Beyond Luck
Mathematics and Games

R. Verfürth

Fakultät für Mathematik
Ruhr-Universität Bochum

www.ruhr-uni-bochum.de/num1

Milan / November 19th, 2012

Overview

Two Main Aspects

- Chance
- Strategy

Importance of Strategy

- Low: Throwing dices, Lotteries
- Medium: Poker, Bridge, Monopoly
- Crucial: Othello, Nine Men's Morris, Checkers, Chess, Go

Structure

- Two players are drawing alternatively.
- For every possible position there is a finite number of possible draws.
- There is a finite number of possible final outcomes (Win, Loss, Tie).
- The gain of one player equals the loss of the other one (zero-sum-game).
- There is no infinite sequence of positions.
- Chance is excluded.

Problems

- Evaluate a given position.
- Find an optimal draw.
- Can winning be enforced?
- Is the game fair?

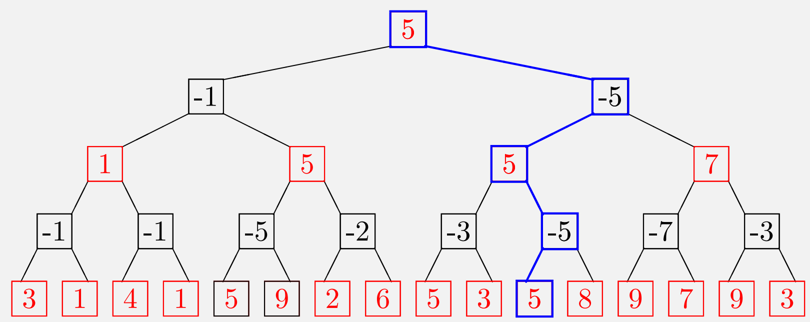
Evaluation of Positions

- $F(p)$ is the value of position p for the player with the right to draw.
- $-F(p)$ is the value for the other player.
- Value of a terminal position:

$$F(p) = \begin{cases} \infty & \text{for a win} \\ -\infty & \text{for a loss} \\ 0 & \text{for a tie} \end{cases}$$

- Value of a non-terminal position?

Tree



Value of a Non-Terminal Position

- Position p allows d possible draws which give rise to positions $p_1, \ldots, p_d$.
- $F(p)$ should give the value of p.
- $F(p) = \max\{-F(p_1), \ldots, -F(p_d)\}$
- Recursive definition!
- Neg-max search

Naive Realisation

- Search the complete tree and evaluate F recursively.
- This is too costly.
- Use a fixed depth for the search or a time-monitor.
- This only yields an approximation for F.

Naive Programme

```
int value(Position p) {
    if( endPosition(p) )
        return F(p);
    int v = - INF;
    while( nextChild(p, q) )
        v = max( v, -value(q) );
    return v;
}
```


Goals

- Try to decide as early as possible whether a given branch of the tree yields a better result.
- Try to prune (cut) inefficient branches as close as possible to the root (= current position).
- Try to achieve an increased search-depth with a shorter computing time.

α-β-pruning (Donald N. Knuth)

- Construct a function $G(p, \alpha, \beta)$ such that:

$$G(p,\alpha,\beta) = \begin{cases} \alpha & \text{if } F(p) \leq \alpha, \\ F(p) & \text{if } \alpha \leq F(p) \leq \beta, \\ \beta & \text{if } \beta \leq F(p). \end{cases}$$

- At the root: $G(p, -\infty, \infty)$

Realisation

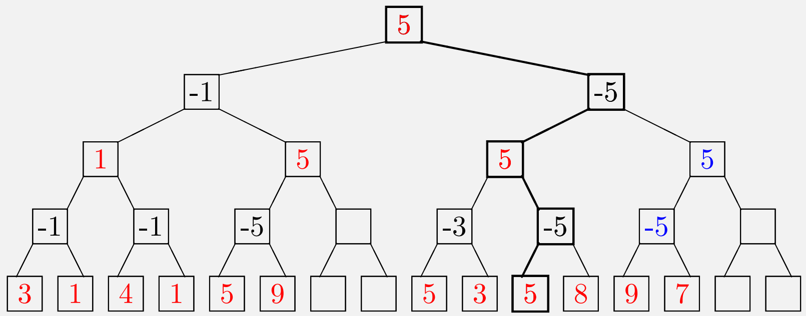
- Initialise the value v of G with $v = \alpha$.
- Stop maximising as soon as $v \geq \beta$.
- On the next level set

$$\alpha_{\text{new}} = -\beta_{\text{old}},$$
$$\beta_{\text{new}} = -v.$$

Improved Programme

```
int value(Position p, int alpha, int beta) {
    if( endPosition(p) )
        return F(p);
    int v = alpha;
    while( nextChild(p, q) && v < beta )
        v = max( v, -value(q, -beta, -v) );
    return v;
}
```

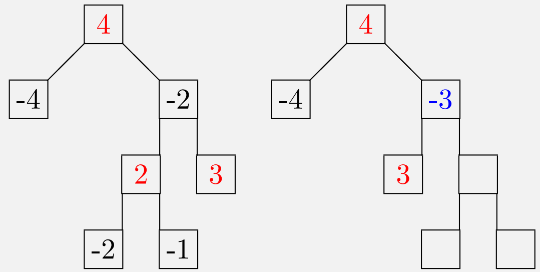

Tree with Pruning



Best Case

- For every position the first possible draw leads to an optimum, i.e. the values are ordered increasingly on every level.
- The algorithm only checks the optimal positions.
- Any other algorithm has at least the same cost.

Counter Example



Worst Case

- There is always a permutation of the positions such that all positions must be checked.
- Any other algorithm has at least the same cost.

Generic Case

- h height of the tree (= number of levels)
- d width of the tree (= number of possible draws per position)
- The cost is proportional to $(r_d)^h$ with a number r_d depending on d.
- $1 \leq r_d \leq d$
- $r_2 \approx 1.8$
- $r_3 \approx 2.5$
- $r_d \approx \frac{d}{\ln d}$

Mathematical Tools

- Combinatorics
- Generating functions for the analysis of the recursions
- Perron-Frobenius Theorems for the asymptotic behaviour of r

Game-Dependent Modifications

- Ad hoc evaluation of positions without search
- Use of symmetries
- Hash-tables of already analysed positions
- Opening libraries
- End-game libraries

Winning Strategies

- Player A has a winning strategy, if player B cannot prevent A from winning even when playing optimally.
- At most one player can have a winning strategy.

Strategies for a Tie

- Player A has a strategy for a tie, if player B cannot avoid a tie even when playing optimally.
- Both players can have a strategy for a tie.

Decidability

- A game is decidable, if one player has a winning strategy or both players have a strategy for a tie.
- A game is fair, if both players have a strategy for a tie.

Known Results

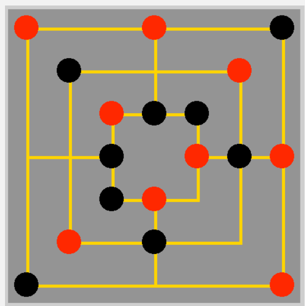
- 4×4 Othello: black wins.
- 6×6 Othello: black wins.
  (Feinstein 1993, 5 weeks on a workstation, 10^{10} positions)
- Nine Men's Morris is fair.
  (Gasser-Nievergelt 1994, 3 years on a PC-cluster, forward-backward-search, 49 sub-spaces with $10^6 - 10^{10}$ positions, use symmetries)
- Checkers is fair.
  (Schaeffer-Lake 2007, forward-backward-search, sub-spaces with up to 10^{18} positions)

Expected Results

- 8×8 Othello may be fair
  (about 10^{37} positions)
- Chess may be fair
  (about 10^{44} positions)
- Go may be fair (??)
  (about 10^{170} positions)

"High Noon"

- Who draws looses.



- Red looses after 37 draws.
- Black looses after 30 draws.

A quoi bon?

- Economy (Nash-equilibrium)
- Data mining
- Classification and regression trees
- Fast solvers for upwind discretizations of diffusion-convection equations

References

J. Bewersdorff
Luck, logic and white lies: The mathematics of games
A K Peters, 2004

Handout of this talk
www.rub.de/num1
Link: Vorträge / Talks

Bachelor theses on "Mathematics and Games"
www.rub.de/num1
Link: Abschlussarbeiten / Theses