Aalto-yliopisto
Perustieteiden
korkeakoulu

Improving a minimax search algorithm for chess (aihe-esittely)

Kristian Wasastjerna

21.02.2022

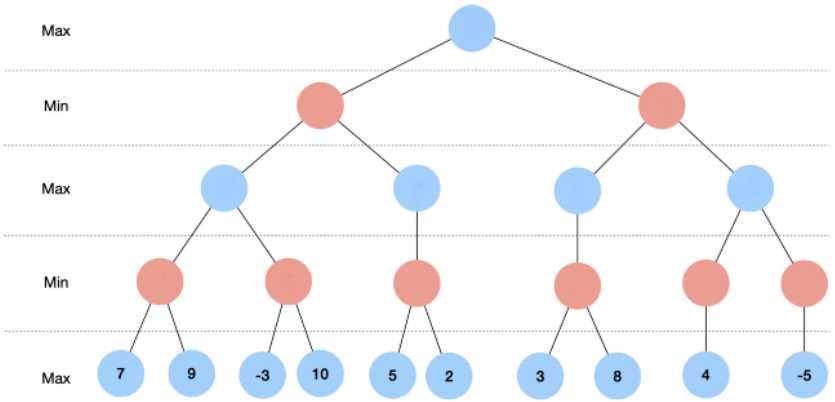
Ohjaaja: Prof. Kai Virtanen

Valvoja: Prof. Kai Virtanen

Systeemianalyysin
laboratorio

Tausta 1/3: Minimax-algoritmi

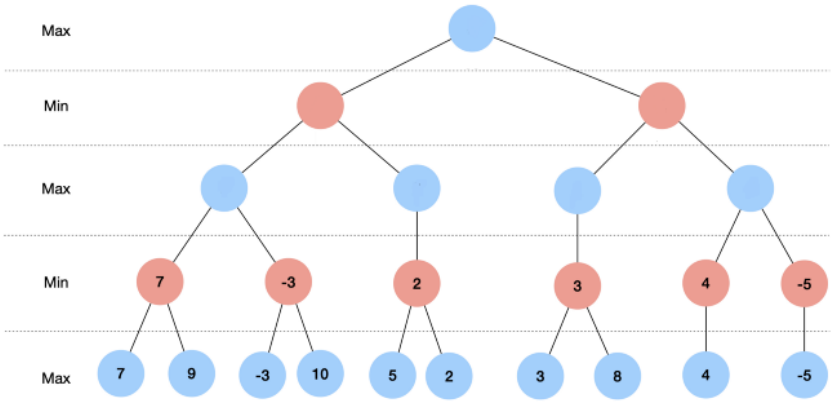
- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Hakupuussa
  - otetaan huomioon kaikki käyvät siirrot kunkin siirtovuoron tilassa
  - ennakoidaan n-siirtovuoroa eteenpäin
- Hyödyn arviointi hakupuun lehtien tiloissa
- Puu ratkaistaan lehdistä juurisolmuun valitsemalla
  - minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto
- Juurisolmun hyöty = Lehden hyöty, johon päädytään, kun pelaajat valitsevat parhaimman siirrot



https://en.wikipedia.org/wiki/File:Minimax.svg

Tausta 1/3: Minimax-algoritmi

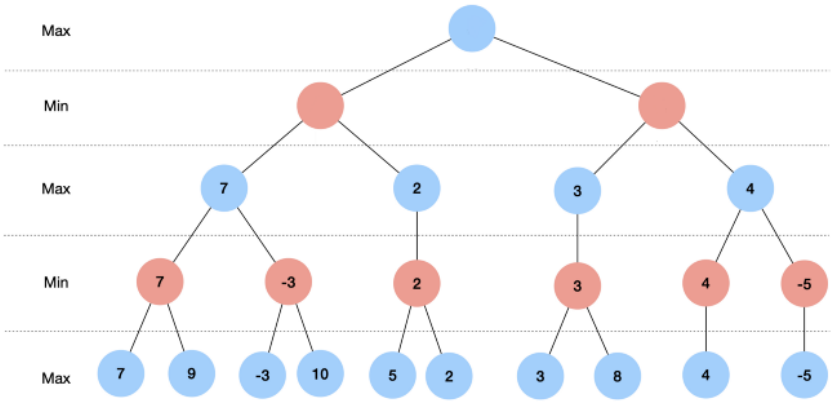
- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Hakupuussa
  - otetaan huomioon kaikki käyvät siirrot kunkin siirtovuoron tilassa
  - ennakoidaan n-siirtovuoroa eteenpäin
- Hyödyn arviointi hakupuun lehtien tiloissa
- Puu ratkaistaan lehdistä juurisolmuun valitsemalla
  - minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto
- Juurisolmun hyöty = Lehden hyöty, johon päädytään, kun pelaajat valitsevat parhaimman siirrot



https://en.wikipedia.org/wiki/File:Minimax.svg

Tausta 1/3: Minimax-algoritmi

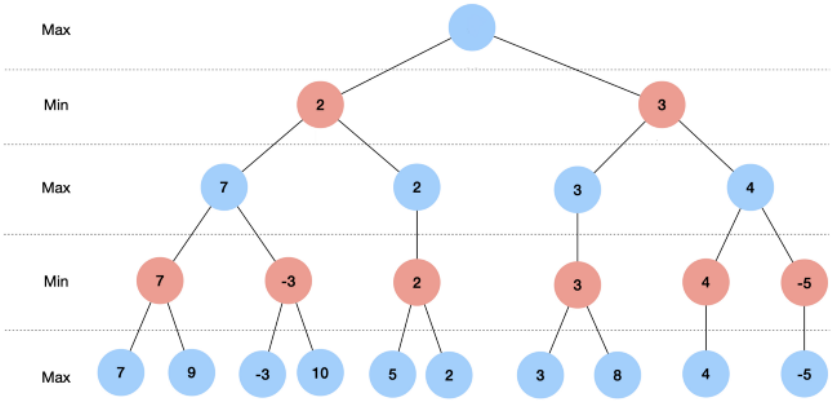
- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Hakupuussa
  - otetaan huomioon kaikki käyvät siirrot kunkin siirtovuoron tilassa
  - ennakoidaan n-siirtovuoroa eteenpäin
- Hyödyn arviointi hakupuun lehtien tiloissa
- Puu ratkaistaan lehdistä juurisolmuun valitsemalla
  - minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto
- Juurisolmun hyöty = Lehden hyöty, johon päädytään, kun pelaajat valitsevat parhaimman siirrot



https://en.wikipedia.org/wiki/File:Minimax.svg

Tausta 1/3: Minimax-algoritmi

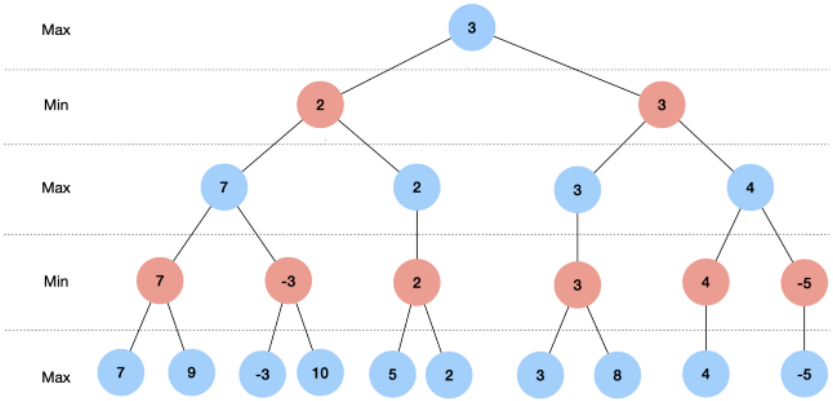
- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Hakupuussa
  - otetaan huomioon kaikki käyvät siirrot kunkin siirtovuoron tilassa
  - ennakoidaan n-siirtovuoroa eteenpäin
- Hyödyn arviointi hakupuun lehtien tiloissa
- Puu ratkaistaan lehdistä juurisolmuun valitsemalla
  - minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto
- Juurisolmun hyöty = Lehden hyöty, johon päädytään, kun pelaajat valitsevat parhaimman siirrot



https://en.wikipedia.org/wiki/File:Minimax.svg

Tausta 1/3: Minimax-algoritmi

- Syvyyssuuntainen algoritmi: Maksimoidaan minimihyöty
- Hakupuussa
  - otetaan huomioon kaikki käyvät siirrot kunkin siirtovuoron tilassa
  - ennakoidaan n-siirtovuoroa eteenpäin
- Hyödyn arviointi hakupuun lehtien tiloissa
- Puu ratkaistaan lehdistä juurisolmuun valitsemalla
  - minimoivan pelaajan kohdalla hyödyn minimoiva siirto
  - maksimoivan pelaajan kohdalla hyödyn maksimoiva siirto
- Juurisolmun hyöty = Lehden hyöty, johon päädytään, kun pelaajat valitsevat parhaimman siirrot



https://en.wikipedia.org/wiki/File:Minimax.svg

Tausta 2/3: Minimax-algoritmin haasteet shakissa

- Shakin pelitilasta on keskimäärin noin 35 mahdollista siirtoa
- Hakupuun koko kasvaa liian suureksi käytössä olevalle laskentateholle puun syvyyden kasvaessa
- ”Horizon effect”-ilmiö
  - Hakupuu syvyyteen n
  - Pelitilan evaluointi ei realistinen, koska n+1 tason siirto voi vaikuttaa merkittävästi hyötyyn

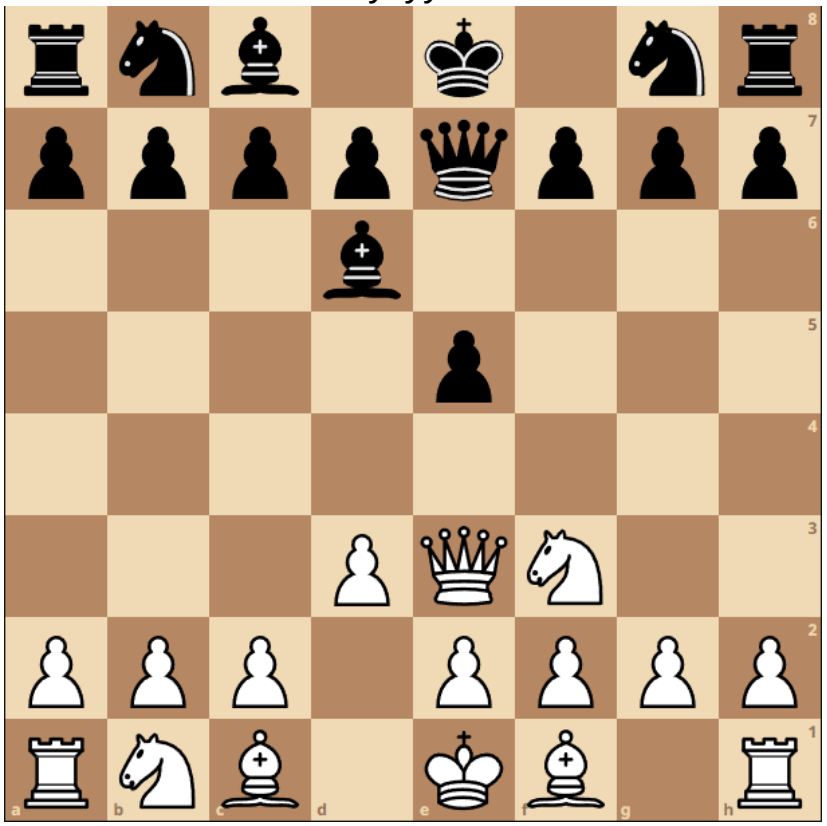
Tausta 3/3: Esimerkkiongelmia

Mahdolliset positiot n-siirron jälkeen aloitus asemasta

0	1
1	20
2	400
3	8,902
4	197,281
5	4,865,609
6	119,060,324
7	3,195,901,860
8	84,998,978,956
9	2,439,530,234,167
10	69,352,859,712,417

https://www.chessprogramming.org/Perft_Results

”Horizon effect”
Minimax-algoritmi syvyyteen 3
Syvyys 0



Tausta 3/3: Esimerkkiongelmia

Mahdolliset positiot n-siirron jälkeen aloitus asemasta

0	1
1	20
2	400
3	8,902
4	197,281
5	4,865,609
6	119,060,324
7	3,195,901,860
8	84,998,978,956
9	2,439,530,234,167
10	69,352,859,712,417

https://www.chessprogramming.org/Perft_Results

”Horizon effect”
Minimax-algoritmi syvyyteen 3
Syvyys 1

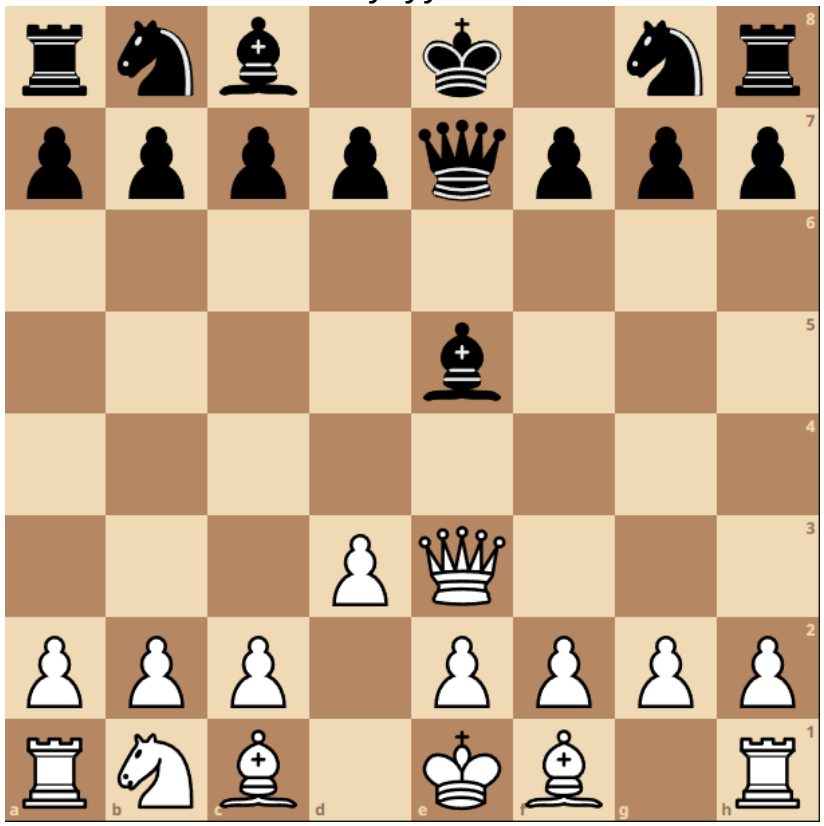
Tausta 3/3: Esimerkkiongelmia

Mahdolliset positiot n-siirron jälkeen aloitus asemasta

0	1
1	20
2	400
3	8,902
4	197,281
5	4,865,609
6	119,060,324
7	3,195,901,860
8	84,998,978,956
9	2,439,530,234,167
10	69,352,859,712,417

https://www.chessprogramming.org/Perft_Results

”Horizon effect”
Minimax-algoritmi syvyyteen 3
Syvyys 2



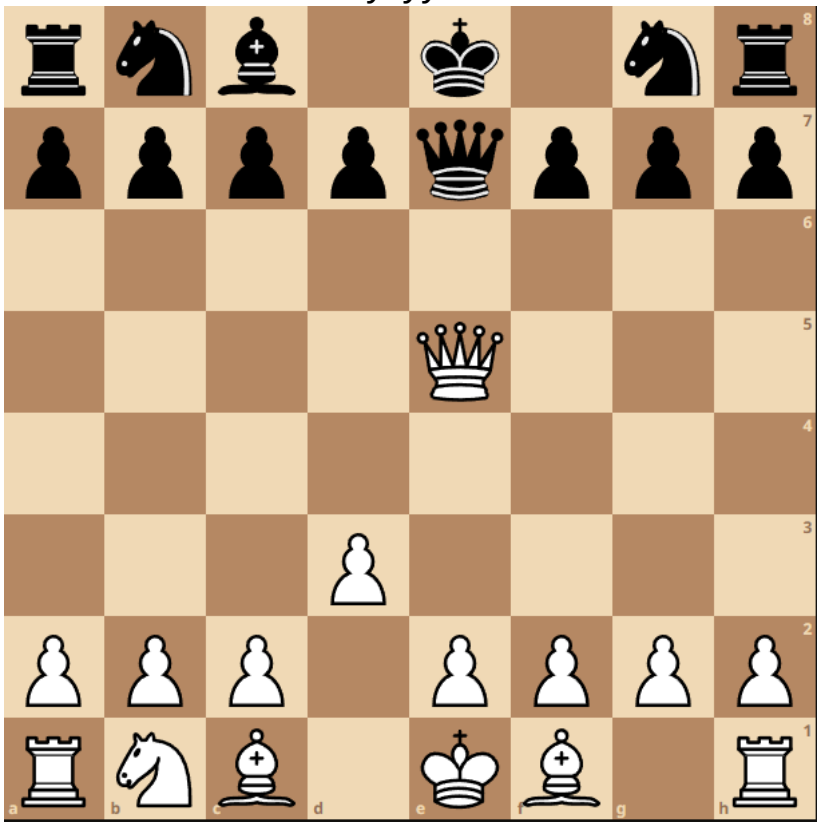
Tausta 3/3: Esimerkkiongelmia

Mahdolliset positiot n-siirron jälkeen aloitus asemasta

0	1
1	20
2	400
3	8,902
4	197,281
5	4,865,609
6	119,060,324
7	3,195,901,860
8	84,998,978,956
9	2,439,530,234,167
10	69,352,859,712,417

https://www.chessprogramming.org/Perft_Results

”Horizon effect”
Minimax-algoritmi syvyyteen 3
Syvyys 3



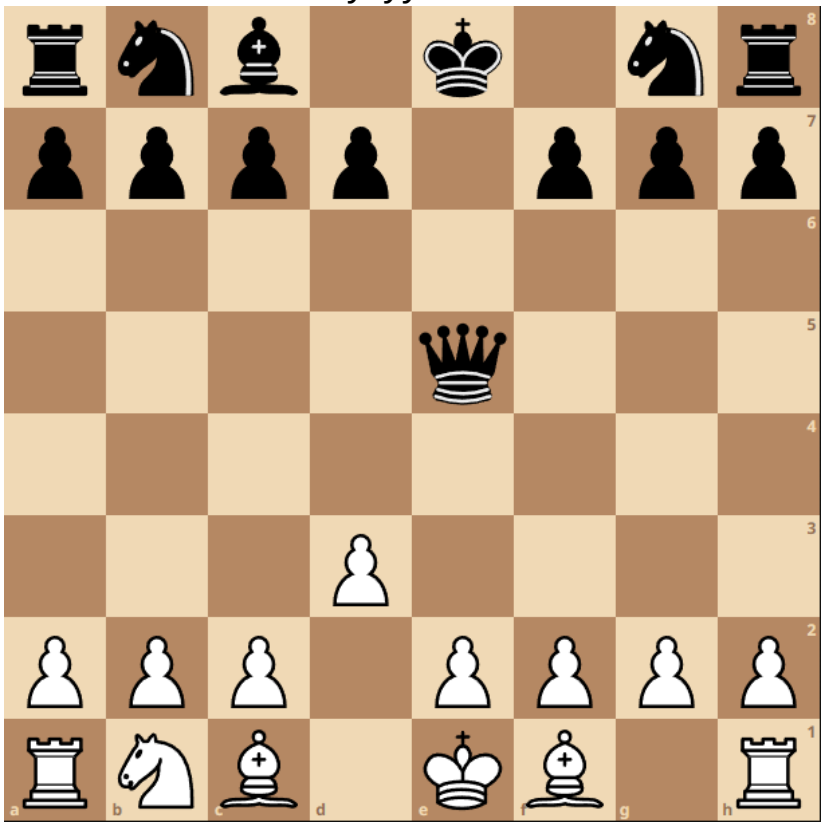
Tausta 3/3: Esimerkkiongelmia

Mahdolliset positiot n-siirron jälkeen aloitus asemasta

0	1
1	20
2	400
3	8,902
4	197,281
5	4,865,609
6	119,060,324
7	3,195,901,860
8	84,998,978,956
9	2,439,530,234,167
10	69,352,859,712,417

https://www.chessprogramming.org/Perft_Results

”Horizon effect”
Minimax-algoritmi syvyyteen 3
Syvyys 3+1



Työn tavoite 1/2

- Vertaillaan hakupuiden tuottamiseen käytettäviä algoritmin vaihtoehtoisia eri versioita
- Minimoidaan hakupuun kokoa
  - Hakupuiden solmujen määrän vertailu
- “Horizon Effect” ilmiön minimointi
  - Siirtojen hyvyyden vertailu
    - Siirron hyvyys arvioidaan ulkoisella työkalulla
  - Mitä parempi hakupuun ehdottama ”paras” siirto, sitä paremmin minimoidaan ”Horizon Effect” ilmiötä

Työn tavoite 2/2

- Implementoidaan vaihtoehtoiset versiot minimax-algoritmiin
- Esimerkkejä tarkasteltavista versioista
  - ”Alpha-Beta Pruning”: Hylätään solmuja ja niistä syntyviä alipuita, jos on löytynyt parempi siirto aikaisemmin
  - ”Null move pruning”: Tuotetaan alaraja siirron ”hyvyydelle” olettamalla, että löytyy vähintään siirto, joka on parempi kuin ”tyhjä” siirto ja hylätään rajan alle jäävät solmut
  - ”Quiescence search”: Kun päästään haluttuun syvyyteen n, jatketaan hakupuun generointia, mutta generoidaan vain tietyt siirrot

Rajaukset

- Tarkastellaan vain minimax-algoritmia ja vertaillaan sen eri versioita shakkipelissä
- Hakupuun nopea generointi ei kuulu työhön
  - Tekniikat hakupuun nopeaan generointiin ja läpikäyntiin sivuutetaan
  - Generointi- ja läpikäyntinopeus raportoidaan kuitenkin työssä
- Toteutetaan pelitilan hyödyn evaluointi ja käypien siirojen generointi – eivät osa työtä

Työkalut

- www.lichess.org - Analysis Board
  - Analysoi shakkipositioita ja tuottaa eräässä mielessä optimaaliset siirrot
  - Käytetään siirtojen hyvyyden evaluoinnissa
- GitHub - Lähdekoodin säilytys
  - Työn koodit ja tulokset
- Visual Studio Code - C++
  - Minimax-algoritmin vaihtoehtoisten toteutuksien implementointi

Aikataulu

- Aineistoon tutustuminen 02/2022
- Aiheen esittely 02/2022
- Työn kirjoittaminen 02-05/2022
- Minimax-algoritmin testaus ja tulokset 02-05/2022
- Tulosten analysointi 04-05/2022
- Valmis työ 31/05/2022

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