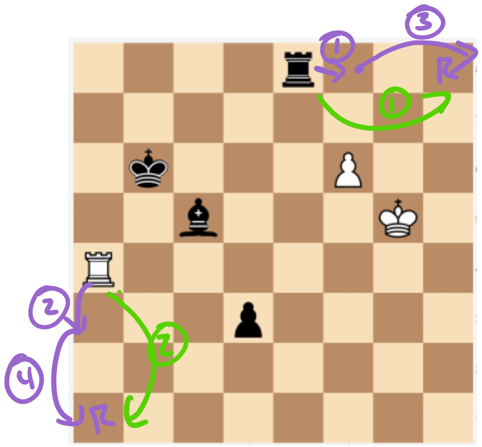
Transposition Table

Positions may be reachable by multiple sequences of moves



Keep table of values for all positions examined in tree

Keys: positions

Values: value/bound, move, depth

Add check at start of A-B

```
if pos in transpo table and searched depth ≥ current depth bound
        if value for pos is exact, return it
        else if upper bound ≤ α return it
        else if lower bound ≥ β return it
```

Save returned values in table

```
        if value ≥ β store lower bound, value, d
        else value ≤ α store upper bound, . . .
        else           store exact value
```

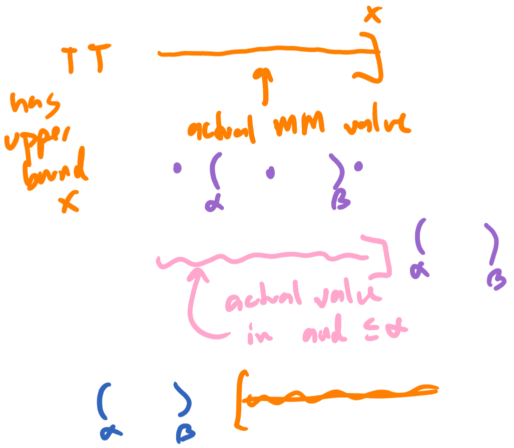


table can get quite large !

replacement policy

least recently used

FIFO

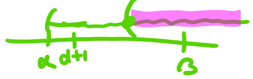
⋮

Scout (principal variation search)

Alpha-beta wants to know
and asks 1) is next child better than best so far, and if so, by how much
α ....(α, β)
result $\leq \alpha$: answer NO
$\underset{\geq}{\text{result}}$ $> \alpha$: answer YES

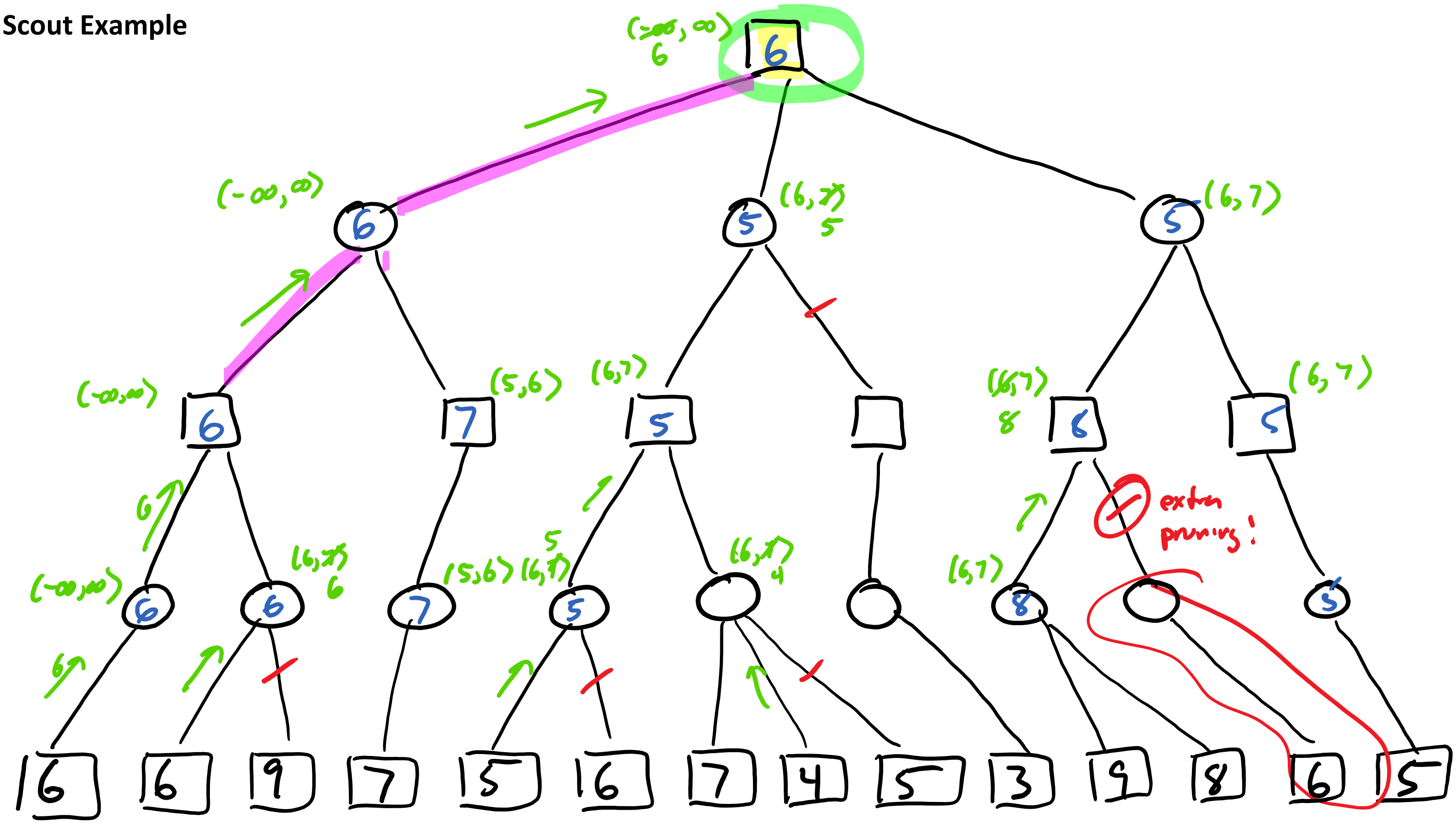
Scout wants to know
and asks 1) is next child better than best so far — pass $\underset{\leq}{(\alpha, \alpha+1)}$ ↑ more pruning
NULL window
NO answer: result $\leq \alpha$
goal: have answer be NO most of time
get answer quickly

2) if answer is Y, how much better is it?
re-search with (result, β) — YES answer result $> \alpha$

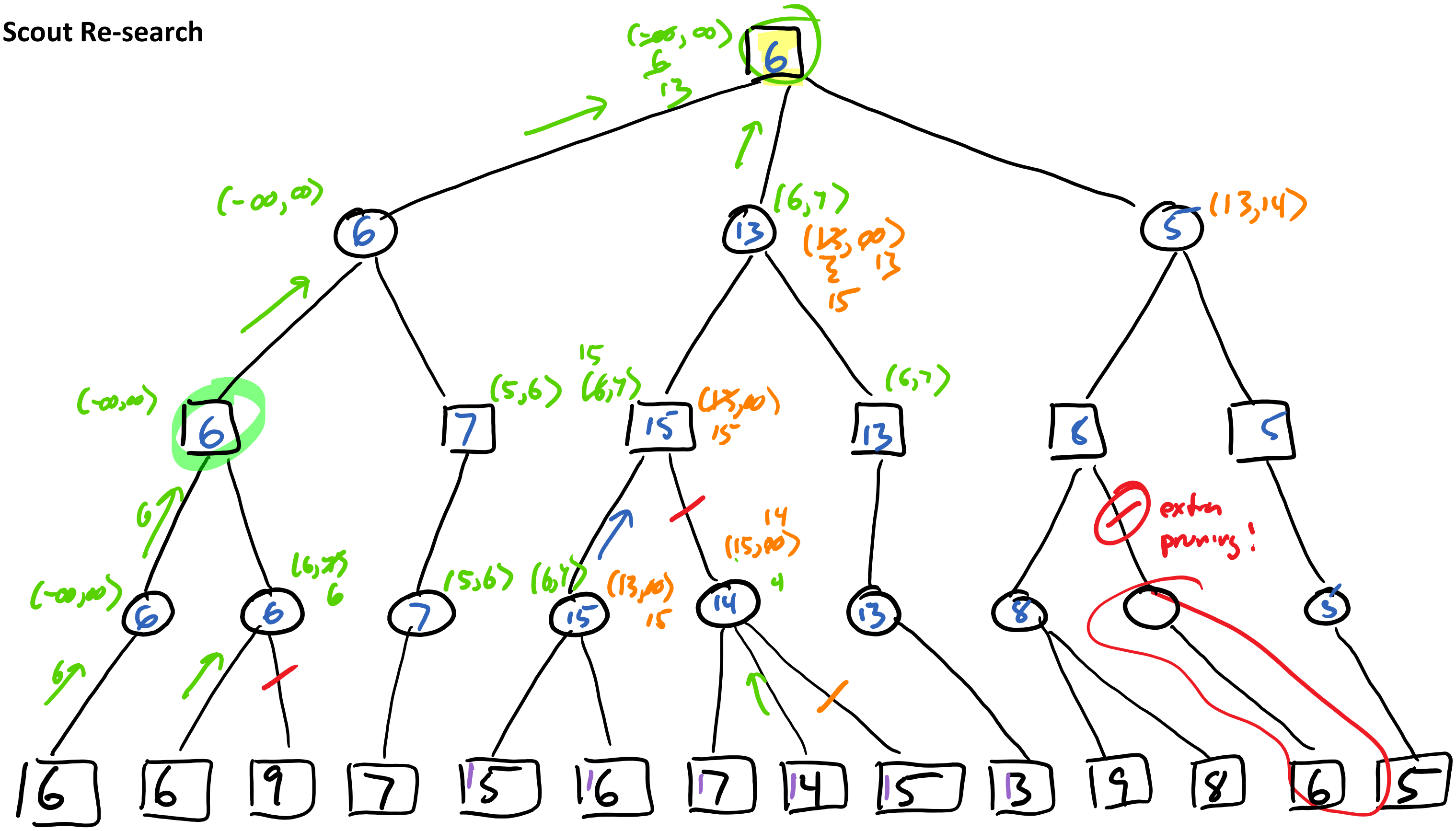


```
Scout (p, α, β, h, depth)
    if p is terminal then return value(p)
    if depth = 0 then return h(p)
    if p is a max position

                                                   in rough order from best... worst
                                                   (using transpo table or fast
                                                    heuristic or...)
        for each reachable position p' and while α < β
            if p' is first child
                score ← Scout(p', α, β, h, depth - 1)
            else
                               AB          null window
                score ← Scout(p', α, α+1, h, depth - 1)
                if  α < score < β                              score ≤ α → Not better than
                                                                           best so far
                    score ← Scout(p', score, β, h, depth-1)   score ≥ β → p's value also
                                                                           ≥ β - α/β
                                                                             cutoff
            α ← max(α, score)
        return α

    else
        ⋮
```

Scout Example



Scout Re-search



MTD-f

node to evaluate

depth bound

first guess

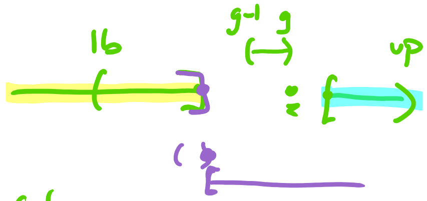
the better, the fewer iterations
correct → 2 iterations
get guess from result of
prev searches

Memory-enhanced test driver

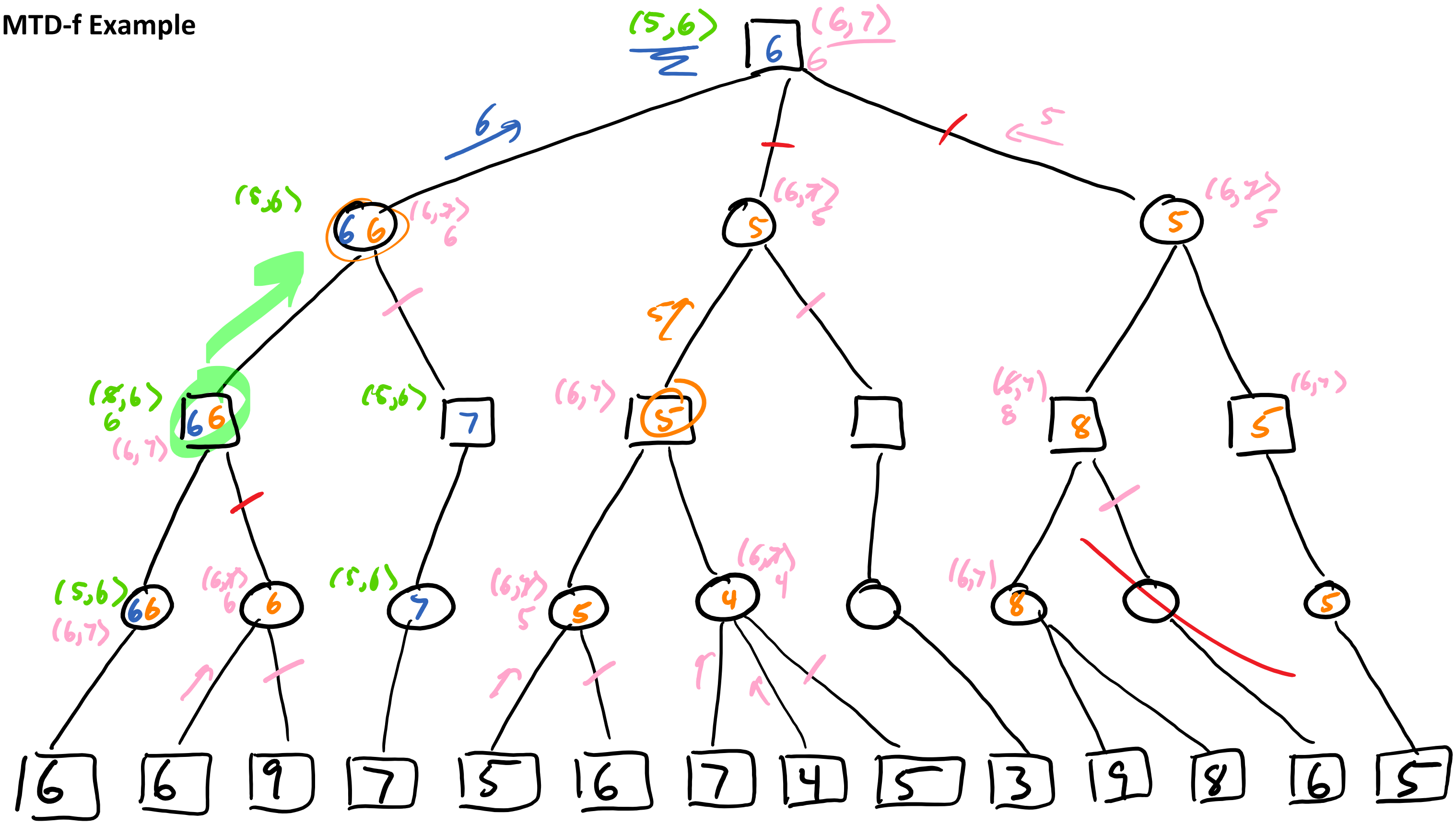
heuristic

current guess of value of n

```
MTD-f (n, f, d, h)
    lowerBound ← −∞
    upperBound ← ∞
    g ← f
    while lowerBound < upperBound
        β ← max(lowerBound + 1, g)
                                                    g-1   g
        g ← A-B-with-transposition-table(n, β-1, β, d, h)
        if   g < β    then upperBound ← g
        else          then lowerBound ← g
    return g
```



MTD-f Example


(5,6)
6
(6,7)
6
6
5
(5,6)
6 6
(6,7)
6
(6,7)
5
(6,7)
5
5
(5,6)
6
6 6
(6,7)
(5,6)
7
(6,7)
5
(6,7)
8
8
(6,7)
5
(5,6)
6 6
(6,7)
(6,7)
6
6
(5,6)
7
(6,7)
5
5
(6,7)
4
4
(6,7)
8
5
6
6
9
7
5
6
7
4
5
3
9
8
6
5