Null Move pruning

Lucas Bradstreet
bradsl03@csse.uwa.edu.au

AAI411 Seminar
Introduction

- Computer chess popular AI field — well known by public.
- Current computer chess players use pruning methods to cut down search tree.
- Pruning done at the cost of search stability...
Recall that Minimax search is at cost $b^m$.

With $\alpha/\beta$ pruning cost can be reduced to $b^{\frac{m}{2}}$. 
**Minimax algorithm with** $\alpha/\beta$

$maxValue$ (Node n, $\alpha$, $\beta$, depth)

if TerminalCondition(n) then
    return utility(n)
end if

for all valid moves for node n do
    Clone node n and update with action
    $\alpha = \text{Min} (\text{minValue}(n, \alpha, \beta), \alpha, \text{depth}-1)$

    if $\beta \leq \alpha$ then
        return $\beta$
    end if
end for

return $\alpha$
Limitations

- Assuming perfect move ordering, cost is now $b^\frac{m}{2}$.
- Reduction in Minimax cost is not enough.
- Deep Blue custom evaluated 200 million nodes per second.
Null Move Pruning

- Introduction to unsafe-pruning.
- Pruning allows deeper searches to be made.
- Worthwhile if tactical mistakes can be limited.
Null Move Pruning - extended

- Uses Null move as an indication of strength of position.
- Idea is that no move is the worst move.
- Humans do this without thinking about it.
- Passes play back to opponent with reduced depth.
- Uses this result to prune.
Minimax with Null move extension

$maxValue$ (Node n, $\alpha$, $\beta$, depth)
if TerminalCondition(n) then
    return utility(n)
end if
if AllowNullMove(n) then
    Make null move
    // R is reduction factor, reduces depth of search
    nullVal = minValue (nullnode, $\beta - 1$, $\beta$, depth - R - 1);
    if $nullVal \geq \beta$ then
        return $\beta$
    end if
end if
Conditions on the Null-Move

- Conditions exist where null move is a good move!
- Includes *zugzwang* positions, frequent in end games.
- Don’t use null move in end-game.
- Cannot be done while King is in check (illegal).
- Don’t use Null-move if parent is a null move.
A zugzwang position

Figure: A zugzwang position — making a null move would be error prone, may lead to loss of a draw. Image from http://www.seanet.com/brucemo/topics/nullmove.htm
Other problems

- With care zugzwang positions can be avoided...
- Main other issue is the *horizon effect*.
- Horizon effect caused by depth reduction in Null Move search.
Adaptive Null-Move Pruning

- Recent work to do with horizon effect.
- Bigger depth reduction faster but may cause tactical errors.
- Best constant depth reduction $R = 2$ [2].
Adaptive Null-Move Depth Reduction

\[ R_{\text{adaptive}} = \begin{cases} 
2 & \text{if } (\text{depth} \leq 6) \text{ or } ((\text{depth} \leq 8) \text{ & } (\text{maxpieces} < 3)) \\
3 & \text{if } (\text{depth} > 8) \text{ or } ((\text{depth} > 6) \text{ & } (\text{maxpieces} \geq 3))
\end{cases} \]  

1. Maximises use of depth reduction 3 where safe.
Results for Adaptive Null-Move

On standard Test Suite chess board configurations, Adaptive $R$ compared to $R = 2$:

- 10% – 30% reduction in number of nodes evaluated.
- Number of games solved is the almost the same as $R = 2$.

Adaptive $R$ compared to $R = 3$

- 17% – 19% increase in number of nodes evaluated.
- Increases number of solutions found by 2%.
Verified Null Move Pruning

- Attempts to limit missed solutions when cutoffs occur immediately.
- Searches null move with $R = 3$.
- If cutoff occurs note it and continue search.
- If child node also has null-move cutoff, return utility.
Null-move search indicates a fail-high:
1. Reduce the remaining depth by one ply.
2. Conduct standard null-move pruning in that node’s subtree.

Figure: Illustration of verified null-move pruning [3]
Verified null move pruning also finds more solutions for tested configurations than $R = 2$ and $R = 3$.

**Figure:** Tree sizes for $R = 2$, $R = 3$ and verified $R = 3$ [3]
Conclusion

- Null move pruning useful technique for cutting down search tree.
- Very effective — used by almost all top computer chess players.
- Verified null move pruning gives biggest reduction and misses less solutions.
References

- **C Donninger.**
  Null move and deep search: Selective search heuristics for obtuse chess programs.

- **Ernst A. Heinz.**
  Adaptive null-move pruning.

- **Omid David Tabibi and Nathan S. Netanyahu.**
  Verified null-move pruning.