Aalto-yliopisto
Perustieteiden
korkeakoulu

# Imporving a minimax algorithm for chess

*Kristian Wasastjerna*

*26.08.2022*

Ohjaaja: *Prof. Kai Virtanen*

Valvoja: *Prof. Kai Virtanen*



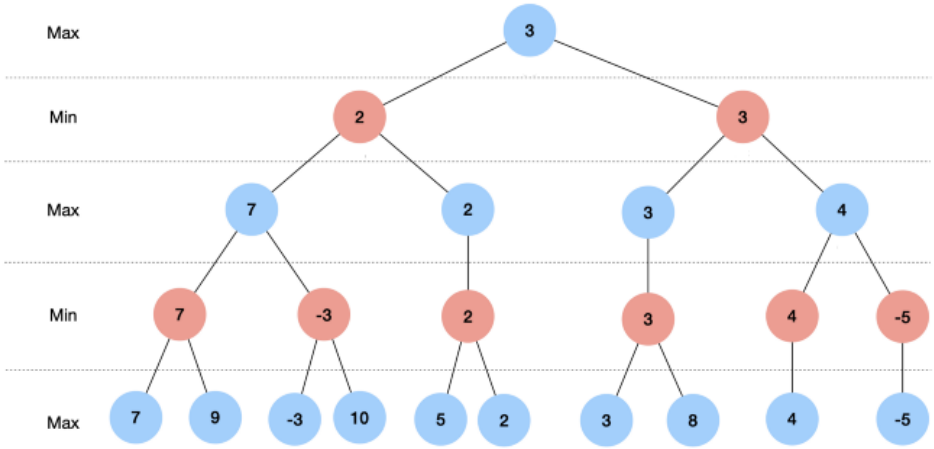
Systeemianalyysin
laboratorio

# Tausta

- Shakkiohjelman kolme osaa
  - Siirtogeneraattori
    - Generoi kaikki lailliset siirrot annetusta positiosta
  - Evaluointifunktio
    - Palauttaa positiosta evaluoinnin
  - Hakualgoritmi
    - Luo siirtogeneraattorin avulla hakupuun ja käyttää evaluointifunktioita antamaan jokaiselle hakupuun solmulle arvon
- Työssä käsitellään vain minimax-hakualgoritmia

# Minimax-algoritmi

- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Puu luodaan ottamalla huomioon juurisolmusta kaikki mahdolliset siirrot n askelta eteenpäin
- Puu ratkaistaan lehdistä juurisolmuun seuraavasti
  - Lehti solmuissa arvo evaluointifunktiolla
  - Minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - Maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto



# Minimax-algoritmin parantaminen

- Algoritmi käy kaikki mahdolliset siirrot n siirtoa eteenpäin
  - Paljon turhaa laskentaa siirtoihin jotka ovat huonoja
  - Lasketaan sama positio useampaan kertaan
- Algoritmi evaluoi lehtisolmut staattisesti
  - ”Horizon effect”
  - Evaluointi voi olla voimakkaasti vääristynyt jo yhden siirron eteenpäin lehtisolmusta
  - Lehtisolmun evaluointi ei ole aina luotettava
- Parannuksilla saadaan eri versioita alkuperäisestä algoritmista

# Algoritmien vertailu

- 24 testipositiota
  - Hakupuiden koko
  - Lehtisolmujen määrä
  - Aika
- Vastakkain pelaaminen - Kolme aloitus positiota
  - Pelataan molemmat puolet
  - Yhteensä kuusi peliä

# ”Alpha-Beta pruning”

- Hakualgoritmi pitää kirjaa maksimoivan ja minivoivan pelaajan siihen asti parhaista löydetyistä siirroista
  - Jos uusi siirto näitä siirtoja huonompi, sitä ei lasketa pidemmälle ja se hylätään
- Ei vaikuta lopputulokseen
- Hakupuun läpikäyntijärjestys vaikuttaa hylättyjen solmujen määrään

# ”Alpha-Beta pruning”

- Pelkistetty minimax-algoritmi
  - Keskimäärin 1,72*10^8 solmua
- AB-algoritmi
  - Keskimäärin 2,98*10^5 solmua
- Vertailtiin neljää vaihtoehtoista hakupuun läpikäyntijärjestystä

| | Average Nodes | Total Nodes | Percentage of Leaf Nodes | Average Branching Factor | Average Time |
|---|---|---|---|---|---|
| Version 1 | $1.089416 \cdot 10^7$ | $2.61459855 \cdot 10^8$ | 88.5215% | 8.88412 | 4.8040 s |
| Version 2 | - | - | - | - | - |
| Version 3 | $1.721357 \cdot 10^7$ | $4.13125714 \cdot 10^8$ | 85.7950% | 8.53434 | 8.9213 s |
| Version 4 | $1.595609 \cdot 10^7$ | $3.82946100 \cdot 10^8$ | 85.4641% | 8.54064 | 8.2946 s |

# ”Null-move pruning”

- Lisäys ”AB-pruning” algoritmiin
  - Kun saavutaan solmuun, toteutetaan suppeampi ”null-move”-haku syvyyteen n-R
    - Jos suppea haku AB-arvoja huonompi, solmu hylätään
    - Jos arvo on parempi, toteutetaan haku loppuun normaalisti
- Voi vaikuttaa lopputulokseen
- Kuinka suppea ”null-move”-haku on, vaikuttaa tarkkuuteen

# ”Null-move pruning”

- Vertailussa ”AB-version 1”-algoritmi
- Vertaillaan kolmea eri vaihtoehtoa: R={1, 2, 3}

| | Average Nodes | Total Nodes | Percentage of Leaf Nodes | Average Branching Factor | Average Time |
|---|---|---|---|---|---|
| AB Version 1 | $1.089416 \cdot 10^7$ | $2.61459855 \cdot 10^8$ | 88.5215% | 8.88412 | 4.8040 s |
| NM R=1 | $5.369111 \cdot 10^6$ | $1.28858680 \cdot 10^8$ | 90.1071% | 8.83318 | 8.5463 s |
| NM R=2 | $5.561990 \cdot 10^5$ | $1.3348766 \cdot 10^7$ | 75.0400% | 4.46826 | 1.0667 s |
| NM R=3 | $5.277818 \cdot 10^6$ | $1.26667634 \cdot 10^8$ | 88.0728% | 7.95092 | 2.5789 s |

# ”Null-move pruning”

- Vastakkain pelaaminen, koska NM vaikuttaa lopputulokseen

| | Points |
|---|---|
| AB Version 1 | 9.5/18 |
| NM R=1 | 8.5/18 |
| NM R=2 | 9/18 |
| NM R=3 | 9/18 |

# ”Transposition table”

- Jokaisesta uniikista positiosta tallennetaan relevantit tiedot
  - Arvo
  - Syvyys
- Position arvo saadaan taulusta laskematta sitä uudestaan
- Taulukon position syvyys voidaan antaa olla eri kuin solmun syvyys
- Erittäin implementaatio riippuvainen

# ”Transposition table”

- TT-algoritmia verrataan pelkistettyyn minimax-algoritmiin
- TT:stä kolme versiota
  - Taulukon syvyys sama kuin solmun syvyys
  - Taulukon syvyys on pienempi tai yhtä suuri kuin solmun syvyys
  - Taulukon syvyydellä ei merkitystä

| | Average Nodes | Total Nodes | Percentage of Leaf Nodes | Average Branching Factor | Average Time |
|---|---|---|---|---|---|
| Minimax | $1.71608186 \cdot 10^8$ | $4.118596476 \cdot 10^9$ | 96.6764% | 37.4773 | 39.9585 s |
| TT version 1 | $4.8383341 \cdot 10^7$ | $1.1612002 \cdot 10^9$ | 97.2613% | 37.0026 | 54.2508 s |
| TT version 2 | $4.8233201 \cdot 10^7$ | $1.157596829 \cdot 10^9$ | 97.2701% | 37.0026 | 52.6646 s |
| TT version 3 | $3.6350633 \cdot 10^7$ | $8.72415198 \cdot 10^8$ | 97.4408% | 36.9840 | 40.9875 s |

# ”Transposition table”

- TT-versiot muuttavat loppuarvoa, joten niitä peluutetaan vastakkain
  - Versio 1 on identtinen tuloksen kannalta minimax-algoritmiin

| | Points |
|---|---|
| TT version 1 | 7.5/12 |
| TT version 2 | 4/12 |
| TT version 3 | 6.5/12 |

# ”Quiescence search”

- Itsestään päättyvä haku, joka kutsutaan lehtisolmuissa evaluointifunktion tilalla
- Haussa otetaan vain huomioon aktiiviset siirrot
  - Syönnit ja ylennykset
- Kasvattaa hakupuun kokoa

| | Average Nodes | Total Nodes | Quiescence nodes | Percentage of Leaf Nodes | Average Branch-ing Factor | Average Time |
|---|---|---|---|---|---|---|
| NM R=2 | $5.561990 \cdot 10^5$ | $1.3348766 \cdot 10^7$ | 0 | 75.0400% | 4.46826 | 1.0667 s |
| QS primary search depth +0 | $1.6192559 \cdot 10^7$ | $3.886214 \cdot 10^8$ | $3.852213 \cdot 10^8$ | 51.1840% | 2.3312 | 28.5708 s |
| QS primary search depth -1 | $7.471061 \cdot 10^6$ | $1.793054 \cdot 10^8$ | $1.787808 \cdot 10^8$ | 50.1020% | 2.2469 | 11.4 s |

# ”Quiescence search”

- Pääsyy QS:n käyttöön on ”horizon effect”:n minimointi
  - Evaluointifunktio kutsutaan vain epä-aktiivisissa siirroissa

| | Points |
|---|---|
| NM R=2 primary search depth +2 | 2/12 |
| QS primary search depth +0 | 8.5/12 |
| QS primary search depth -1 | 7.5/12 |

# Yhteenveto

- ”AB pruning” vähentää laskentaa tehokkaasti
- ”NM pruning” vähentää myös laskentaa heikentämättä lopputulosta
- ”TT” vähentää laskentaa heikosti
  - Implementaatio hidas, joten ei auttanut nopeuteen
- ”QS” lisäsi laskentaa merkittävästi
  - Paransi lopputulosta
- Loppujen lopuksi algoritmi jossa oli AB, NM ja QS yhdistettynä suoriutui parhaiten

# Viitteet ja aineisto


- Donninger, C., 1993. Null move and deep search. ICCA J., Vol. 16 No. 3 1993, pp. 137-143
- Shannon, C., 1950. Programming a Computer for Playing Chess. Philosophical Magazine, Ser.7, Vol. 41, No. 314, March 1950
- Kaindl, H., Searching to Variable Depth in Computer Chess. 760-762, 8th IJCAI 1983
- Tabibi, O.D., Netanyahu, N.S., Verified Null-Move Pruning, ICGA Journal, Vol. 25, No. 3, pp. 153-161, 2002

# Viitteet ja aineisto


- Beal, D.F., An analysis of minimax
  M.R.B. Clarke (Ed.), Advances in Computer Chess 2, Edinburgh University Press, Edinburgh 1980, pp. 103-109
- Beal, D.F., A generalised quiescene search algorithm, Artificial Intelligence, Vol. 43, Issue 1, April 1990, pp. 85-98
- Schrüfer, G., A Strategic Quiescence Search, ICGA Journal, Vol. 12, No. 1, pp. 3-9, 1989