Eötvös Loránd Tudományegyetem

Természettudományi kar

Játékok és a számítógép

BSc szakdolgozat

Készítette:

Szabó Dávid

Matematika BSc

Alkalmazott matematikus

szakirány

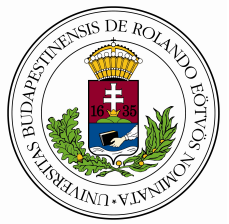
Témavezető:

Szőnyi Tamás

egyetemi tanár

Számítógéptudomány

Tanszék



Budapest

2010

Tartalomjegyzék

1. fejezet

Játékelmélet

1.1. Kezdetek

A játékelmélet a matematika egyik ága. A racionális viselkedés kérdésével foglalkozik olyan helyzetekben, ahol minden résztvevő döntéseinek eredményét befolyásolja a többiek lehetséges választása. Tehát a játékelmélet a stratégiai problémák elmélete. A játékelmélet megteremtését Neumann János és társa 1928-ban kitalált minimax elvéhez köthetjük, majd 1944 – ben kiadták a The Theory of Games and Economic Behavior, azaz a Játékelmélet és gazdasági viselkedés című művet.

1.2. Játékok felosztása

Ahhoz, hogy egy játékot játszani lehessen, a következő paraméterek ismerete nélkülözhetetlen:

- A játékban résztvevő játékosok száma
- A kezdő állás, illetve a játék menete során kialakulható állások
- A szabályos lépések ismerete
- A játék vége, és ezzel egyidejűleg nyertese is
- A játékosok a játék során milyen információkkal rendelkeznek
- Véletlen tényezők közrejátszása a játék menete során

Az előző felsorolás alapján a játékokat a következőképpen kategorizálhatjuk:

1.2.1. Játékosok száma szerint

- egyszemélyes
- kétszemélyes
- három vagy több személyes
- személytelen játékok

1.2.2. Véletlen szerepe szerint

- determinisztikus – A játék lefolyása közben a véletlen nem játszik szerepet
- Sztochasztikus/valószínűségi – A véletlen is szerepet játszik a játék lefolyásában

1.2.3. Információ mennyisége szerint

- A Teljes információjú játékok – Minden játékos rendelkezik az összes információval
- Részleges információjú játékok – Amikor az egyes kimenetelek nem egyértelműek

1.2.4. Végesség szerint

- Véges játék – A játék minden állásában véges sok lépési lehetőség van
- Nem véges játékok – Amikor előfordulhatnak olyan szituáiók, hogy a játék örökké tart

1.2.5. Nyereség és veszteség szerint

- Zérus összegű játékok – A játékosok nyereségének és veszteségének összege nullával egyenlő.

Mindezeket összefoglalva kategorizálhatjuk egy adott játék hovatartozását. Példaként teljes információjú determinisztikus játék a sakk vagy a Button madness. Ahol az előbbi kétszemélyes, az utóbbi pedig egyszemélyes játék. Teljes információjú, ámde valószínűségi

játékok közé sorolhatjuk a társasjátékok nagy részét mint a Ki nevet a végén. Részleges információjú determinisztikus játék az aknakereső. Illetve ugyanúgy részleges információjú, de valószínűségi játékok a kártyajátékok nagy része, mint a póker vagy a bridzs.

2. fejezet

A Játékok elemzése a játékosok számának függvényében

2.1. Személytelen játékok

Bár ez egy eléggé meglepő kategória, de ezen típusú játékok is léteznek, vagy csak egyszemélyes játékoknak tituláljuk őket. A játék lefolyásában a játékosnak nincs valódi szerepe, maximum csak megfigyelőként vesz részt a játékban, követi a játék menetét. A személytelenség azon alapszik, hogy mind a kezdő lépés, mind a további lépések egyértelműen adottak. A játékszabály pontosan megadja mit kell tenni a következő lépésben. Például a szeret – nem szeret játék, aminek játéktere nem más, mint egy virág összes szirma. Játékszabálya pedig abból áll, hogy a szirmokat tépkedve egymás után a páratlan sorszámú letépett szirmok egy „szeret” illetve a páros sorszámú letépett szirmok egy „nem szeret” jelzőt kapnak. A játék akkor ér véget ha szirmaink elfogynak. A játékos nyertesnek érezheti magát, ha az utolsó letépett szirom a „szeret” jelzőt kapta. De valójában, mint ezt láthatjuk, nem a játékosunkon múlott a kimenetel, hanem a virág szirmainak számosságán. Így játékosunk csak megfigyelte a folyamatot.

2.2. Három vagy több személyes játékok

Álltalában, ha egy adott játékot kettőnél többen játszák, akkot többszemélyes játékról beszélünk. A játék során minden játékos arra törekszik, hogy a játék végeztével minél

sikeresebb eredménnyel zárjon. A cél érdekében ezért gyakori jelenség, hogy egy vagy több játékos koalícióra lépve ezt biztosíthatja magának. Ezáltal a többszemélyes játékok egy részét máris kétszemélyessé redukáltuk. Példaként hozhatjuk fel a vízilabdát, ahol 14 játékos mérkőzik meg és ebből 7-7en koalícióra lépve, máris két csapat meccsét láthatjuk. Ezt pedig ha a csapatokat nézzük tekinthetjük kétszemélyes játéknak.

2.3. Egy – és kétszemélyes játékok

A továbbiakban legyen szó most csak a determinisztikus, teljes információjú, véges és zérusösszegű játékokról. Azaz a játék menetéről csak a játékosok döntsenek és ne egy véletlen generátoron, például egy dobókockán múljanak az egyes kimenetelek. A játékosok az összes kimenetelről legyenek informálva, illetve kétszemélyes játéknál az egyik fél győzelme a másik veszteségét jelentse. Nevezzük azokat a játékokat melyek ezen paramétereknek megfelelnek *kombinatorikai játékoknak*. Amennyiben egy kombinatorikai játéknak nincs kijelölt kezdő állása, akkor ezt a játékot indefinitnek nevezzük. Ellenben, ha egy indefinit kombinatorikai játékot kezdőpozícióval látunk el, ebben az esetben már definit kombinatorikai játékról beszélhetünk. Ha adott egy kombinatorikus játékunk ami szimmetrikus és végesfokú akkor ezt a játékot röviden egyszerű játéknak nevezhetjük. Ahol a végesfokú tuladonság annyit jelenet, hogy egy adott állásból, csak véges számú különböző lépést tehetünk. Illetve egy játék szimmetriája alatt azt értjük, hogy a játékosok ugyanazon lépéseket tehetik meg. Ebből kifolyólag például a sakk a-szimmetrikus játék, mivel egyik játékos sem léphet a másik bábujával.

2.3.1. Implementálás gráffal

Amennyiben adva van egy az előzőekben definiált kombinatorikai játékunk, akkor minden esetben készíthetünk egy hozzá tartozó gráfot . Ehhez meg kell keresni játékunk azon tulajdonságait, amelyek fontosak a játék lefolyásában. Ilyen tulajdonságok lehetnek, például szín, méret, pozíció vagy bármi más. Ha ezen tulajdonságokat elnevezzük valamilyen változóval, akkor ezen változókból alkotott vektor fogja jellemezni egy állapotunkat a játékunkban. Ha az egyik vektorváltozónkat megváltoztatjuk, akkor egy teljesen más állapotot kapunk. Ahhoz, hogy ezt elérjük, operátorokra van szükségünk, melyek maguk-

ban hordozzák a játék minden tulajdonságát, azaz a szabályos lépések összességét. Az egyes állapotokban tehát minden paraméternek egy-egy értéke van. Az állapottér nem más mint az összes lehetséges vektorváltozónk halmaza. A gráfunk csúcsai nem lesznek mások, mint maguk az állapotok. Ezek után az egyik csúcsból akkor és csak akkor irányítunk élt egy másik csúcsba ha a játékunk szabályai azt megengedik, vagyis egy állapotból csak megengedett lépéssel juthatunk el egy másik állapotba. Ezen felül még meg kell adnunk egy kezdő csúcsot, vagyis azt az állapotot, ahonnét a játékunk indul. Amennyiben pedig egy olyan pontra lépünk rá a gráfban, amiből nem vezet ki irányított él, abban az esetben egy végállapotba értünk. Mivel nullösszegű játékokról van szó, ezért ebben az állapotban megszületik játékunk nyertese és vesztese is. Ha precízek akarnánk lenne, akkor egy játék gráffal való reprezentálását az (A, O, k, C) négyes segítségével definiálhatunk, ahol:

- A – a játékunk állapottere, ahol az egyes állapotok egy – egy csúcsban kódoltak
- O – operátoraink halmaza melyek megeggyeznek a gráfban lévő éleinkkel
- k – a kezdőállapot vagy kezdőállapotaink halmaza ($k \in A$), vagyis az a csúcs ahonnét játékunk indul
- C – célállapot vagy célállapotok halmaza ($C \in A$), gráfunk azon csúcsai, melyek nyelők lesznek

A játékok gráffal való reprezentálása természetesen annál bonyolultabb, minél bonyolultabb játékot játszunk. Példaként a sakk összes lehetséges állapota és állapotterének összeállítása elég komoly problémát okozna. Ha gráfunk készen áll, akkor két játékos esetén a kezdőállapotból indulva, egymás után felváltva lépkednek a gráfunk csúcsain. A valóságban persze esetleg mit sem sejtve erről. Továbbá mindketten arra törekednek, hogy előbb érjék el a célállapotok valamelyikét. Amennyiben pedig „egyszemélyes” játékról lévén szó a végállapot mihamarabbi elérése a cél. Előfordulhat, hogy ezen implementálás során körök keletkeznek gráfunkban. Ez azonban azt eredményezné, hogy játékunk nem lenne véges, vagyis játékosunk vagy játékosaink ezen kör csúcsain lépkedve sosem fejeznék be a játékot. Ezt az esetet viszont kizártuk, mivel játékunkat végesnek definiáltuk.

3. fejezet

Kétszemélyes játékok nyerő stratégiája

Legyen adott egy kétszemélyes játék. Játékosainkat jelöljük meg A, illetve B névvel. Mindketten válasszanak egy tetszőleges stratégiát. Ha egy olyan játékról beszélünk, ami nem kombinatorikus játék, egyértelmű, hogy ebben az esetben sem A-nak sem pedig B játékosnak nincsen nyerő stratégiája. Amennyiben az adott játékunk kombinatorikai és kritériumai megfelelnek az előző fejezetben megfogalmazott játékkal, ebben az esetben Neumann János tétele alapján a kétszemélyes definit kombinatorikai játékokban vagy az egyik játékosnak van nyerő stratégiája, vagy mindkét játékosnak van biztonságos stratégiája. Ezek közül valamelyik biztosan teljesülni fog.

3.1. Grundy – számozás

Sprague és Grundy a harmincas években egymástól függetlenül észrevették, hogy egy egyszerű játék magját a következő módon lehet definiálni: Keresni kell egy γ függvényt a játék állásainak a halmazán, amely egy tetszőleges olyan álláshoz, amelyből pontosan a b_1,b_2...b_k állásokba lehet eljutni, azt a legkisebb nem negatív egészet rendeli, amely különbözik a $\gamma(b_1)$,$\gamma(b_2)$...$\gamma(b_k)$ függvényértékek mindegyikétől. Ez a γ függvény kiterjeszthető halmazokra is, vagyis egy még Grundy – szám nélküli pozíció értékét megkapjuk úgy, hogy nézzük a rákövetkező pozíciók Grundy – számainak halmazát és ezen hajtjuk végre az előző hozzárendelést.

Például: $\gamma(0) = 1; \gamma(1) = 0; \gamma(0,1,3) = \gamma(0,1) = \gamma(0,1,3,7,8,9) = 2$

A játék magja pedig ezen γ függvény zérus helyeinek halmaza lesz, azaz ahol $\gamma(a) = 0$. Ebből következik, hogy amennyiben egy $\gamma(a) = 0$ állásban vagyunk, akkor innét egy b állásba lépve $\gamma(b) \neq 0$ állásba juthatunk csak, illetve ha „a" végállás akkor a játék szabályai szerint nem léphetünk sehová, hiszen a játék véget ért.

3.1. Definíció . *[]$\gamma(p) = mex\{\gamma(q) \mid q \in P, (p,q) \in L\}$ összefüggés adja meg egy játék Sprague-Grundy függvényét, ahol P a játékunk állásainak halmaza, $L = (L1 \cup L2)$ reláció a P halmazaon, vagyis az első és második játékosunk lépéseinek halmaza. A „mex" pedig a legkisebb kizárt angol rövidítése. (minimal excluded)*

3.2. Tétel. *[] Minden egyszerű játéknak létezik egyetlen Sprague-Grundy függvénye.*

Bizonyítás. Először azt kell bizonyítanunk, hogy egy egyszerű játék minden egyes p állásához van egy olyan szám, nevezzük n-el ($n \in N_0$ ahol N_0 a nyerő állás, vagy esetleg nyerőállások halmaza), hogy minden játszmának amelyiknek az előbb említett p a kezdőállása n lépésből végigjátszható. Indirekten feltesszük, hogy ez nem így van, vagyis létezik egy olyan egyszerű játék, aminek van egy olyan p_0 kezdőállása, amelyből kiindulva akármilyen hosszú játszma lejátszható. Ez azt jelenti, hogy végtelen sok különböző olyan játszma van, aminek ez a p_0 lesz a kezdőállása. Mivel az egyszerű játékunk végesfokú, ezért e játszmák között végtelen sok különböző olyan játszma van, ami ugyanazzal a (p_0, p_1) lépéssel kezdődik. Ezt a gondolatmenetet folytatva van végtelen sok különböző olyan játszma, amely $p_0p_1p_2$ állásokkal kezdődik. Ezt hasonlóan folytatva minden n számhoz találunk olyan p_n állást, hogy végtelen sok játszma a $p_0p_1p_2...p_n$ állássorozattal kezdődik. Ekkor viszont a (p_{n-1}, p_n) eleme lesz A vagy B játékos lépés halmazának, ezért az egyszerű játékunkban létezik $p_0p_1...p_n...$ játszma, ami lehetetlen hiszen az egyszerű játék véges fokú. (Ez tulajdonképpen nem más mint a König Lemma, vagyis azt láttuk be ha egy véges fokú gráf valamelyik pontjából végtelen sok különböző út indul ki, akkor abból a pontból végtelen hosszú út is indul.) Az $o(p)$ számot nevezzük a p állás rendjének, ahol $o(p)$ a leghosszabb játék hossza p állásból számolva. Definiálunk P-n egy γ függvényt a következő módon: Ha $o(p) = 0$ akkor p egy végállás, azaz $\{q \in P \mid (p,q) \in L\}$ üres, hiszen egy végállásból nem tudunk tovább lépni. Legyen $\gamma(p) = 0$. Ekkor $\gamma(p)$ – re teljesül 3.1. Tegyük fel, hogy $\gamma(q)$ – t minden olyan q eleme P-re , amelyeknek a rendje kisebb, mint n, már úgy definiáltuk, hogy teljesül

rá 3.1. Legyen $o(p) = n$, és legyenek q_1, q_2,..., q_k az összes olyan állapotok, melyekre (p, q_i) eleme L $(i = 1, ..., k)$. A rend értelmezéséből következik, hogy minden q_i rendje kisebb mint n. $\gamma(q_1)$,..., $\gamma(q_k)$ függvényértékeket már definiáltuk, és azokra is teljesül 3.1. Legyen $\gamma(p) = mex\{\gamma(q_i) \mid i = 1, ..., k\}$. Ekkor $\gamma(p)$ – re is teljesül 3.1. Teljes indukció mutatja ezekből, hogy minden n – re ami eleme az N_0-nak igaz lesz, hogy γ értéke az összes n rendű állásokra, úgy definiálható, hogy teljesüljön 3.1. Mivel minden állásnak van rendje, $\gamma(p)$ minden $p \in P$ állásra definiálható úgy, hogy szintén teljesüljön 3.1. Ezzel az egyszerű játék Sprague – Grundy függvényének létezését beláttuk. Ha lenne olyan játék, aminek egyszerre két különböző Sprague – Grundy függvénye is lenne, akkor ezen játéknak lenne legkisebb rendű olyan állapota, amelyen a két függvény különböző. Ez azonban ellentmond annak, hogy 3.1 szerint a Sprague – Grundy függvény értékét minden p állásban egyértelműen meghatározzák e függvény a p rendjénél kisebb rendű állásokon felvett értékei. Így az egyszerű játékoknak csak egyetlen Sprague – Grundy függvényük van. □

3.2. Király játék

Nézzük meg az előzőeket egy konkrét példán. A választott játék legyen Király című játék. A játék egy sakktáblán játszódik és egy bábu (király) van elhelyezve a jobb alsó sarokban. A játékosok felváltva léphetnek, mégpedig a sakkbéli király lépéseivel megegyezően. Viszont visszafelé nem haladhatnak. Tehát csak az egyet balra, egyet felfelé elv érvényesül, vagy az átlósan lépések közül választhatnak a játékosok. Ha precízen akarnánk leírni az előzőek alapján, akkor Király = (A, O, k, C), ahol az „A” jelen esetben a sakktábla egyes négyzeteinek halmaza. Az „O” nem más, mint az az operátor, ami megváltoztatja az épp aktuális állapot koordinátáit. Azaz, ha az A_{ij} állapotban vagyunk, akkor ezt megváltoztatva kerülhetünk az $A_{i+1,j}A_{i,j+1}$ illetve az $A_{i+1,j+1}$ állapotba. A „k” kezdőállapot nem lesz más, mint az A_{11} es indexű állapot – azaz a sakktábla jobb alsó négyzete. A „C” végállapotok halmaza. Jelen esetben csak egy ilyen van, mégpedig bal felső négyzet. A játék egyes állásaihoz, vagy másképpen a gráf egyes pontjaihoz hozzárendeljük a Grundy számokat a következő szabály szerint:

Ahhoz a pozícióhoz, amelyiknek már nincs rákövetkezője, azaz az egyik játékos veszíteni fog, 0 – t írunk – ez lesz a végállapot. Ha egy A_{ij} pozíciónak már minden rákövetkezőjéhez írtunk számot, akkor A_{ij} pozícióhoz ezek legkisebb közös kizártját rendeljük.(azaz a „γ" függvényértékünk.)

0	1	0	1	0	1	0	1
1	2	3	2	3	2	3	2
0	3	0	1	0	1	0	1
1	2	1	2	3	2	3	2
0	3	0	3	0	1	0	1
1	2	1	2	1	2	3	2
0	3	0	3	0	3	0	1
1	2	1	2	1	2	1	2

3.1. ábra. Király játék Grundy – táblája

3.3. Tétel. *[] Egy véges játék minden pozíciójának van Grundy száma.*

Bizonyítás. Tegyük fel ennek a tételnek az ellentettjét, azaz a hozzárendelési szabályok szerint már egyetlen játékbeli állapothoz sem tudunk Grundy számot rendelni. Ez úgy következzen be, hogy még van illetve vannak olyan pozíciók melyekhez még nem rendeltünk számot. Nevezzük a még kitöltetlen pozíciót X_1 – nek. Ez csak akkor lehetséges ha X_1 – nek van olyan X_2 rákövetkezője, amihez még úgyszintén nem rendeltünk számot. Ha nem lenne rákövetkezője, akkor a 0 Grundy számot kapná, ha pedig minden rákövetkezője kitöltött lenne, ebben az esetben rákövetkezői legkisebb közös kizártjának értékét kapná. Ezt a gondolatmenetet folytatva X_2 – nek is kell lennie olyan rákövetkezőjének mely még nem kapott Grundy számot, és így tovább. Ezáltal kapunk egy X_1, X_2, X_3... végtelen hosszú pozíció sorozatot ahol X_{i+1} rákövetkezője az X_i pozíciónak. A játékunkban azonban csak véges sok pozíció van, ezért ebben a sorozatban valamelyik pozíciónak ismét elő kell fordulnia. Legyen például az X_k pozíció ami megegyezzik X_1-el, ami azt jelenti, hogy gráfunkban egy kör keletkezett, azaz a játék nem véges. Ami ellentmond feltevésünknek. □

3.3. A Grundy – számok értelmezése

Ahogy az a 3.1 – es ábrából is kiolvasható a Grundy – számokról:

- A vesztes pozíció Grundy száma = 0,
- Ha egy pozíció Grundy száma $n \geq 0$, akkor egyetlen rákövetkező pozíció sem lehet n,
- a egy pozíció Grundy száma n > 0, akkor a rákövetkező pozíciók között biztosan szerepelni fog $n-1$, $n-2$,... $1, 0$ Grundy számú pozíciók is.

Ezen felsorolásból rögtön adódik:

3.4. Tétel. *[] Egy véges játékban Kezdőnek akkor, és csak akkor van nyerő stratégiája, ha a kezdő pozíció Grundy – száma 0-tól különbözik.*

Bizonyítás. Ha ugyanis a kezdő pozíció Grundy – száma nullától különböző, akkor a harmadik tulajdonság miatt van 0 Grundy – számú rákövetkezője. Ebben az esetben a kezdő játékos választhatja a 0-s pozíciót. A második játékosnak ekkor természetesen a nulla pozícióból kell ellépnie. Azaz a második játékos vagy veszít, ezt tükrözi az első tulajdonság, vagy kénytelen lesz olyan pozícióba lépni aminek a Grundy – száma nullától különböző (ezt a második tulajdonság adja). Ennek következményeképpen a kezdő játékos ismételten nem nulla pozícióból léphet!

Ha a kezdő játékos ezt a stratégiát követi, akkor mindig pozitív, nem nulla pozícióból léphet, aminek a harmadik pont szerint mindig van 0 rákövetkezője. Ebben az esetben a kezdő játékosunk nem veszítheti el a „meccset”, s mivel a játék véges ebből adódóan a játék egyszer véget ér.
A folyamat megfordítva is igaz lesz. Ha a kezdő játékosnak kell a 0 Grundy számú pozícióból lépnie, azaz csak n > 0 értékre léphet. Ebben az esetben a Második játékosnak van lehetősége 0 – s pozíciót választania, azaz övé a nyerő stratégia.

□

Ezen ismeretek tudatában bármelyik időpillanatban (azaz a játék bármelyik állásban) meg tudjuk mondani, hogy melyik játékosnak van nyerő stratégiája. Azonban ha a nyerő stratégiát használó játékos egyszer is elvéti lépését, ezzel átadja ellenfelének a nyerés esélyét.

Az előző tétel bizonyítása a Grundy – számoknak szintén ezen alfejezet kezdetén felsorolt három tulajdonságán múlott, valamint azon a tényen, hogy a játék előbb, vagy utóbb véget ér. A Grundy – számozást és ezzel együtt az 3.4 – es tételt néhány nem véges játékra is lehet általánosítani a következő módon. (3.2 ábra)

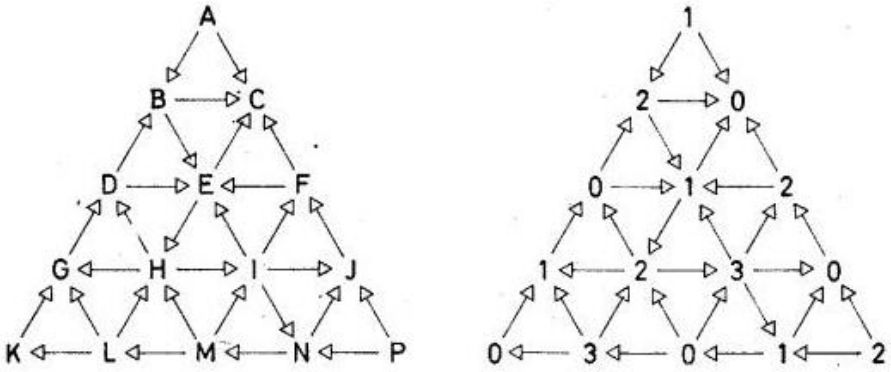
3.4. Általánosítás nem véges játékokra

Ezidáig véges játékokról volt szó, de a tétel általánosítható nem véges játékokra is. Azaz, ha például a játékterünkön szerepelnek körök. Ebben az esetben a számozást kicsit módosítanunk kell. Mégpedig úgy, hogy megkeressük azt a pozíciót amelyiknek nincs rákövetkezője. Ezen pozíció Grundy – számát egyenlővé nullának definiáljuk. A továbbiakban egy még Grundy – szám nélküli X pozícióhoz abban az esetben rendelünk hozzá egy $n \geq 0$ számot,

- Ha n a legkisebb kizártja azoknak a számoknak, amelyek X megszámozott rákövetkezőihez tartoznak,
- Ha X minden szám nélküli rákövetkezőjének van olyan megszámozott rákövetkezője, melynek Grundy – száma éppen n. (3.2 ábra)

Ezek után amennyiben felmerülne az az eset, hogy maradnak Grundy – szám nélküli pozíciók, ebben az esetben a szokásos módosítatlan eljárás szerint kitölthetjük azokat. Ezt az eljárást illusztráljuk a *Dugó* játékon. (3.2 ábra)

Tehát először meg kell keresnünk azokat a pozíciókat melyeknek nincs rákövetkezőjük, vagyis nem vezet ki belőlük él. Csak egy ilyen csúcsunk van a C. Ezután a D csúcs kaphat még 0 Grudy számot, mivel rákövetkezői még számozatlanok (B,E), és mindkettőjüknek van 0 rákövetkezője. Ezek után G és K csúcsunk egyértelműen kitölthetők, hiszen G – nek csak D a rákövetkezője, így G az 1 Grundy számot kapja, és mivel K – nak csak a G a rákövetlezője így az a 0 számot kapja. Most csak az E csúcsot számozhatjuk meg,



3.2. ábra. Dugó játék

ami a szabályok szerint az 1 Grundy – számot kapja. Ezután az E csúcs rákövetkezője, a kitöltetlen H csúcsból, ahonnét eljuthatunk az 1 Grundy – számú G – be és a 0 Grundy számú D – be. így a H csúcs a 2 Grudy számot kapja. Ezek után a B csúcs már könnyen kitölthető, hiszen rákövetkezői a 0 és az 1 számozású csúcsok. Azaz B értéke 2 lesz, és hasonló módon az A csúcsunk megkapja az 1 Grundy számot. Ezek után ha a normál kitöltési szabály szerint haladunk végig az F,J,I,H,L,M,N,P pontokon, nem ütközünk akadályba, mivel mikor elérjük a következő pontot a kitöltési sorredben, akkor már minden rákövetkező Grundy – számuk adott lesz.

A nem véges játékokra a is hasonlóan működnek az előző alfejezetben leírt nyerési és vesztési pozíciók. Azaz a kezdő játékosnak pontosan akkor van nyerő stratégiája, ha a kezdő pozíció valamelyik rákövetkezője nulla Grundy számmal ékeskedik, ill. a második játékosnak akkor, ha a kezdő pozíció kapja a 0 számot. Minden más esetben ha mindkét játékos nyerő stratégiára törekszik, akkor a játék örökké tart. Ebből fakadóan sem nyertesünk sem vesztesünk nem születik.

3.5. Játékok összege

Tekintsük a Halom játékot, melynek kezdetén legyen egy halom kavicsunk. Két játékosunk felváltva elvesznek belőle tetszőleges számú kavicsot, de legalább egyet. Ez a játék így önmagában nem érdekes, hiszen a kezdő játékos az összes kavics elvételével

megfosztja ellenfelét a lépésétől, így az első nyer. Ám ha néhány szabállyal módosítjuk, akkor már érdekesebbé tehetjük.

3.5.1. Bachet játéka

Legyen kezdetben egy kupacunk, mely 100 kavicsból áll. 2 játékos felváltva vesz el 1 – 10 közötti értéknek megfelelő kavicsot. Grundy számaink segítségével itt is könnyen meghatározhatjuk a nyerő állásainkat. A 0 elemű halomnak, mivel nincs rákövetkezője, Grundy száma 0 lesz. Ezek után nézzük az 1, 2,... 10 elemű kupacokat. Az 1 elemű kupacnak egyetlen rákövetkezője van a 0 elemszámú kupac, így Grundy száma 1 lesz. Ez így megy egészen 10-ig. A 11 elemű kupacnak már csak az 1, 2, ... 10 elemszámú kupacok lehetnek a rákövetkezői, ezért a Grundy függvény szabályai szerint a 11 elemű kupac megkapja a 0 Grundy számot. Tehát az n elemű kupacunk Grundy száma n moduló 11 lesz. Ebből könnyen látszik, hogy a nyerő pozíciók, melyek a 0 Grundy számú kupacok nem mások, mint a 11 és az egész számú többszörösei 100 – ig.

Ezt a játékot megfogalmazhatjuk álltalánosan is. A kezdetben 100 helyett most legyen „n”. Játékosaink az adott n elemű kupacból k darab kavicsot vehetnek el, ahol $1 \leq k \leq n$. Az konkrét játék meggondolása alapján egy m kavicsból álló kupac (ahol $m \leq n$) Grundy száma m modulo $(k+1)$ lesz, és nyerő pozíciói pedig a $(k+1)$ egész számú többszöröse n – ig.

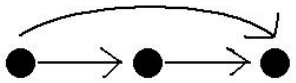
3.5.2. Halmok összege

Most nézzük azt a játékot, amikor a Halom játék nem egy, hanem kettő, vagy több kupaccal kezdődik. Kezdő állásként megadjuk a kupacok elemszámát. Ezek után a játékosok felválltva lépnek, mégpedig úgy, hogy tetszőlegesen választanak egy kupacot, majd arra a kupacra felhasználják a Halom játék játékszabályait, azaz tetszőleges 1 és a kupac elemszáma közötti kavicsot elvesznek belőle. A nyertes pedig az lesz, aki az utolsó játszma nyertese lesz.
Ezt nevezzük két, vagy több halom játék összegének. Ezek alapján a következőt definiál-

hatjuk.:

3.5. Definíció . *[] Kettő, vagy annál több játék összegén azt értjük, hogy minden játékban megadunk egy – egy kezdőállást, ezek után pedig mindkét játékos egymást váltogatva a játék szabályainak megfelelően az általa választott játékban lép egyet. A nyertes pedig az lesz, aki a második, illetve az utolsó játszma nyertese lesz.*

Jelöle J_1 és J_2 a játékainkat. Összegük gráffal való magvalósítása nem lesz más mint két játékunk Descartes szorzata, azaz $J_1 \times J_2$. Né zzük meg például két 2 kavicsból álló halom játék összegét. J_1 játékunk állapotai legyenek (p_0, p_1, p_2), illetve J_2 játékunké pedig (q_0, q_1, q_2). Ahol a 0. indexü pozíciók jelentsék azt, hogy a kupacunkban 0 kavics van, az 1 és 2 – es indexűek pedig azt, hogy kupacunkban rendre 1 illetve 2 kavics van. Ha külön külön szeretnénk őket ábrázolni, akkor ezen állapotok lennének gráfjaink csúcsai, az éleket pedig a szabályok szerint az ábrán látható módon húzhatjuk be. (3.3 -bra)

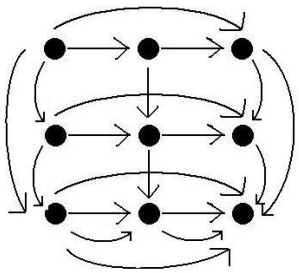


3.3. ábra. Halom játék 2 kavicsra

Nézzük most e két gráf Descartes szorzatát, ami azt jelenti, hogy J_1 minden pontjához hozzárendeljük J_2 minden pontját. Így kapunk ehhez a konkrét játékhoz 9 állapotot, melyeket szintén tekinthetünk 9 gráfbeli csúcsnak. Legyenek ezen állapotok / csúcsok indexei $p_0q_0, p_0q_1, ..., p_2q_2$. Két pont között pedig irányítsunk élt akkor, ha az egyik játékbeli koordinátát rögzítjük (példaként csak azt a sort nézzük ahol a p_0q_i indexek szerepelnek) ettől kezdve pedig a másik játékunk szerinti szabályos lépéseket követve rajzoljuk be éleinket. (3.4 ábra)

3.6. Bástya játék

Tekintsünk most meg még egy játékot, a Bástya játékot. Adott egy 8×8 – as sakktáblánk, rajta egy bástyával. A játékosok a bábuval csak balra vagy felfelé léphetnek egy



3.4. ábra. két halom összege

adott pozícióból és az lesz a nyertes aki a bal fölső sarokba helyezi a bábut. Tehát a bal fölső saroknak megfelelő állás Grundy – száma 0 lesz. Ezek után a többi mezőt is könnyen kitölthetjük Grundy – függvényünk segítségével. (3.5 ábra)

0	1	2	3	4	5	6	7
1	0	3	2	5	4	7	6
2	3	0	1	6	7	4	5
3	2	1	0	7	6	5	4
4	5	6	7	0	1	2	3
5	4	7	6	1	0	3	2
6	7	4	5	2	3	0	1
7	6	5	4	3	2	1	0

3.5. ábra. Bástya játék Grundy – táblája

Ha jobban megfigyeljük ezt a játékot, akkor rájöhetünk, hogy ez tulajdonképpen nem más, mint két halom játék összege. Kezdetben van két darab, 7 kavicsból álló kupacunk. Játékosunk válasszon az első kupacból és begyen el belőle tetszőleges számú kavicsot. Ez pontosan annak felel meg, mintha a Bástya játékban lépnénk vízszintesen ugyanannyit, ahány kavicsot vettünk el a halomból. Illetve hasonló módon most válasszunk a második kupacból. Ez a Bástya játékra kivetítve pedig olyan, mintha ott lépnénk, csak most nem vízszintesen, hanem függőlegesen. Így nem meglepő, ha 2 halom játék összegének szeretnénk megnézni a Grundy számait, pont a Bástya játék Grundy tábláját kapnánk, hiszen a két játék ekvivalens.

Jelölje ismét J_1 az első illetve J_2 a másodikat játékunkat, akkor a definíció alapján ezen két játékunk összegén a $J_1 + J_2$ játékot értjük. Ez a művelet kommutatív, mivel $J_1 + J_2 = J_2 + J_1$ illetve asszociatív is, ami a definíció alapján úgyszintén adódik. Amennyiben J_1 és J_2 játékunk ekvivalens, azaz $J_1 = J_2$ akkor összegüket $J + J = 2J$ alakban is megadhatjuk.

3.7. Két játék Sprague – Grundy függvénye

Sprague és Grundy felfedezésük után azt is észrevették, hogy ha ismerik az egyszerű játékok Sprague – Grundy függvényét, akkor azon játékok összegének Sprague – Grundy függvénye is egyszerűen kiszámolható. Ehhez definiálnunk kell a következő összeadást:

3.7.1. Az $\oplus$ művelet és tulajdonságai

3.6. Definíció . *[] Legyenek i és j nem negatív egész számok, és jelölje $i \oplus j$ (olvasd: oplusz) az $(i+1)$ sor és $(j+1)$ oszlop metszéspontját. Ezt a műveletet nim – összeadásnak nevezzük.*

Például a Bástya játékban (3.5 ábra) nézve $i = 3$ és $j = 2$ – re: $4 \oplus 3 = 7$.

Ha még nem ismerjük az $i \oplus j$ Grundy – számát egy adott i, j – re, amennyiben adott az összes rájuk következő pozíció értéke, a következő módon könnyen kiszámolhatjuk:

3.7. Definíció . *[]* $i \oplus j = mex\{i \oplus 0, i \oplus 1, ...i \oplus (j-1), 0 \oplus j, 1 \oplus j, ..., (i-1) \oplus j\}$

Ebből következik, hogy bármely a, b illetve c nemnegatív egész számra igaz, hogy ha $c < a \oplus b$ akkor vagy van olyan $a' < a$ melyre $c = a' \oplus b$, vagy létezik olyan $b' < b$ melyre $c = a \oplus b'$ teljesül.

Az imént leírtak merőben hasonlítanak a folytonosság deffiníciójára, ezért használjuk a nim – összeadásra a nim – monoton kifejezést. A nim – monoton tulajdonság megfordítása viszont nem igaz, hiszen ez azt jelentené, hogy az $x \oplus y$ függvény monoton,

azaz x ás y változójában szigorúan monoton. A 3.5 – as ábra is jól mutatja, hogy ez nem igaz.

A 3.7 – es definíció alapján mondhatjuk azt is, hogy az $\oplus$ művelet a, b, c nemnegatíve egészekre kancellatív. Vagyis $a \oplus b = a \oplus c$ – ből következik, hogy $b = c$ és $b \oplus a = c \oplus a$ egyenlőségből is következik b és c egyenlősége.

A $\oplus$ művelet tulajdonságai:

1. Kommutatív – Tegyük fel, hogy ez nem igaz, azaz legyen egy olyan a és b legkisebb összegű pár melyre érvényes az $a \oplus b < b \oplus a$. Ekkor mindenképpen kell lennie egy $b' < b$, melyre $a \oplus b = b' \oplus a$. Mivel $b' + a < a + b$, teljesül hogy $b' \oplus a = a \oplus b'$, ezért $a \oplus b = a \oplus b'$, amelyre $b = b'$ ami ellentmondás. Az „a" – ra hasonlóképpen bizonyítható.

2. Asszociatív – Tegyük fel szintén hogy ez nem igaz. Ekkor legyen egy a, b, c legkisebb összegű számhármas, melyre $a \oplus (b \oplus c) \neq (a \oplus b) \oplus c$. Mondjuk hogy a baloldal kisebb a jobb oldalnál, ekkor létezik egy olyan $d < (a \oplus b)$. De ekkor $a \oplus (b \oplus c)$ = $d \oplus c$. Vagy pedig van olyan $c' < c$ melyre $a \oplus (b \oplus c) = (a \oplus b) \oplus c'$. Nézzük ezt az estet. Mivel $a + b + c' < a + b + c$ ezért $(a \oplus b) \oplus c' = a \oplus (b \oplus c')$. Innen a $o + (b \oplus c) = a \oplus (b \oplus c')$. Erre kétszer alkalmazva a kancellativitást kapjuk $c = c'$ egyenlőséget, ami ellentmondás.

3. $a \oplus b \leq a + b$ minden a, b számra

4. $\oplus$ egységeleme a 0 szám

5. Minden szám inverze önmaga.

   Legyen egy p pozitív egész szám nim – prím, ha nincsenek nála kisebb olyan számok amelyeknek p a nim – összege.

6. Bármely pozitív egész szám vagy nim – prím vagy nála kisebb nim összegére bontható a tagok sorrendjétől eltekintve egyetlen módon

7. Egy szám akkor és csak akkor nim prím, ha 2 hatványa.

8. Különböző nim – prímek összege egyenlő közönséges összegükkel

A (2), (4) és (5) tulajdonságokból megállapítható, hogy a nem negatív egészek a nim – összeadásra nézve csoportot alkotnak, ahol az (5) – ös pont miatt minden elem önmaga inverze lesz.

A (6),(7),(8) meghatározások a (4) és (5) függvényében pedig meghatározzák nekünk két szám nim összeadásának szabályait. Vagyis a következőt mondhatjuk:

3.8. Definíció . *[] A nim – összeadandókat 2 különböző hatványainak összegére bontjuk, s összeadjuk azoknak a kettő hatványoknak egy – egy példányát, amelyek páratlan számú nim – összeadandóban lépnek fel.*

Ez a módszer kicsit másképpen megfogalmazva annyit jelent, hogy az összeadandó számainkat felírjuk kettes számrendszerben. Az azonos helyi értékű számjegyeket összeadjuk, moduló kettővel nézve az összegeket, azaz átvitel nélkül számolunk. Ezzel a művelettel már megfogalmazhatjuk azt a tételt amellyel két játék összegének Sprague – Grundy függvényét megállapíthatjuk.

3.9. Tétel. *[]* $\gamma_{J+K}(p,q) = \gamma_J(p) \oplus \gamma_K(q)$*, ahol* p *és* q *nem más mint rendre* J *és* K *játékok állásai illetve* γ *függvénybeli értékük.*

Nézzük meg példaként a két halom játék összegének egy tetszőleges Grundy számát, hogy ellenőrizni tudjuk az ezzel ekvivalens Bástya játékon a tételünk helyességét. Legyen az egyik kupacunkban 3 kavics, azaz mint korábban megtudtuk ennek Grundy száma 3, illetve a másik kupac kavicsszáma 5 analóg módon ennek Grundy száma sem lesz különb 5 – től. Szükségünk van a 3 illetve az 5 kettes számrendszerbeli alakjára. Azaz $3 = 11_2$ illetve az $5 = 101_2$. Akkor az előzőek alapján:

$$\begin{array}{r} 11 \\ \oplus\underline{101} \\ 110 \end{array}$$

Azaz a keresett Grundy – szám az 110_2. Ezt visszaalakítva tízes számrendszerbeli számmá 6 – ot kapunk, pont azt az értéket amit a szerettünk volna, illetve a táblázat alapján ellenőrizhetünk. Ez a tétel azonban nem csak két azonos játék összegének, Grundy számának meghatározására alkalmas, hanem akár két nem ekvivalens játékra is, a lényeg csak annyi, hogy ismerjük Grundy számaikat az összes pozícióra.

4. fejezet

Egyszemélyes játékok

Ahogy az előző pontban is kielemeztem, az egyszemélyes játékok is reprezentálhatók gráf formájában. Annyi a különbség a kétszemélyes játékokhoz képest, hogy itt csak egy játékos teszi a lépéseket, és nincs ellenfele aki megakadályozná, hogy egy kezdőállapotból egy végállapotba érkezzen. Ámde van egy kis hátulütője az ezen kategóriába tartozó játékok némelyikének. Bár az előző pontokban leírtak alapján csak a véges játékokkal foglakozom, ám ezek nagy része nem mondható végesnek, mivel a gráffal való reprezentálásukban is jól látszódhat, hogy köröket tartalmaznak. Így a játékunk már nem is véges. Azonban ha a köröket valahogy kiszűrjük a játékmenetünkből, akkor máris véges játékot kaptunk.
Az egyszemélyes játékoknál az a célunk, hogy minél hamarabb, azaz minél kevesebb lépéssel eljussunk az adott kezdő állapotunkból, a kívánt végállapotba. Erre lehetőségül szolgálnak nekünk a gráf kereső algoritmusaink, mint például a mélységi bejárás a Dijkstra algoritmus vagy egy speciális esete – egyenlő élsúlyozásra – a szélességi bejárás.

4.1. Szélességi bejárás

A szélességi bejárással tulajdonképpen egy feszítőfát keresünk egy irányított, vagy irányítatlan gráfban, gráfunk egy kezdőpontjából, egy általunk választott végpontjáig. A végpont egy konkrét játéknál nem lesz más, mint a már korábban taglalt gráfbeli pontok, melyekből nem futnak ki élek, azaz a nyelők. A feszítő fa megad egy legrövidebb utat az általunk kért két pont között, amennyiben erre van lehetőség.

4.1. Definíció . *[] Legyen egy $G = (V, E)$ irányított vagy irányítás nélküli gráf és $s, u \in V$ csúcsok, és $s \rightsquigarrow v$ út $< v_0, v_1, ..., v_k >$, ahol $s = v_0$ és $u = v_k$*
Az út hossza legyen, az út mentén érintett élek száma, azaz

$$| \ s \rightsquigarrow u \ | = | < v_0, v_1, ..., v_k > | \ -1 = k$$

Az u csúcs s – től való távolsága legyen az $s \rightsquigarrow v$ utak közül a legrövidebb élszáma, azaz

$d(s, u) = min\{| \ s \rightsquigarrow v \ |\}$ *Ha nincs $s \rightsquigarrow v$ út a gráfban, akko legyen $d(s, u) = \infty$*

Az algoritmus megvalósításához szükségünk van két halmazra és 2 tömbre. Legyen az egyik halmazunk H, amely tartalmazza azokat a csúcsokat amelyeket már elértünk a bejárás során. Illetve legyen egy Q halmaz, amely tartalmazza azokat a csúcsokat melyek ellenőrzésre várnak, azaz még nem értük el őket. Ezenkívül szükségünk van még két tömbre, egy $d[1..n]$ mely a kezdőcsúcs és egy tettszőleges csúcs távolságot tárolja, illetve egy $\Pi[1..n]$ mely az egyes csúcsokhoz tartozó szülőt kódolja majd. Egy u csúcs szülője alatt pedig azt az u' csúcsot értjük, ahonnét eljutottunk az u–ba. Az n pedig gráfunk pontjainak a számát jelöli.
Ezek után vezessük be a következő jelölést a gráf csúcsainak színezésére nézve, ahol u jelentsen egy tetszőleges csúcsot a gráfunkban:

- Fehér – ha $u \notin Q$ és $u \notin H$, azaz még nem értük el csúcsunkat
- Szürke – ha $u \in Q$ és $u \in H$, ami annyit jelent, hogy a csúcsunkat elértük , de a gyerekeit, vagyis azokat a csúcsokat amik u-ból elérhetőek, még nem találtuk meg
- Fekete – ha $u \notin Q$, illetve $u \in H$, ami azt jelenti, hogy u – t már feldolgoztuk, és megtaláltuk a gyerekeit.

A kezdetben H és Q is egy üres halmaz, és d tömb minden egyes elemének végtelen értéket adunk, hiszen még nem értük el, egyetlen csúcsot sem. Π minden egyes elemének pedig egy NIL értéket adunk. Ezek után H és Q értéknek megadjuk a kezdőcsúcsunkat, d

első elemét 0-ra állítjuk, hiszen a kezdőcsúcs eljutási ideje nulla, illetve Π első eleme változatlan marad, hiszen egy kezdőcsúcsnak nem létezik szülője. A kezdőcsúcsunk színét szürkére állítjuk. Ezek után nézzük Q első elemét, ami az imént berakott kezdő csúcs lesz, és megkeressük a gyerekeit. A gyerekek sorszámát belerakjuk sorban a Q halmazunkba és színüket szürkére változtatjuk. A megtalált gyerekek sorszáma $d[gyerekek] = 1$ hisz ők az első elértek a kezdőcsúcsból. A kezdőcsúcsunk immár fekete színt ölthet, hiszen már bejártuk. Ezek után Q halmazunkból, mint egy veremből elölről vesszük ki az elemeket, (melyek tulajdonképpen a kezdőcsúcs funkcióját töltik be) és keressük meg gyerekeiket. A megtalált gyerekeket analóg módon szürkére változtatjuk, és bekerülnek a Q végére. Ha azonban egy Q – beli elemhez keressük a gyerekeket és olyat találunk melynek a színe fehértől különbözik ebben az esetben nem foglakozunk vele, és folytatjuk az algoritmust. A $\Pi[]$ értékében mindig az aktuális gyerek szülőjét rögzítjük. A $d[]$ értékét pedig rendre egyesével növeljük szintlépésenként. Az algoritmus akkor ér véget, ha a Q halmaz kiürül, illetve ezzel egy időben d illetve Π tömbök utolsó cellaértékei is megváltoznak.
Az algoritmussal kiküszöböltük bármely játékunk végtelenig tartó lefolyását, hiszen olyan pontokat nem érinthet a feszítőfánk amelyeket már egyszer megtalált az algoritmusunk.

5. fejezet

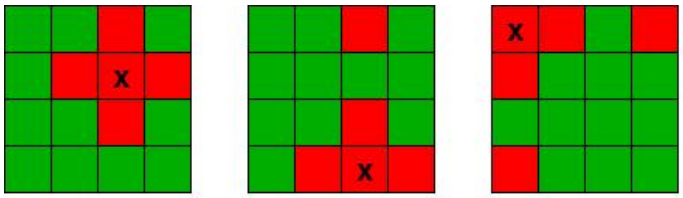
Button Madness

5.1. A játék leírása

Még az első fejezetben említésre került egy Button madness nevezetű játék. Ez nem más, mint egy egyszemélyes kattintgatós játék. A játék szabályai viszonylag egyszerűek. Az eredeti játék egy 4×4 – táblán játszódik. Szóval adott egy 4×4 – es tábla, melyben minden egyes cella egy nyomógomb. Az egyes négyzetecskék mindegyike vagy zöld, vagy piros színt ölt fel magára. Ezek után kezdődhet a játék. A játékban egy lépés abból áll, hogy kattintanunk kell egy általunk kiválasztott mezőre. A kattintás után a következő történik:

- Az általunk kattintott mező színe ellenkezőjére vált (piros-zöld)
- Az általunk választott mező jobb, illetve bal szomszédja színt változtat
- A választott mező feletti és alatti mező is színt változtat
- Amennyiben egy szélső pontra kattintottunk, akkor ha az előző két pontban leírtak kicsúsznának a táblánkról, ebben az esetben a színváltás tóruszosan, azaz az átellenes oldalon érvényesül.

A játék célja az, hogy egy adott állapotból (szélsőséges esetben a csupa zöld állapotból) a csupa piros állapotba jussunk. Vagyis minden egyes négyzetecskénk piros színű legyen.



5.1. ábra. Játékszabályok – X kattintási hellyel

Az ebben a fejezetben bemutatott eredmények Blokhuis [3] dolgozatát követik, részletesebb magyarázatokkal.

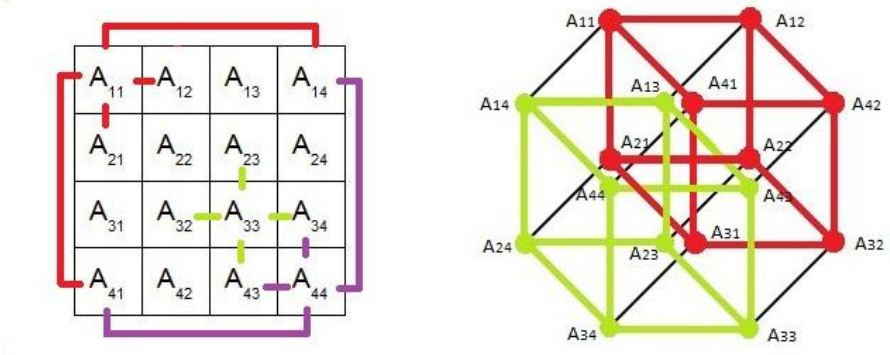
Ha meg akarnánk számolni, hogy egy ilyen 4×4 – es játéktábla hány játéklehetőséget nyújt nekünk, azt könnyen megtehetjük. Minden nyomógombunk színe kétféle lehet: piros és zöld. Összesen 16 nyomógombunk van, vagyis 2^{16} különböző játékunk, illetve a már korábban taglalt állapotunk van, ha a táblánk szimmetriáit nem számítjuk bele.

Minden egyes állapotunkhoz tartozzék pontosan egy „nyomáshalmaz”, azaz azon gombok halmaza, melyek megnyomása után elérjük játékunk célját, vagyis az egyszínű piros táblát. A nyomáshalmazunk-béli elemek sorrendje nem számít, vagyis bármilyen permutációjuk eljuttat minket a játékunk céljához. Továbbá nevezzük egy gomb lenyomását jelentéktelennek, ha egy játékban ezt a gombot páros sokszor nyomjuk meg, hiszen ebben az esetben olyan mintha meg sem nyomtuk volna. Ebből fakadóan, egy gomb 1-nél többszöri lenyomása szintén értelmetlen dolog.

5.2. Érdekes ábrázolás

Érdekességképpen próbáljuk meg ábrázolni a játékunkat geometriailag. Gráfunk minden egyes csúcsa feleljen meg a 4×4 – es táblánk egy – egy nyomógombjának. Vegyünk ezután játékunk első sorának megfelelő gráfbeli csúcsainkat, és kössük őket össze a megengedett szabályok szerint. Így kapunk egy 4 hosszú kört. Ugyanezt a folyamatot hajtsuk végre játékunk első oszlopának megfelelő csúcsain, ahol hasonlóképpen egy 4 hosszú kört kapunk majd a tóruszosság miatt. Ezen két, 4 hosszú körünk Descartes szorzata adja meg a játékunkból készíthető gráfot, ami ugyanazt a végeredményt adja, mintha minden egyes csúcsunkat végigjártuk volna, és két csúcs közé pontosan akkor

húztunk volna élt, ha azt a játékunk szabálya engedi. A két darab négy hosszú körünk Descartes szorzata pedig egy 4 dimenziós kockát rajzol ki nekünk a síkba, ami az 5.2 –es ábrán szépen látható.



5.2. ábra. 4 – dimenziós kocka

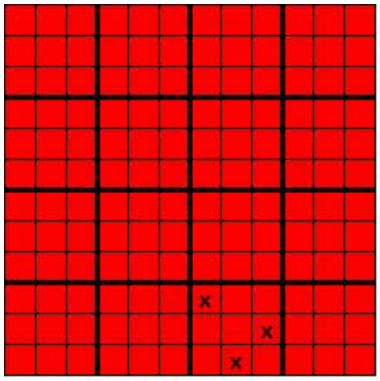
Egy másik gráfbeli ábrázolása is érdekes lehet ezen játéknak. Hasonló módon, mint az előbb szükségünk van a 16 csúcsra, ami itt is rendre a nyomógombjainkat tükrözi. Ezek után szükségünk van még egy 16 csúcsból álló halmazra, amelyeket rendre megfeleltethetünk az előző 16 csúcs mindegyikével. Gondolatban ezt az új 16 pontból álló halmazt helyezzük az előző fölé. A továbbiakban az első halmaz mindegyik pontját kössük össze a második halmazunk azon pontjaival úgy, hogy azok játékunk szabályait kielégítsék, azaz ha egy első halmazbeli pontot választunk (ami azt tükrözi, hogy a játékban megnyomtuk a neki megfelelő gombot), akkor azt pontosan akkor kötjük össze egy második halmazbeli ponttal, ha a játékban ezen nyomógombok színt váltanak. Figyelembe véve játékunk első szabályát, – azaz az általam választott mező színe az ellenkezőjére vált – így a gráfkban minden pont össze lesz kötve a második halmazban lévő megfelelőjével. Minden csúcsunk foka így 5 lesz és az elkészült alakzat pedig egy 5 dimenziós kockát fog ábrázolni.

5.3. Button Madness nagyobb méretekben

Nevezzünk a továbbiakban egy m számot rossznak abban az esetben ha az $m \times m$ –es Button Madness tábla nem megoldható. Azaz létezik egy olyan kezdő állapotú játék, melyből nem tudunk eljutni a teljes egyszínűig. Az $m \times m$ – es játék szabályai változatlanok a 4×4 táblájéhoz képest.

Mivek ugyanannyi állapotunk van, mint ahány nyomáshalmazunk, ha az $n \times n$ – es táblára megoldható a Button Madness, akkor egyféleképpen oldható meg. Ha viszont van olyan állapot, amelyre nem oldható meg, akkor olyan is van, amely két lényegesen különböző módon. Ha az m rossz, akkor az azt jelenti, hogy létezik olyan kiindulási állapot az állapotterünkön, amely több mint egyféleképpen oldható meg. Azaz 2 vagy annál több nyomáshalmaz tartozik az állapotunkhoz. Ezen nyomáshalmazokra teljesülnie kell annak is, hogy lényegesen különbözzenek, vagyis nem csak sorrendjükben térnek el egymástól. Példaképp nézzünk meg egy $m \times m$ szélsőséges esetet az m rossz mivoltának eldöntésében. Legyenek az $m \times m$ – es játékunk nyomógombjai már mind egyszínű pirosak. Azaz játékunk készen van, nulla lépésből eljutunk a végjátékig, hiszen abban vagyunk. Ámde tegyük fel, hogy találtunk egy olyan pár lépésből álló nyomássorozatot amivel szintén visszakapjuk a csupa piros állapotot, feltételezve, hogy ugyanoda nem kattintunk, mert az egy elhanyagolható lépés lenne. Mivel az állapotok száma egyenlő a nyomássorozatok számával, így ha van megoldás, akkor az egyértelmű. Ebből adódóan eljutottunk első lemmánkhoz, azaz:

5.1. Lemma. *[] Ha m rossz és m osztója n – nek, akkor n is rossz lesz.*



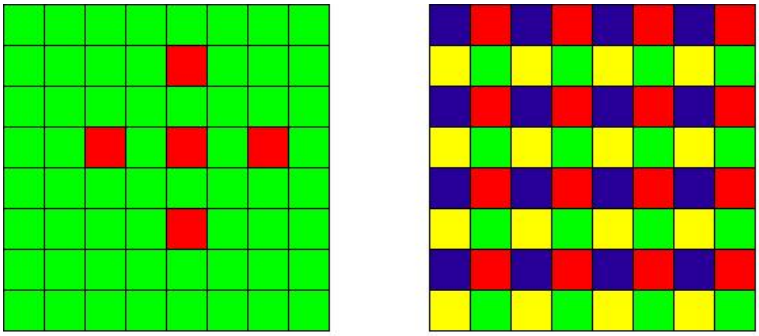
5.3. ábra.

Legyenek például 5.3 ábrán látható x – ek melyekre kattintva tételezzük fel, hogy a csupa piros állapotból a csupa pirosba jutunk vissza. Ha azon az $m \times m$ es táblán ahol az x – eink vannak ez megtörténik, akkor ez az összes többi $m \times m$–es résztáblán az x–eknek megfelelő mezőkre kattintva is teljesül. Azaz a 5.1 – es lemmánkat igazolja.

A következő játékra vonatkozó lemmánk legyen:

5.2. Lemma. *[] Ha m az nem rossz, akkor a 2*m sem rossz*

5.3. Definíció . *[] [Kettős Kereszt] A láthatóság kedvéért vegyük a csupa zöld állapotot egy 2m*2m es játéknál. Ekkor ha egy tetszőleges gombra kattintunk, majd annak négy szomszédjára, a kettős kereszt konfigurációt kapjuk. (5.4 ábra bal oldala)*



5.4. ábra. Kettős kereszt effektus

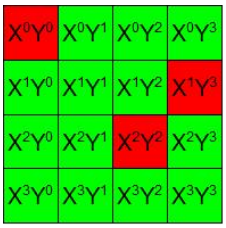
A kettős kereszt mindegyik gombra végigjátszva egyértelműen megadja, hogy a $2m \times 2m$ – es játékunkat felbonthatjuk diszjunkt részjátékok halmazára. Ezek száma ahogy az az 5.4.ábra jobb oldala is mutatja négy lesz. A részjátékok mérete egy $m \times m$ Button Madness játénak felel meg. Ebből már következik a lemmánk, vagyis ha az $m \times m$ –es játék megoldható, akkor a $2m \times 2m$ is.

Továbbá, ha az m jó, akkor ahogy azt már korábban említettem az $m \times m$ – es tábla minden állapotához tartozik egy nyomássorozat, ami egyértelmű és a csupa piros állapotba vezet minket. Ezen nyomássorozat tagjait kedvünkre permutálhatjuk. Mivel nem számít, hogy egy nyomógomb illetve szomszédai színét mikor válltoztatjuk meg, ugyanúgy a csupa piros állapotot kapjuk.

5.4. Algebrai okoskodás

A továbbiakban közelítsük meg a problémát algebrai módszerek segítségével. Legyen adva egy $n \times n$ – es Button Madness táblánk. Jelölje i a táblánk sorainak a számát, mely fusson 0 – tól $(n-1)$ – ig. Az első sor indexe legyen 0, az n. – ik utolsó soré pedig

$n-1$, illetve analóg módon jelölje j az oszlopok indexét. Ezek után minden egyes kis négyzetecskét címkézzük fel X^iY^j monommal. Ha ez megvan akkor tekintsük azt a polinomot, amely összegzi a piros címkéknek megfelelő tagokat.



5.5. ábra.

Például az 5.5 – ös ábrán lerajzolt állapothoz tartozó polinom a következőképpen néz ki:

$$f(X,Y) = X^0Y^0 + X^1Y^3 + X^2Y^2.$$

Ezek a polinomok pedig egy kételemű test fölött vannak definiálva, azaz elemei az $F_2[X,Y]$ – nak. Az F_2 test elemei a 0 illetve az 1 értékek, ezek lesznek a polinom együtthatói. Nyilván a zöld mezők 0 együtthatóval fognak rendelkezni a pirosak pedig eggyel a polinomunkban. A játék tóruszos tulajdonsága miatt felírható az

$$R = F_2[X,Y]/(X^n - 1, Y^n - 1)$$

faktorgyűrű ami tulajdonképpen annyit jelent, hogy a polinomokkal való számolásnál az X^n -t helyettesítjük 1 – gyel. Az eredeti játékunkban ez annyit jelent, hogy ha valamelyik szélső pontra kattintunk, például az X^0Y^3 címkével ellátott mezőre, a tőle jobb oldali mező is színt vált a szabályok szerint, de ugye tőle jobbra már nincs mező. Ezért ugrunk az átellenes oldalra.

Továbbá tekintsük az

$$i(X,Y) = 1 + X + X^{-1} + Y + Y^{-1}$$

polinomot, ami tulajdonképpen játékunk szabályainak levetítése algebrai alakra. Hiszen, ha az (i, j) – re kattintunk, akkor az $X^iY^j(1 + X + X^{-1} + Y + Y^{-1})$ szorzat megadja, hogy állapotunk hogyan változik. Ugyanis $X^iY^j \cdot 1$ jelöli azt, hogy a kattintott helyünk színe megváltozik, a X és $X^{-}1$ – el való szorzat épp a jobb és bal oldali szomszéd koordinátáját adja, illetve az Y és Y^{-1} való szorzata az X^iY^j – nek a kattintott mezőnk feletti és alatti mezőt adja. Szóval ezzel megkapjuk a változó mezők koordinátáit és a teendőnk már csak annyi, hogy ezeket hozzáadjuk $f(X, Y)$ - hoz. Azaz egy kattintás után a következő történik:

$$f(X, Y) := f(X, Y) + X^iY^j(1 + X + X^{-1} + Y + Y^{-1}).$$

Előfordulhat ebben az esetben, hogy valamelyik polinom változónk együtthatója 0 és 1 – től különböző, azaz 2. De mivel az F_2 testben vagyunk ezért modulo 2 nézve az a koordináta el is fog tűnni. Viszont ha megnézzük az eredeti játékunkat akkor ez nem is baj, hiszen az $f(X, Y)$ polinom csak a piros mezőinket tartalmazza. Ha egy kattintás során egy X^iY^j tag i,j konkrét értékre újra bekerül a polinomba, azaz $2 \cdot X^iY^j$ fog szerepelni az összevonás után. Ez jelenti, hogy játékunkban az i,j koordinátának megfelelő gomb színe pirosról zöldre változott, de ezek után így semmi keresni valója az $f(X, Y)$ polinomunkban.
Az $i(X, Y)$ polinom többszöröseinek halmaza legyen I, ahol az $(i(X, Y))$ főideál R – ben.

Induljunk ki mondjuk a csupa piros zöld állapotból (azaz a 0 polinomból), és szeretnénk eljutni abba az állapotba, amikor a (0,0) koordinátája Button Madness táblánknak piros, az összes többi pedig zöld. Az n akkor és csak akkor jó ha ezt lényegében egyértelműen meg tudjuk tenni. Legyen $S(X, Y)$ a megnyomott mezőknek megfelelő X^iY^j monomok összege, amivel a kívánt állapotunk elérhető. Ekkor az $f(X, Y)$ csak a $X^0Y^0 = 1$ értékkel egyenlő, vagyis

$$S(X, Y)(1 + X + X^{-1} + Y + Y^{-1}) = X^0Y^0 = 1.$$

Ez azt jelenti, hogy az 1 eleme I – nek, azaz 1 – nek többszöröse lesz $i(X, Y)$ – nak, ebből pedig következik, hogy minden lehetséges polinom többszöröse lesz $i(X, Y)$

– nek, vagyis $I = R$. Megfordítva, ha $I = R$, akkor a csupa zöld állapotból el tudunk jutni tettszőleges állapotokba, hiszen minden $f(X, Y)$ felírható $S(X, Y) \cdot i(X, Y)$ alakban. Azaz adott n akkor és csak akkor lesz jó, ha $I = R$ – rel.

Az eddigi polinomjainknál X és Y helyébe csak 0 vagy 1 – et tudtunk behelyettesíteni, mert ezek az F_2 test elemei. De mi többet szeretnénk! Vegyük elő az előző helyettesítésünket, azaz $X^n = 1$ -et. Ekkor X helyébe nyilván az n – edik egységgyökök jöhetnek szóba. Ámde ezeket az értékeket F_2 testünk nem tartalmazza. Egy olyan testre lenne szükségünk melyek tartalmazzák F_2 elemeit, illetve az összes $n.$ – edik egységgyököt egyaránt. Ezt úgy érhetjük el legkönnyebben, ha F_2 egy bővítését nézzük. F_2 véges bővítései pedig 2^r elemű véges testek. Az $n.$ – dik egységgyökök akkor lesznek F_{2^r} – ben, ha $n \mid (2^r - 1)$ – nek, például azért mert F_{2^r} multiplikatív csoportja ciklikus lesz.

Ami biztosan jó lesz nekünk e bővítésre az a $2^{\varphi(n)}$ elemszámú test, ahol φ az Euler-féle függvény. Illetve még azt is tudjuk az 5.1 – es lemma alapján, hogy csak a páratlan n – eket érdemes vizsgálni.

5.4. Tétel. *[] Az n szám akkor és csak akkor rossz, ha az $i(X, Y) = 1 + X + X^{-1} + Y + Y^{-1} = 0$ egyenletnek létezik megoldása az n – edik egységgyökök körében (a bővített F fölött).*

Bizonyítás. „$\Longleftarrow$”: Legyenek a és b olyan egységgyökök, amelyekre az $i(a, b)$ polinom egyenlő lesz 0 – val. Legyen a Ψ olyan, hogy az $F_2[X, Y] \to R$. $\Psi(f(X, Y)) = f(a, b)$ leképezés egy olyan nem triviális gyűrű homomorfizmust definiál, melyre $I \subset \ker \Psi$, akkor $I \neq R$ – rel.
Más szóval, ha az $i(X, Y)S(X, Y) = 1$ az R–ben, akkor az $i(a, b)S(a, b) = 1$ az F – ben, így az $i(a, b) \neq 0$ – val.

„$\Longrightarrow$” :Tegyük fel, hogy az $I \neq R$ – rel. Akkor meg kell mutatnunk, hogy léteznek olyan egységgyökök, legyenek ezek szintén a és b, melyekre $i(a, b) = 0$. Tegyük fel indirekten, hogy nem léteznek ilyen a és b. Ekkor $i(X, Y)$ invertálható lesz R – ben. A tagonkénti négyzetre emelés használatával, (ami lehetséges hiszen $F_{2^{\varphi(n)}}$ felett a négyzet-

re emelés a tagonként végezhető), és felhasználva, hogy $2^{\varphi(n)} = 1$ modulo n, azt kapjuk, hogy $i(X,Y)^{2^{\varphi(n)}} = i(X,Y)$. Azaz $i(X,Y)$ pontosan akkor invertálható, ha:

$$f(X,Y) := i(X,Y)^{2^{\varphi(n)}-1} - 1 = 0.$$

Tekintsünk úgy az $f(X,Y)$ – ra, mint egy közönséges polinomra, melynek mind X – ben mind Y – ban a foka kisebb, mint n. Az a és b egységgyökökre fennáll, hogy $i(a,b) \neq 0$, így $f(a,b) = 0$.
Mindezekből következik, hogy $f(X,Y)$ azonosan 0, bizonyítván az állítást.

□

Ennek következményeképp új bizonyítást adtunk az 5.1 –es és 5.2 –es lemmánkra:

5.5. Tétel. *[] Ha az n szám rossz, akkor annak bármely többszöröse, és n/2 is rossz, amennyiben ez értelmes.*

Az eddigieknél mélyebb eszközöket (Weil becslés véges test fölötti görbe pontjainak számára) használ az alábbi két tétel bizonyítása:

5.6. Tétel. *[] $2^k - 1$ rossz minden k – ra, ha $k \neq 1, 3$.*

5.7. Tétel. *[] $2^k + 1$ rossz, minden k > 0 – ra.*

6. fejezet

Button Madness megoldása másodpercek alatt

Mint azt már korábban is említettem, és ami tulajdonképpen természetes, hogy az egy, vagy akár több személyes játékok számítógépes implementálása által, könnyen és sokkal gyorsabban találhatunk megoldást problémánkra. Olyan dolgokról van itt szó, amit emberi agyunk már – már kénytelen átlátni, hiszen számítógépek segítségével több ezer feltételt, lehetőséget, stratégiát végig tudunk nézni másodpercek lefolyása alatt.
Mikor kiválasztottam egy konkrét játékot, a 4×4 Button Madnesst, amelyet a szakdolgozatomba illesztek, még nem tudtam pontosan, hogy hogyan is fog kinézni az a program, ami majd egy random Button Madness egy adott állásból a végállásba juttat engem. Voltak kósza, és gyorsan elvetendő ötleteim, mint például az, hogy először megnézem, hogy 1 lépéssel megoldható e a feladat, ha nem akkor 2 lépésre, és ezt folytatva addig, amíg nem kapok kézzel fogható eredményt. Mindezt pedig visszalépéses kereséssel. De ez elég bonyolultnak tünt, és áttekinthetetlennek. Ekkor jött az ötlet, hogy valamiképp egy gráfot kéne konstruálni a játékból, és ha ez megvan akkor valamilyen fabejárással már sínen vagyok. Ahol a gráf csúcsai jelentenék a játék egy állapotát, és akkor menne két csúcs között él, ha egy adott csúcsban tárolt állapotból el lehet jutni egy másikba. Az biztos, hogy minden csúcs foka a gráfban 16 lesz. Mert egy állásában a játékunkban 16 különböző helyre kattinthatunk, ami egy állapotból így 16 állapotba vezet minket. Az állapotunk összessége is ismert, hiszen minden egyes nyomógomb színe vagy piros, vagy zöld színt ölthet magára, azaz 2^{16} állapotunk lesz. Így a probléma megoldásának

kivitelezésének útjába már csak az állt, hogy hogyan fogom az összes állapotot létrehozni. Végül rájöttem arra, ami lehet egy rutinos programozónak már egyértelmű. S csak miután rájön az ember – kézenfekvő a megoldás. Mivel két színű lehet egy nyomógombunk, ezért jelöljük binárisan a nyomógombokat, azaz 0-ással ha zöld, és 1essel ha piros, vagy fordítva. Ebből már látható, hogy akkor minden egyes állapotunk a játékban 16 hosszú, 0 és 1 – ekből álló kettes számrendszerbeli számokból áll. Így az összes állapotunk létrehozása nem más, mint 0 – tól kezdve, egészen $2^{16} - 1$ – ig 10-es számrendszerbeli számok kettesbe való konvertálásának az eredménye. Példának okáért, ha egy 10-es számrendszerbeli számunk kettes számrendszerbeli alakja csak 5 bitből áll a szám elé annyi 0-ást írunk, hogy végeredményként 16 hosszú számsorunk jöjjön ki, így ugye nem változtatva a bináris számunk értékén. Így a 0 10-es számrendszerbeli számhoz a kettesbeli 0 tartozik, és kibővítve egy 16 csupa nullákból álló sorozat. A 0 szám az előzőekben leírt zöld színt jelenti, ami a csupa zöld állapot lesz. A csupa piros, pedig nyilván a 16 hosszú 1 – esekből álló karaktersorozat, ez pedig pont a $2^{16} - 1$ 10 – es számrendszerbeli értéke. A programom is ezen ötlet alapján készült el, a választott nyelv pedig a C++.

6.1. Az algoritmus felépítése

Először is deklaráltam egy struktúrát, ami a gráfbeli pontokat paraméterezi

```
struct node
{
bool color[16];
int edge[16];
};
```

6.1. ábra.

Azaz áll egy 16 hosszú boole változóból, ami megadja az egyes állapotunk színeit, illetve tartozik még hozzá, egy ugyancsak hosszú változó, melyben minden egyes állapot 16 szomszédját tudom eltárolni, majd lefoglaltam a memóriában 2^{16} darab „node”– t a memóriában. Ezek után egy ciklussal végigfutva 0 – tól $(2^{16} - 1)$ – ig a számokat átalakítom kettes számrendszerbelivé, ezzel megkapva az állapotokat. Majd ezek után az épp

kiszámolt állapot elemein végigmenve:

```
sor= l / 4;
oszlop=l%4;

//A kattintás helye
segedcolor[4*sor+oszlop]=(segedcolor[4*sor+oszlop]+1)%2;

//A kattintástól jobbra és balra:
szam=(oszlop+1)%4;
segedcolor[4*sor+szam]=(segedcolor[4*sor+szam]+1)%2;

szam=(oszlop-1+4)%4;
segedcolor[4*sor+szam]=(segedcolor[4*sor+szam]+1)%2;

 //A kattintástól fel és le
 szam=(sor+1)%4;
 segedcolor[4*szam+oszlop]=(segedcolor[4*szam+oszlop]+1)%2;

 szam=(sor-1+4)%4;
 segedcolor[4*szam+oszlop]=(segedcolor[4*szam+oszlop]+1)%2;

 for (int m=0;m<16;m++)
  {
   graph[i].edge[l]=(graph[i].edge[l])+(segedcolor[m]*kettohatvany[m]);
  }
```

6.2. ábra.

Azaz mivel az állapotok színét egy egydimenziós tömbben tároltam, ha az adott szín zöld, akkor egy 0 – ással, amennyiben pedig piros 1 – essel. Ezért vissza kellett keresni melyik elemről van szó a 4×4 – es táblámon. Majd meghatároztam minden egyes lehetséges kattintásból következő állapotot a fenti kép alapján. Ekkor az így kapott állapotot ismét visszaalakítottam 10-es számrendszerbeli számmá, és a grapf[i].edge[l] értékben tároltam, vagyis a gráf i-edik pontja (ami már egy-egy sima 10-es számrendszerbeli indexszel fut), és az l index szerinti értékét adja, azaz azt az állapotoknak a 10-es számrendszer beli alakját ami akkor következik be, ha a sor = $l/4$ és az oszlop = l modulo 4 helyre kattintok.

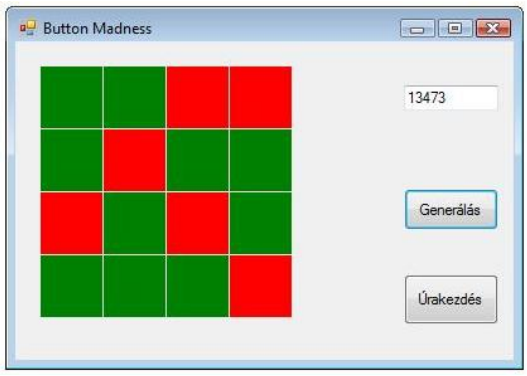
Ezzel a gráf már el is készült. Ebben minden csúcs egy 10-es számrendszerbeli indexszel rendelkezik 0 – tól $(2^{16} - 1)$ – ig, és amint azt az előbb tárgyaltam, ezek pont játékunk állapotai lesznek, illetve az is meghatározható pontosan, hogy melyik csúcsból hova vezet él, azaz melyik állapotból hová lehet eljutni.

Ezek után már nincs más feladat, minthogy valamilyen fabejáró algoritmussal egy adott kezdő, ami tükrözi játékunk egy adott állapotát, eljussunk abba a pontba ami a csupa piros állapotba vezet, azaz a $2^{16} - 1$ indexű pontba. Amennyiben ezt megtaláltuk, algoritmusunk leállhat.

A fabejáró algoritmusok közül a szélességi bejárást kódoltam le, ami azt is elősegíti, hogy a program futása véges időn belül befejeződik. Tudniillik a gráfunkban körök is találhatók. Hiszen már az is egy körnek számít, hogy egy adott állapotból eljutunk egy másikba, akkor nyilván vissza is eljuthatunk, eredeti játékunk kétszer ugyan azon cellájára kattintva, amit a korábbiakban felesleges lépésnek definiáltam. De a szélességi bejárás, mivel egy adott pontot csak egyszer vizsgál, ezért szerencsésen kizárja azt az esetet, hogy körökbe bonyolódjon ciklusunk, így garantálva a programunk, sőt játékunk végességét. A szélességi bejárás megad nekünk egy megfelelő utat a kívánt csúcsunkból, így nincs már más feladat mint ezen út kiíratása, amit a szélességi keresés szintén tárol nekünk, csak visszafelé kell haladnunk a csupa piros állapot csúcsától a kezdeti pontunkig.

6.2. Button Madness

Szakdolgozatomhoz csatoltam még a játékot magát, is amit C# programmal készítettem el. A program maga nem egy bonyolult dolog, 6.2.ábrához hasonló módon adtam meg, hogy egy kattintásnál minek hogyan kell változnia. A program futtatása során a csupa zöld állapotú játék tárul elénk, illetve egy játék generálása során a következőt láthatjuk:



6.3. ábra.

És már indulhat is a játék. A bal fölső sarokban egy számot láthatunk beírva, ami a játékunk kódját tárolja. Tulajdonképpen nem más, mint a már használt játékunk táblájából kigenerált kettes számrendszerbeli szám 10 – es számrendszerbeli alakja. Amennyiben ezt a számot a felhasználó beírja a C++ – os programok bemeneti paramétereként, akkor az

kigenerálja nekünk az adott Button madness tábla azon lépéseit, melyek által eljutunk a kívánt csupa piros állapotba. A 6.3 –as képpel illusztrált játékhoz például a következőt:

```
A kattintások helyei:
4. sor  2. oszlop
2. sor  3. oszlop
4. sor  3. oszlop
3. sor  3. oszlop
3. sor  2. oszlop
1. sor  3. oszlop
4. sor  1. oszlop
1. sor  2. oszlop
```

6.4. ábra.

Ha megfigyeljük az ábrát látunk rajta még két gombot egy „Újrakezdés” illetve egy „Generálás” gombot. Az Újrakezdés gombbal visszaállíthatjuk kigenerált állapotunkat. Generálás gomb segítségével egy új kezdőállapotát hozhatjuk létre játékunknak. A program ezen érdekessége talán az, hogy az adott állapotot az éppen milisecundumra aktuális időből generálja.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Szőnyi Tamásnak, aki segített felkutatni a szakirodalmat, és idejét nem sajnálva segített az idegen nyelvű szövegekek lefordításában és értelmezsében. Aki a nyelvtani és szerkezeti hibák rendkívül kitartó keresésével és iránymutatásaival a szakdolgozatom elkészüléséhez nélkülezhetetlen segítséget nyújtott. Illetve szeretnék még köszönetet mondani csoporttársaimnak, akik segítettek eligazodni a LaTeX világában.

Irodalomjegyzék

[1] CSÁKÁNY BÉLA, *Diszkrét matematikai játékok*

[2] CSIRMAZ LÁSZLÓ, *Játékok és Grundy számaik, Kömal 1980. december*

[3] A.BLOKHUIS, *Button Madness című jegyzete*

[4] FEKETE ISTVÁN, *Gráfalgoritmusok.pdf – oktatási segédanyaga az algoritmusok II. című tárgyhoz*

[5] DEMETER M. IBOLYA, *Visual Studio 2005 & 2008*

[6] http://hu.wikipedia.org