baudiš (Master Thesis)
  Pachi
- Shih-Chieh Huang (PhD Thesis)
  Erica

MCTS

while the time is available:

- s = root Position
- while node(s) is not Leaf:
  - s = $\pi_T(s)$ (select some child of s)

} selection

- n = node(s)
- MaybeExpandNode(n)

} expansion

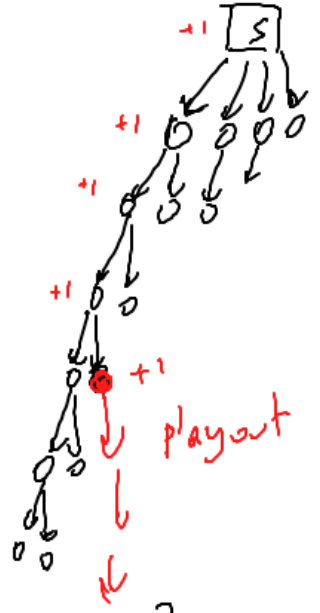
- while s is not final position:
  - s = $\pi_S(s)$ (select position after some move in position s)

} simulation

- result = Evaluate(s) (note: s is the final position, i.e., Game Over)
- while n exists:
  - Update(n, result)
  - n = parent(n)

} back-propagation

Play = Play(argmax$_{n \in child(Root)}$ simulations(n))



selection:

e.g. select the node maximising

$$\frac{w_i}{n_i} + c\sqrt{\frac{\ln N}{n_i}}$$

where

w_i = # wins after move i

n_i = # sims — " —

N = # sims after parent move

c = const. e.g. $\sqrt{2}$

Everything is from White's perspective while to move