Kolmen laudan ”puolueeton” misääri ristinolla

”misère play of 'impartial' tic-tac-toe on 3 boards”

Leo Kiiski
25.1.2016

Selvennetään aluksi hieman otsikkoa. Vastoin kuin normaalissa ristinollassa, jossa toinen pelaaja pelaa risteillä, ja toinen nollilla, ”puolueettomassa” ristinollassa molemmat pelaajat pelaavat risteillä. Misääri (eng. Misère) tarkoittaa pelitapaa, jossa pyritään olemaan voittamatta klassisten pelisääntöjen mukaan ts. pyritään siihen, että vastustaja ajautuu tilanteeseen, jossa hän saa kolmen ristin suoran, jolloin hän häviää. Yhden laudan sijaan pelissä on kuitenkin kolme lautaa, joista vasta viimeinen ratkaisee voittajan. Ensimmäiset kaksi lautaa eliminoituvat pelistä heti, kun niihin tulee kolmen ristin suora, jonka jälkeen niihin ei saa enää laittaa lisää ristejä. Se pelaaja, joka siirrollaan aiheuttaa kolmen ristin suoran viimeiseen jäljellä olevaan lautaan on häviäjä. Tässä **video**, joka selventää hyvin pelin idean.

Työni tarkoitus on toteuttaa tekoäly, joka pelaa kyseistä peliä. Huomioitavaa on, että kyseinen peli on käytännössä matemaattisesti “ratkaistu”, eli sitä täydellisesti pelaava kone tai ihminen voittaa aina, jos kone/ihminen aloittaa, ja pelattavien lautojen määrä on pariton. Vastaavasti parillisella määrällä lautoja toinen pelaaja voittaa aina pelatessaan täydellisen pelin.

Hyvä analyysi pelistä löytyy seuraavasta **tieteellisestä julkaisusta**. Tiivistetty versio koneen tekemästä laskennasta/analyysistä on seuraava:

Analyysi

Isomorfiaa vailla on olemassa 102 tapaa laittaa 9 ristia 3x3 ruudukkoon (*Kuva 2)*. Isomorfia tarkoittaa tässä tapauksessa sitä, että esimerkiksi lauta, jossa on vain yksi risti, joka on vasemmassa yläkulmassa, on isomorfinen kaikkien muiden lautojen kanssa, jossa on vain yksi risti jossain kulmassa. Isomorfiset vertailut voidaan tehdä kiertämällä lautaa 90 asteen askelissa. Esimerkiksi yksi 90 asteen kierros myötäpäivään tuottaa ensin mainitusta laudasta sellaisen laudan, jossa on risti oikeassa yläkulmassa (huom. myöhemmin käytän positioista numeromerkintää 1-9, joka viittaa luvun sijaintiin numeronäppäimistössä. Esimerkiksi positio 7 vastaa vasenta yläkulmaa *Kuva 1*).

Kuva 1: Numeronäppäimistö

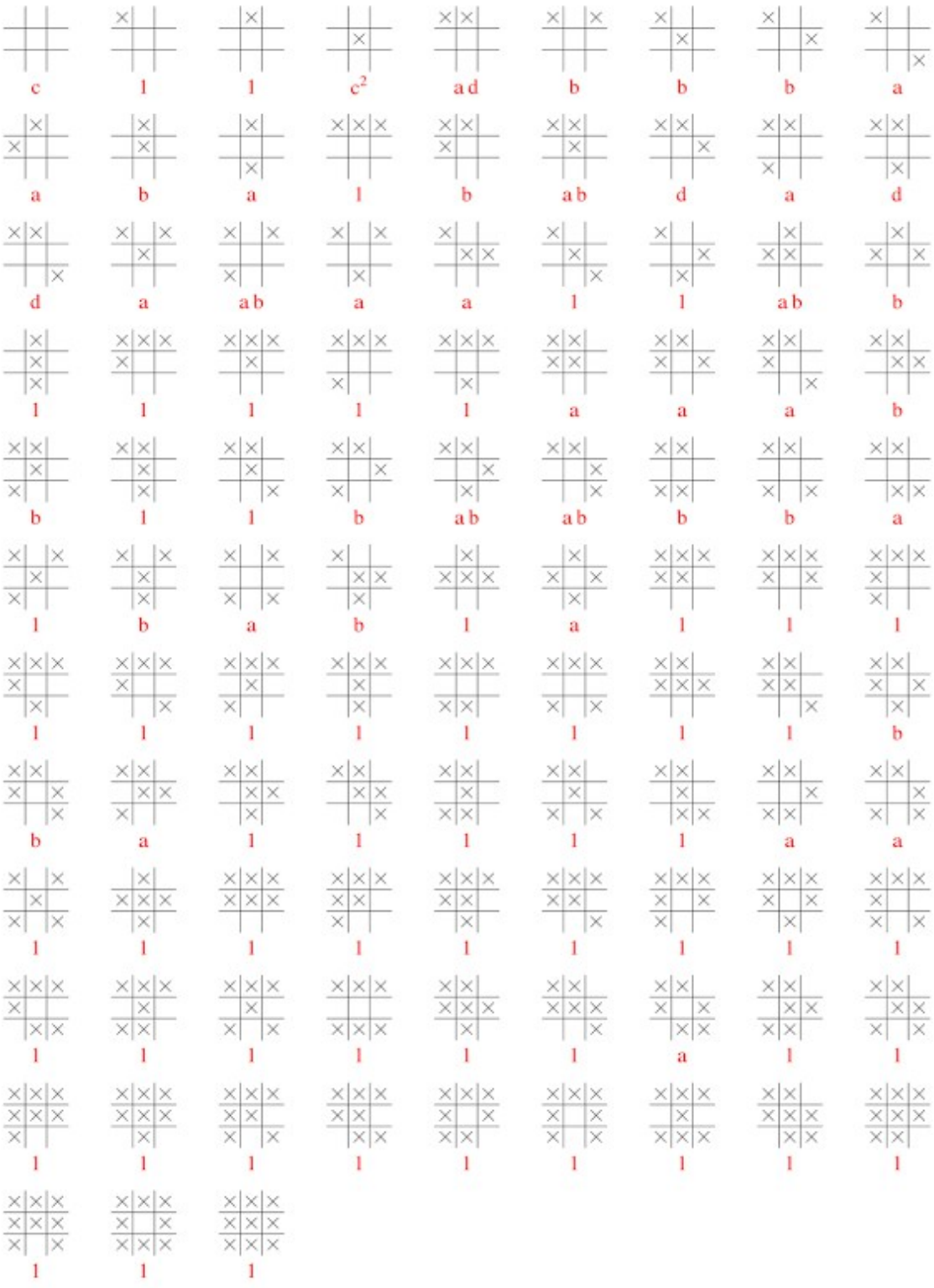

Figure 6: The 102 nonisomorphic ways of arranging zero to nine X's on a tic-tac-toe board, each shown together with its corresponding misere quotient element from Q.

Kuva 2: Tavat järjestää 0-9 ristiä 3x3 ruudukkoon, ja niiden misere osamäärät

Jokaisella kolmesta pelilaudasta on sitä vastaava misere osamäärä, joka näkyy kuvasta 2. Kutsutaan näitä uniikkeja tiloja jatkossa *sormenjäljiksi*. Koko pelin tila lasketaan näiden kolmen laudan misere osamäärien tulona. Esimerkiksi alkutilaa, jossa on kolme tyhjää lautaa vastaa misere osamäärien tulo:

$$ccc = c^3$$

Pelin tilat voidaan jakaa kahteen ryhmään: *N-tilat* ja *P-tilat.* N-tilat ovat sellaisia tiloja, että seuraavaksi pelaava pelaaja voittaa pelatessaan täydellisen pelin. P-tilat ovat sellaisia, että toisena pelaava pelaaja voittaa aina pelatessaan täydellisen pelin (toisella tapaa sanottuna: se, joka siirrollaan saa pelin P-tilaan voittaa, jos ei mokaa). Pelissä on olemassa neljä eri P-tilaa:

$$P = \{a, b^2, bc, c^2\}$$

Kaikki muut tilat ovat N-tiloja. Täydellisesti pelaava pelaaja saa siirrollaan pelin aina johonkin P-tilaan siten, että toisella pelaajalla ei ole muuta vaihtoehtoa kuin saattaa peli N-tilaan. Jos tyhjän pelin alkutila oli ccc, niin aloittava pelaaja voisi esimerkiksi laittaa mihin tahansa lautaan ristin positioon 7, jolloin misere osamäärien tulo olisi:

$$1\mathrm{cc} = c^2$$

Tämä on P-tila, eli kolmella laudalla pelattaessa aloittava pelaaja voittaa aina (pelatessaan täydellisesti). Tässä välissä on hyvä huomauttaa, että misere osamäärien tulot ovat komutatiivisia ts. 1cc = c1c = cc1, bc = cb, jne.

Käytettävät algoritmit, tietorakenteet ja menetelmät

Valmiita algoritmeja tekoälyn toteuttamiseksi on olemassa (esim iPhonella on peli nimeltään *Notakto)*. Aiheesta kirjoitettu julkaisu on kuitenkin melko tuore, joten en ole varma kuinka monta avoimen lähdekoodin versiota algoritmista on, ja kuinka tehokkaita/millä kielillä ne on kirjoitettu. Ajattelin joka tapauksessa lähteä liikkeelle esitetystä matemattisesta analyysistä, ja kehittää peliä pelaava algoritmi itse. Tästä johtuen on melko vaikeaa ennustaa esimerkiksi millaisia erilaisia tietorakenteita tulen tarvitsemaan. Varmana voisin sanoa, että tavallista listaa ja yksittäiselle 3x3 laudalle tehtyä omaa tietorakennetta tulen käyttämään. Eri *sormenjäljet* voidaan jakaa eri tasoihin sen mukaan montako ristiä on laudassa. Esimerkiksi laudasta, jossa on tietty kolmen ristin *sormenjälki* voi päästä vain rajoitetuun joukkoon neljän ristin *sormenjälkiä* – eli neljän ristin *sormenjälkien* osajoukkoon. Tästä tosiasta voi päätellä, että laskennan helpottamiseksi voisi olla järkevää toteuttaa eräänlainen hierarkkinen suunnattu verkko, jossa jokaisen *sormenjäljen* seuraajat ovat jokin seuraavan tason osajoukko. Eri sormenjälkien isomorfismien vertailu vaatii myös oman algoritminsa.

Kyseisen pelin (pienestä) äärellisyydestä, ja olemassa olevasta matemaattisesta analyysistä johtuen se on aikavaativuudeltaan käytännössä vakioaikainen. Tilavaativuus on tässä vaiheessa vaikea määritellä, mutta sekin tulee olemaan melko pieni. Tarkempia arvioita ja ehdotuksia optimoinnista tulen antamaan työn edetessä. Jos pelin logiikka toimisi esim. min-max algoritmilla, niin odottaisin aika- ja tilavaativuuksien olevan suuremmat. En kuitenkaan aio logiikkaa tällä algoritmilla toteuttaa. Ajan puitteissa olisi tietty mielenkiintoista myös toteuttaa logiikka min-max algoritmilla vertailujen suorittamiseksi.

Syötteet

Valittavana on kolme eri pelimoodia: 1) Kone VS Ihminen 2) Ihminen VS Ihminen 3) Kone VS Kone.

Kone vastaan kone pelimoodi on olemassa lähinnä suorituskyvyn analysoimista varten, ja testaamaan, että ensimmäinen pelaaja todella voittaa aina. Muissa pelimoodeissa pelaaja antaa omalla pelivuorollaan aina kaksi syötettä: 1) Laudan johon hän haluaa pelata ristinsä 2) Ristin position. Jos pelataan konetta vastaan, niin kone laskee pelaajan ristin lisäämisen jälkeen kyseiselle laudalle ja koko pelitilanteelle uuden arvon, ja laskee sitten kaikki mahdolliset siirrot P-tilaan. Ajattelin tässä käyttää pientä hallittua epädeterministisyyttä ts. kone valitsee satunnaisesti jonkun P-tilaan vievän siirron, jotta pelit eivät etenisi aina samalla tavalla pelaajan näkökulmasta. Tämä tietenkin johtaa siihen, että kone etsii useamman kuin yhden P-tilaan vievän siirron, vaikkei sillä ole voittamisen kannalta merkitystä, mikä hieman hidastaa koneen nopeutta. Ihmistä vastaan pelatessa tällä nopeuden menetyksellä ei kuitenkaan ole merkitystä, sillä ihmispelaaja on joka tapauksessa konetta moninkertaisesti hitaampi pelaaja.

Lähteet

[1] https://youtu.be/h09XU8t8eUM
[2] https://youtu.be/v494yltz_7I
[3] http://arxiv.org/abs/1301.1672