XII. Erdélyi Tudományos Diákköri Konferencia – Kolozsvár, 2009. május 15–17.

On-line Preferansz és robotjátékos változatok

Szerző: Bálint Zsolt
Babeş-Bolyai Tudományegyetem
Matematika-Informatika Kar
Informatika szak, 3. Év

Témavezető: Dr. Simon Károly
egyetemi adjunktus
Babeş-Bolyai Tudományegyetem
Matematika-Informatika Kar
Programozási Nyelvek és Módszerek
Tanszék

Kivonat

A dolgozat a preferansz kártyajáték erdélyi változatának on-line megvalósítását mutatja be. Az első részben az Interneten keresztüli játékra lehetőséget biztosító software architektúrája, valamint a fejlesztésnél felhasznált technológiák és keretrendszerek kerülnek bemutatásra. A második rész robotjátékosok vezérlésére alkalmazható MI módszereket mutat be: egy szabály alapú módszert és egy gráfkeresési algoritmuson alapuló módszert. Az első esetben a robotjátékos kifejlesztése emberi szakértő által összeállított szabályrendszer alapján történik, a második módszer egy klasszikusnak számító megközelítést ötvöz egy Monte Carlo módszerrel, és heurisztikákat alkalmaz a keresési tér szűkítésére. A dolgozat harmadik részében kísérleti eredmények és továbbfejlesztési lehetőségek kerülnek bemutatásra.

1. Bevezető

A dolgozat az erdélyi preferansz kártyajáték számítógépes megvalósítását mutatja be. A bemutatott software lehetőséget biztosít hálózaton keresztüli játékra virtuális játékasztaloknál, és különböző robotjátékos változatokat is tartalmaz.

A preferansz egy német eredetű, vitel alapú, háromszemélyes kártyajáték magyar változata, amelynek egyik alváltozata Erdélyben vált népszerűvé. Az Prefi On-line projekt ennek a változatnak Internetes megvalósítása, melynek kifejlesztését leginkább az a tény motiválta, hogy jelenleg nem elérhető hasonló software (a játék más változataival, például az orosz változatokkal ellentétben).

A kliens-szerver architektúrán alapuló software komplexitása viszonylag nagy, így tervezésének és fejlesztésének folyamán nagy hangsúly került a megfelelő fejlesztési alapelvek betartására és az alkalmazott tervezési minták optimális kiválasztására. Fontos szempont volt a túlterheltség kivédése, hiszen a teljes játékvezérlés a központi szerver feladata, amelynek adott esetben képesnek kell lennie akár több száz bejelentkezett kliens kiszolgálására. A software architektúrájának rövid bemutatása, és a hálózati kommunikáció megvalósítására, valamint a túlterheltség kivédésére alkalmazott módszer rövid leírása is helyet kap a dolgozatban.

A hálózati kártyajáték esetében könnyen előfordulhat, hogy egy asztaltól elveszítjük valamelyik játékost, például hálózati kapcsolatának hibájából, vagy, mert önhibájából lezárja a kliensalkalmazást. Ilyen esetekben a játékos helyét ideiglenesen egy robotjátékosnak kell átvennie. A preferansz játék jellegének következményeként a robotjátékosnak megfelelő mesterséges intelligenciával kell rendelkeznie ahhoz, hogy ne rontson játékostársainak helyzetén. A Prefi On-line projekt több robotjátékos változatot tartalmaz, amelyek szintén bemutatásra kerülnek a dolgozatban. Az egyik megközelítés szerint a javasolt robotjátékos tulajdonképpen egy szakértői rendszer, amelynek kifejlesztése tapasztalt játékosok által összeállított szabályrendszer alapján történt. A másik megközelítés egy a játékok esetében klasszikusnak számító játékfa alapú módszert javasol, amelyet egy Monte Carlo módszerrel ötvöz és heurisztikus szabályokat alkalmaz a keresési tér szűkítésére. A robotok tesztelése és az eredmények kiértékelése külső szakértők segítségével történt, az eddigi eredmények bemutatásra kerülnek a dolgozatban. Megjegyzendő, hogy a robotjátékosok továbbfejlesztése jelenleg még folyamatban van.

A dolgozat második része a preferansz kártyajáték rövid bemutatása, amely tartalmazza a játék történetének rövid áttekintését, az erdélyi változat szabályainak ismertetését, valamint a játék játékelméleti bekategorizálását. A harmadik rész a Prefi On-line software architektúrájának, valamint a felhasznált tervezési mintáknak és keretrendszereknek rövid leírása. A negyedik részben a különböző robotjátékos változatok kerülnek bemutatásra. A robotjátékosokkal elért kísérleti eredmények bemutatása a dolgozat ötödik részében kap helyet, majd a dolgozat végén a következtetések és továbbfejlesztési lehetőségek összefoglalása következik.

A program kifejlesztésénél kiindulási pontként szolgált, a Babe -Bolyai Tudományegyetem, Matematika és Informatika Karán, az Informatika Szak keretén belül 2008-ban kifejlesztett közös projekt, amelynek hallgatóként részese voltam, tervező, vezető programozó és a koordinátor szerepkörökben. Ezúton szeretnék köszönetet nyilvánítani a projekt fejlesztésében résztvevő csoporttársaimnak segítségükért, amellyel hozzájárultak a Prefi On-line software alapját képező prototípus kifejlesztéséhez. A software architektúrájának tervezésénél értékes tanácsokat kaptam Csete Bélától, köszönet ezekért. A szabályalapú robotjátékos nem jöhetett volna létre Bálint Szidónia segítsége nélkül, akinek a tesztelés és eredménykiértékelés terén is sokat köszönhetek. És szeretném megköszönni szakmai irányítómnak, Simon Károly tanár úrnak a segítségét: a projekt gyakorlatilag közös munka eredménye, kifejlesztésének minden szakaszában támogatott.

2. Az erdélyi preferansz

Az alábbiakban a preferansz kártyajáték magyar változatának erdélyi alváltozata kerül bemutatásra: egy rövid történeti áttekintés után dióhéjban ismertetjük a szabályokat, majd játékelméleti szempontból próbáljuk megközelíteni és bekategorizálni a játékot.

2.1 Rövid történeti áttekintés

A preference egy leginkább Kelet-Európában elterjedt kártyajáték [2]. Bizonyos szempontokból hasonlóságokat mutat a Magyarországon közkedvelt ulti játékkal (bár szabályrendszere jóval egyszerűbb). Bár eredeti neve francia szó (préférence), a játék valószínűleg német eredetű. Legvalószínűbb „feltalálói” a dunai svábok, és főleg az Osztrák-Magyar Monarchia területén volt nagyon népszerű a XIX. század első felétől. A játékról az első írásos emlék egy 1829-ben megjelent német könyvben található. Elődje valószínűsíthetően a Boston Whist kártyajáték egyik változata (amely a preference-hez hasonlóan megtévesztő neve ellenére szintén európai kártyajáték) [1]. Az idők folyamán a preference-nek különböző változatai alakultak ki: beszélhetünk osztrák változatról (amely főleg az Osztrák-Magyar Monarchia és Németország területén terjedt el), horvát változatról (amely Horvátország, Szlovénia és Szerbia területén alakult ki), görög változatról és orosz változatról (elterjedt a volt Szovjetunió területén, és több alváltozata is létezik: Leningrad, Sochi, Rostov, és a szabályok szempontjából ezek jobban eltérnek az ősnek tekinthető osztrák változattól). A különböző térségekben a használt kártyapakli is változik: a 32 lapos magyar kártyát (Tell Vilmos kártya), vagy a francia kártya 32 lapját használják a játékosok.

A preferansz a preference kártyajáték magyar változata, amelyet a 32 lapos magyar kártyával játszanak. Ennek némileg módosított formája terjedt el Erdélyben. A játékról nagyon kevés írott dokumentum született, és talán ez az egyik oka annak, hogy a szabályokra nem alakult ki egy egységes „standard”. A különböző játékos közösségek esetében apró eltérések vannak a szabályok alkalmazásának szempontjából. Ennek ellenére létezik egy egységes alap, és különböző körök játékosai nagyon hamar „kiegyezhetnek” a részletekben, így válhatott a preferansz Erdély egyik legelterjedtebb kártyajátékává. Népszerűsége a XX. század végére talán csökkent, de jelenleg is elterjedt, és néhány lelkes játékos körnek és szervezetnek köszönhetően remélhetőleg nem merül feledésbe. Ehhez próbál a dolgozatban leírt projekt is egy kis segítséget nyújtani.

A software elkészítése előtt több preferansz játékossal történt konzultáció alapján lett lerögzítve az a szabályrendszer, amelyen keresztül a játék bemutatásra kerül a következő részben.

2.2 Az erdélyi preferansz szabályai dióhéjban

A preferanszot három játékos játszhatja (esetenként négyen is játszhatják, de ebben az esetben minden játék körben van egy osztó, aki nem vesz részt az aktuális játékban), 32 lapos magyar kártyával. Osztáskor minden játékos 10-10 lapot kap, a maradék két lap a talon.

A játék tulajdonképpen több kisebb játéktípusból áll össze: körökre osztott, és minden kör (osztás) egy-egy ilyen játéktípus szabályai szerint zajlik. A játéktípusokhoz különböző értékek (pontszám) rendelődnek. Az osztást egy licit kör követi. A licitet megnyerő játékos kicserélheti két lapját a talonra (amennyiben nem kézből játszik), és ő dönti el, hogy milyen játékot játszanak a következő körben, az alábbiak közül választva:

Játék neve	Értéke	Teljesítendő cél/feladat
zöld	2	6 vitel, zöld trompf (adu) mellett
sárga	3	6 vitel, sárga trompf mellett
piros	4	6 vitel, piros trompf mellett
makk	5	6 vitel, makk trompf mellett
betli	6	0 vitel, nincs trompf
mízer	7	1 vitel, nem az első és nem az utolsó, nincs trompf
szanzadu (szanadú)	8	6 vitel, nincs trompf
pikoló	9	2 vitel, nem elsőnek, és nem egymás után
preferansz	10	6 piros vitel, piros trompf mellett, talon nélkül (kézből)

1. ***Táblázat***: preferansz játéktípusok

A hat vitelt célként megjelölő játékok esetében hatnál több vitel megvalósítása pozitívum: a játékos több pontot kap. A mízer és pikoló kevésbé közkedveltek, nem mindenütt játszák, és a pikolót néhol a volát helyettesítheti (a feladat: 10 vitel). A pontozás és a teljesítendő feladatok esetében is előfordulhatnak apró különbségek.

Speciális esetek még a körpassz (amikor egyik játékos sem száll be a licitbe, és ilyenkor a cél a minél kevesebb vitel), és a kontrás játékok (az ulti kontrarendszeréhez hasonlóan), amikor a pontszám duplázódik.

Mindenik játék játszható kézből is, ebben az esetben értéke eggyel nő, de a játékos nem kapja meg a talont.

A licitet nyerő játékos játékválasztása után „támadó” szerepet játszik: célja az aktuális játéktípusnak megfelelő feladat teljesítése. A másik két játékos „védő” szerepben játszik az aktuális körben, céljuk megakadályozni, hogy a támadó elérje célját.

A játékszabályok egy részletesebb leírása megtalálható a kliens software „segítség” menüpontjánál, és a későbbiekben felkerül a játék weboldalára.

2.3 Játékelméleti besorolás

Amint az a fentiekből kitűnik a preferansz egy **többszemélyes** (három vagy négy), licit és vitel alapú körökre osztott kártyajáték (trick-taking card game). A játék **véges**, és **nem teljes információjú**: a játékosok csak saját 10 lapjukat ismerik, játékostársuk/ellenfelük lapját nem látják, így 20 kártya aktuális „holléte” ismeretlen számukra. Zéróösszegű: az egyes játékok esetében a lehetséges vitelek száma mindig 10, ezen osztoznak a játékosok, így minél többet visz a támadó, annál kevesebbet a védők: a különböző (védő/támadó) szerepeket betöltő játékosok csak egymás kárára növelhetik nyereségüket. Ezen kívül a játék pontozási rendszerének megfelelően a játék végén a nyereségek összege megegyezik a veszteségek összegével. A játék **sztochasztikus**, nagy szerepet kap benne a véletlen, és a játékosok **kevert stratégiát** követnek: ugyanabban a helyzetben többféle döntést hozhatnak (különböző valószínűségekkel).

A játék alapjaiban **versengő**, nem kooperatív, de az egyes játékkörökön belül a védő szerepet betöltő játékosoknak kooperálniuk kell a támadó akadályozásához. Ez a tény lehetőséget ad számunkra egy egyszerűsítésre: egy játékkörön belül a szerepeket figyelembe véve az esetek nagy részében tulajdonképpen kétszemélyesnek tekinthetjük a játékot.

Játékelméleti szempontból a preferansz több hasonlóságot mutat a bridge kártyajátékkal, amely világszinten közismertebb, így a kapcsolatos cikkek, dokumentumok és számítástechnikai módszerek száma is nagyobb (lásd [3]).

3. A Prefi On-line software bemutatása

A Prefi On-line projekt célja, hogy lehetőséget teremtsen Interneten keresztüli preferansz játékra. Ennek megfelelően egy kliens-szerver architektúrán alapszik, és implementációja Java programozási nyelvben történt, kapcsolódó keretrendszerek és technológiák alkalmazásával.

A felhasználók egy weboldalon regisztrálhatnak, és ugyaninnen letölthetik a kliensalkalmazást. A program elindítása után felhasználónevükkel és jelszavukkal bejelentkezhetnek. Ezután egy központi csevegő-szobába jutnak, ahol látják a bejelentkezett játékosokat és kommunikálhatnak velük. Ezen kívül bejelentkezhetnek létező játékasztalokhoz, vagy saját asztalt hozhatnak létre. Amennyiben egy asztalhoz elegendő játékos jelentkezett be, indulhat a játék. A játék vezérlése a szerver feladata, ő közvetíti a különböző üzeneteket a címzett kliensekhez.

Az alábbiakban a dolgozat nagyon röviden bemutatja a hálózati kommunikációhoz használt keretrendszert, valamint a software architektúráját (a szerver és a kliens oldali alrendszereket). Egy részletesebb leírás a projektről megtalálható a programhoz mellékelt dokumentációban.

3.1 A Mina keretrendszer alkalmazása

A Prefi On-line projekt szerver és kliens része közötti hálózati kommunikáció megvalósítása a Mina keretrendszeren alapszik.

Az Apache Software Foundation által kifejlesztett Mina Framework egy egyszerű, de teljes funkcionalitású hálózati alkalmazás keretrendszer, amely egy egységes API-t biztosít a különböző szállítási protokolloknak [4].
. A Prefi On-line projekt a TCP/IP protokollt használja, amelyet a MINA a Java NIO API-n keresztül valósít meg, de bármely más protokoll használható lenne.

A Mina nagy előnye, hogy egyetlen végrehajtási szálon akár több ezer kliens is tud kapcsolódni az alkalmazás szerver részéhez anélkül, hogy ez gondot okozna az adatok feldolgozásában. Nagyon jól alkalmazható szálmodellje lehetővé teszi a szerver számára egy szálmedence (Thread Pool) használatát, és segíti az aszinkron feldolgozást. A keretrendszer így megkímél több (egyébként költséges) szál-létrehozási művelettől, és segíti a túlterheltség kivédését, ami a preferansz projekt esetében egy nagyon fontos szempont.

A Mina továbbá elősegíti, hogy a beérkező üzenetet több szinten is fel lehessen dolgozni. Ezt a lehetőséget filterek segítségével biztosítja. Az Prefi On-line alkalmazás két filtert implementál. Az első a Loggingfilter, amely a naplózásért felelős, a második pedig a MessageCodecFilter, amely a kimenő és bejövő üzenetek kódolásáért és dekódolásáért felelős.

3.2 A software architektúrája

A software két alrendszerből, egy szerver és egy kliens részből tevődik össze, és a következő csomagokat tartalmazza:

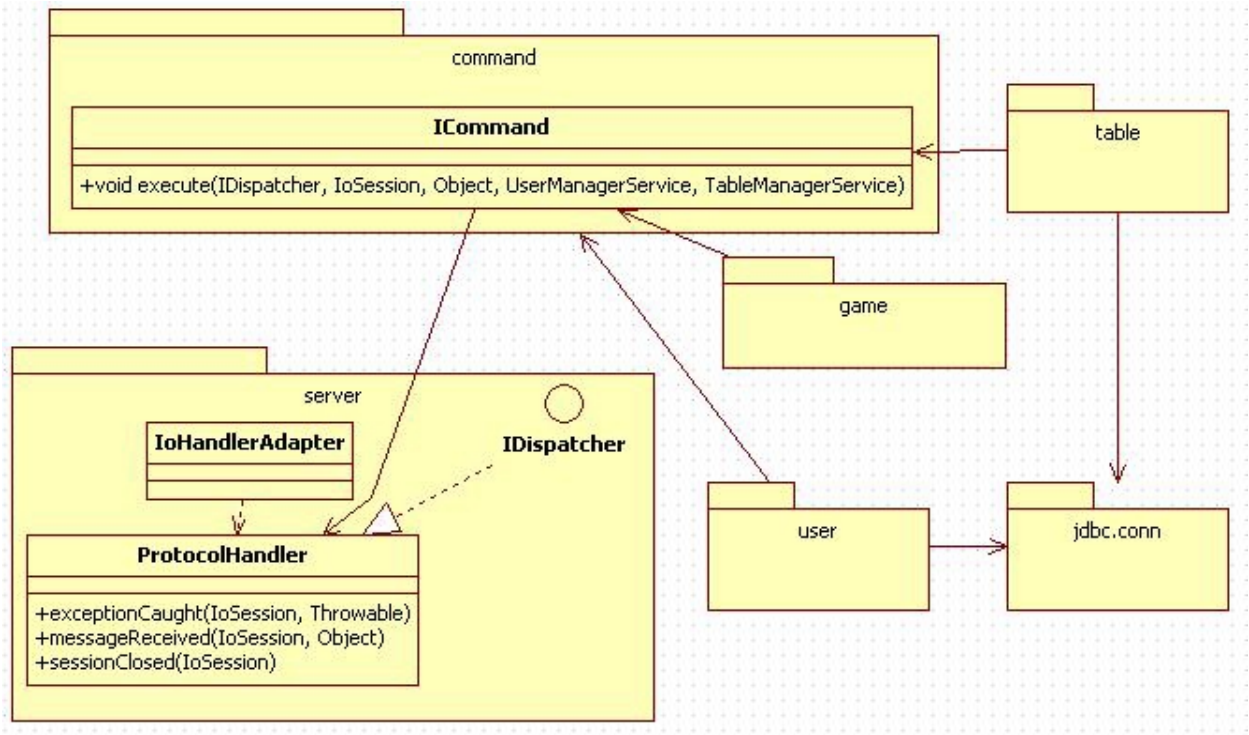
- command: a szervertől érkező utasítások feldolgozása
- connection: a kliens kapcsolódása a szerverhez
- gui: a grafikus felhasználói felülethez szükséges osztályok
- languages: a többnyelvűsítés megvalósítása
- listeners: a back-end és a front-end összekötéséhez használt osztályok (Observer tervezési minta alkalmazása)
- robot: a különböző robotváltozatokhoz tartozó osztályok
- exception: saját kivételek
- message: a kliens és a szerver között küldött üzenetek szerkezete, kódolása és dekódolása
- model: a modell objektumoknak megfelelő osztályok (a projekt magja)

3.2.1 A szerver

A szerver részen található csomagokat, ezeken belül a fontosabb osztályokat és a közöttük lévő kapcsolatokat az 1. ábra szemlélteti.

A szerverhez beérkező üzenetek a ProtocolHandler osztály messageReceived metódusán mennek keresztül. A Factory tervezési minta alkalmazásával minden üzenet külön osztályban kerül feldolgozásra. Ezek a parancs osztályok a command csomagban találhatóak és mindegyik a Singleton tervezési minta szerint implementált. A game csomag tartalmazza a játéktípusokra alkalmazandó szabályokat leíró osztályokat (ki kezdi az adott játékot, adott

körben ki visz, játék végén ki hány pontot kap, stb.). A játékok szétosztása külön osztályokra szintén a Factory tervezési minta alkalmazásával történik. A jdbc.conn csomagban találhatóak az adatbázis kapcsolatért felelős osztályok. A table és a user csomagok az adatbázisban tárolt adatok eléréséért és feldolgozásáért felelősek. Az alkalmazás a többrétegű software architektúrák (multitier architecture) tervezésének alapelveit követi, az adatbázisban tárolt adatok elérését a Data Access Object (DAO) tervezési minta alapján valósítja meg. Az adatelérési réteg (Data Access Layer – DAL) felelős az adatok lekérdezéséért, a szolgáltatás réteg (Service Layer – SL) a lekérdezett adatok segítségével felépíti a megfelelő modell objektumokat, és ezeket továbbítja a kliens réteghez. A rétegeknek megfelelő interfészek egy-egy konkrét implementációjának elérése Factory tervezési minta alkalmazásával történik.



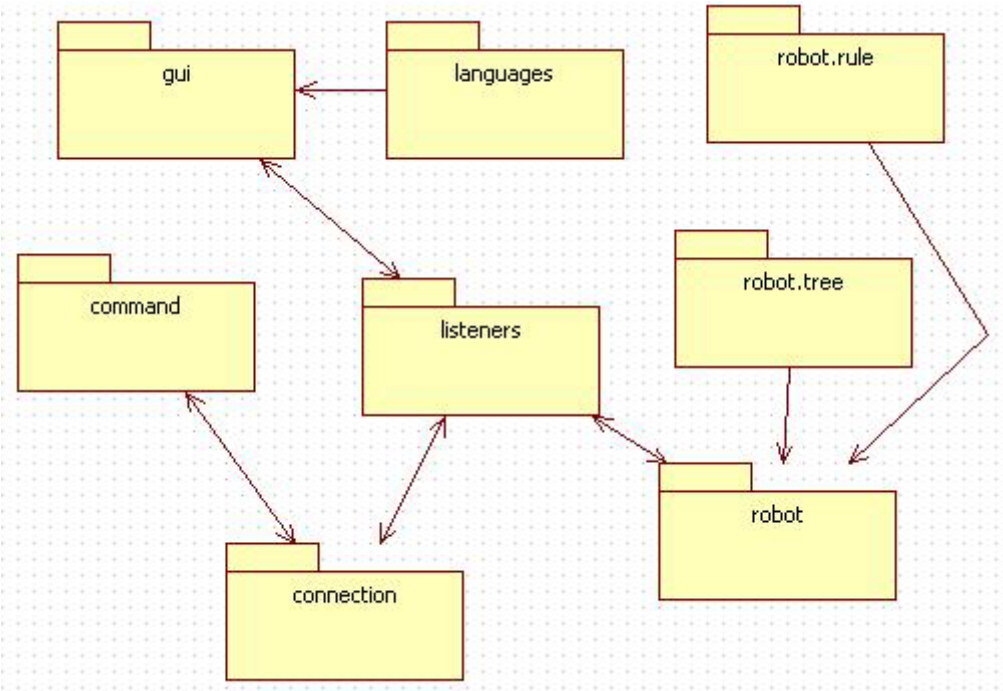
1. ***ábra***: a szerver részen található csomagok, fontosabb osztályok és kapcsolatok

3.2.2 A kliens

A kliens részen található csomagokat és a közöttük lévő kapcsolatokat a 2. ábra szemlélteti.

A klienshez beérkező üzeneteket a connection csomagban lévő Handler osztály messageReceived metódusa osztja el, a Factory tervezési minta alapján. A szerver oldalhoz hasonlóan itt is minden üzenettel egy parancs érkezik, és minden parancsnak külön osztály

felel meg. A parancsok szintén Singleton tervezési minta szerint vannak implementálva. A gui csomag tartalmazza a grafikus felhasználói felülethez szükséges osztályokat. A languages csomag a kliens többnyelvűségét valósítja meg. A listeners csomag a back-en és front-end közötti kapcsolatért felelős, fejlesztése az Observer tervezési minta alapján történt. Minden grafikus felhasználói felületen megjelenített információ a listeners csomag osztályain keresztül jut el a gui csomag osztályaihoz. A robot csomag tartalmazza a robot.rule és a robot.tree csomagokat. A robot.rule csomag a szabályalapú, a robot.tree csomag a játékfa alapú robot megvalósításáért felelős osztályokat tartalmazza.



2. ***ábra***: a kliens részen található csomagok és a közöttük lévő kapcsolatok

4. Robotjátékos változatok

A Prefi On-line projekt esetében, ahhoz, hogy a játék valóban játszható legyen az Interneten keresztül, feltétlenül szükséges megfelelő robotjátékosok biztosítása. A magyarázat: amennyiben egy asztaltól (ideiglenesen vagy véglegesen) elveszítjük valamelyik játékost, például hálózati kapcsolatának hibájából, vagy, mert önhibájából lezárja a kliensalkalmazást, helyét egy robotjátékosnak kell átvennie. Amíg például egy póker program esetében a probléma megoldása relatíve egyszerű egy olyan robot beállításával, amelyik „check-fold” stratégia szerint játszik, addig a preferansz esetében ennél sokkal bonyolultabb a helyzet.

Ennek oka, hogy míg a póker kimondottan versengő játék, a preferansz esetében a védő szerepet betöltő játékosoknak kollaborálniuk kell az egyes körökben. A póker esetében elegendő tehát elfogadható szintre csökkenteni az illető játékos veszteségeit (sőt egy nagyon jó MI motorral rendelkező robot alkalmazása talán igazságtalan is lehetne az ellenfelekre nézve). A preferansz esetében viszont a robotjátékos nem megfelelő viselkedése rontja az aktuális társ eredményét, és igazságtalanul javítja az ellenfél esélyeit, ami elfogadhatatlan (ráadásul a preferansz játékban az emberi játékosok számára is „alapkövetelménynek” tekinthető a jó játéktudás).

A játék nem teljes információs jellege nehezíti robotjátékosok létrehozását. Megjegyzendő, hogy nagyon hasonló a helyzet a bridge kártyajáték esetében is, amire a közelmúltban több módszert is próbáltak javasolni [3]. A preferansz, lévén kevésbé elterjedt, nincs ilyen szerencsés helyzetben, egy megfelelő MI-val rendelkező robot kifejlesztése ezért is jelent kihívást.

A Prefi On-line projekt eddig két módszerrel próbált megoldást találni a problémára, de ezen a téren további fejlesztések is szükségesnek mutatkoznak. A dolgozat következő része a jelenleg alkalmazott módszereket mutatja be: egy szabály alapú módszert és egy gráfkeresési algoritmuson alapuló módszert. Az első esetben a robotjátékos kifejlesztése emberi szakértő által összeállított szabályrendszer alapján történik, a második módszer egy klasszikus megközelítést ötvöz egy Monte Carlo módszerrel, és heurisztikákat alkalmaz a keresési tér szűkítésére.

4.1 A szabályrendszer alapú robotjátékos

A RuBInt PrefBot egy szabályalapú megközelítést alkalmaz: emberi szakértő által összeállított szabályrendszer alapján játszik, tehát tulajdonképpen egy szakértői rendszer (expert system). A szakértői rendszerek, más néven tudásalapú rendszerek (knowledge-based systems), alkalmazása a mesterséges intelligencia területén klasszikusnak számító módszer. Olyan rendszerekről van szó, amelyek megpróbálják „visszaadni” emberi szakértők teljesítményét egy adott probléma megoldásának esetében. Egy szakértők által összeállított „tudásbázist” (knowledge base) alkalmazva szakértők által meghatározott szabályok alapján próbálnak megfelelő döntéseket hozni [5].

A preferansz játék esetében szakértőink tapasztalt preferansz játékosok, akik megpróbáltak összeállítani egy olyan szabályrendszert, amely alkalmazása lehetővé teszi a robot számára, hogy egy potenciálisan nyerő stratégia szerint játsszon. A szabályok alkalmazásával a robot

nem csak arra lesz képes, hogy egy adott helyzetben helyes (a szabályoknak megfelelő) kártyát válasszon, hanem arra is, hogy választásával növelje aktuális nyerési esélyeit.

A módszer jelenlegi megvalósítása teljesen determinisztikus: a szabályok esetében létezik egy szakértők által meghatározott fontossági sorrend. A szabályok kategorizálva vannak aszerint, hogy a robotnak támadóként vagy védőként kell megtennie a következő lépést, továbbá aszerint, hogy első, második vagy harmadik pozícióból kell lapot választania. Támadóként a cél az ütések mihamarabbi átvétele, majd utána az ellenfél aduinak „kicsalása”, és természetesen minél több ütés megvalósítása. Védőként célja, hogy ő vagy társa minél több vitelt „elrabolhasson” az ellenféltől. A szabályok alkalmazásánál szerepet kap az aktuális leosztás ismert része, a robot által potenciálisan megvalósítható vitelek száma (beleértve a „félütések megéltetésére” való törekvést is), a robotjátékos aktuális pozíciója (ellenfél/társ előtt vagy után kell tennie), valamint szerepet kapnak a játék egyéb „íratlan” szabályai.

A teljesség igénye nélkül néhány fontosabb szabály a színjátékok (a vitelek számának maximalizálását célzó játékok) esetében, ha a robotjátékos támadó szerepet játszik:

- ahhoz, hogy a játékos minél hamarabb átvehesse az ütést, az adott színből a lehető legnagyobb lappal kell válaszolnia, az előtte kihívott lapra.
- ha nem tud vinni mindig a legkisebb lapot dobja
- ha az utolsó helyen visz, akkor mindig a legkisebb lappal teszi ezt
- ha a játékos középen van, és előtte egy olyan színt hívnak, amiből neki már nincs, a szükséges legmagasabb aduval válaszol (kivételt képez, hogy ha a király is és a filkó is hiányzik a kezéből, mert akkor az alsóval és nem az ásszal megy be)
- ha a játékos átvette az ütést attól függően hív, hogy hány aduja van, és milyen bizonytalan ütéseket („fél-ütéseket”) akar „megéltetni”: például, ha biztos abban, hogy aduit nem vihetik el („üthetik felül”) (ász, király, filkó vagy ász, király és négy kicsi lap van nála), akkor addig hívja az adukat, amíg az ellenfeleknek elfogy. Ezután a „megéltetendő fél-ütés” színét hívja, és a megmaradt adukkal próbálja meg visszavenni az ütést.
- stb., stb.

Néhány szabály a védőszerepre:

- ellenfél előtt kicsi lappal indul: lehetőleg a kezében lévő „hosszúszínből” (van legalább három belőle) a legkisebb lappal.
- ha valamelyik színből csak egy lapja van („blank”), ami filkó vagy annál kisebb értékű, akkor azzal indul
- a társat nem üti felül (kivéve néhány kivételes helyzetben alkalmazandó szabály esetében, amelyek arra az esetre vonatkoznak, amikor a robot középen van, a támadó előtt)
- stb., stb.

Természetesen ez a felsorolás csak egy nagyon kis kivonata a szabályrendszernek: a felsoroltakon kívül még nagyon sok körülmény figyelembevételére szükség van (meddig licitáltak a többiek, melyik lapot hívta a társ elsőnek, milyen lapot hívott az ellenfél stb.).

A Rubint Prefbot kifejlesztésekor a cél egy alap tudásbázis megadása volt, ami alapján a robot működése már egy olyan elfogadható szintet érhet el, amely lehetőséget ad szakértők általi tesztelésre és továbbfejlesztésre, új, addig figyelmen kívül hagyott szabályok bevezetésével. Ez a továbbfejlesztési tevékenység jelenleg folyamatban van.

4.2 Játékfa alapján megvalósított robotjátékos

A PrefBot GT a preferansz robotjátékos játékfa (game tree) alapján történő megvalósítása. Ez talán a játékok esetében legklasszikusabbnak számító megközelítés, amelyet nagyon széles körben alkalmaztak, a tic-tac-toe-tól a sakkprogramokig [6]. A módszer nagyon jól alkalmazható teljes információjú játékok esetében, azonban nehézség adódik a nem teljes információs játékoknál. Például a preferansz esetében 20 lap aktuális holléte ismeretlen, és minden lehetséges kombinációra külön játékfát felépíteni természetesen lehetetlen. Pontosan ezért a PrefBot GT a klasszikus eljárást egy Monte Carlo (MC) módszerrel ötvözi. Hasonló megoldásokat alkalmaztak például a bridge kártyajáték esetében [3].

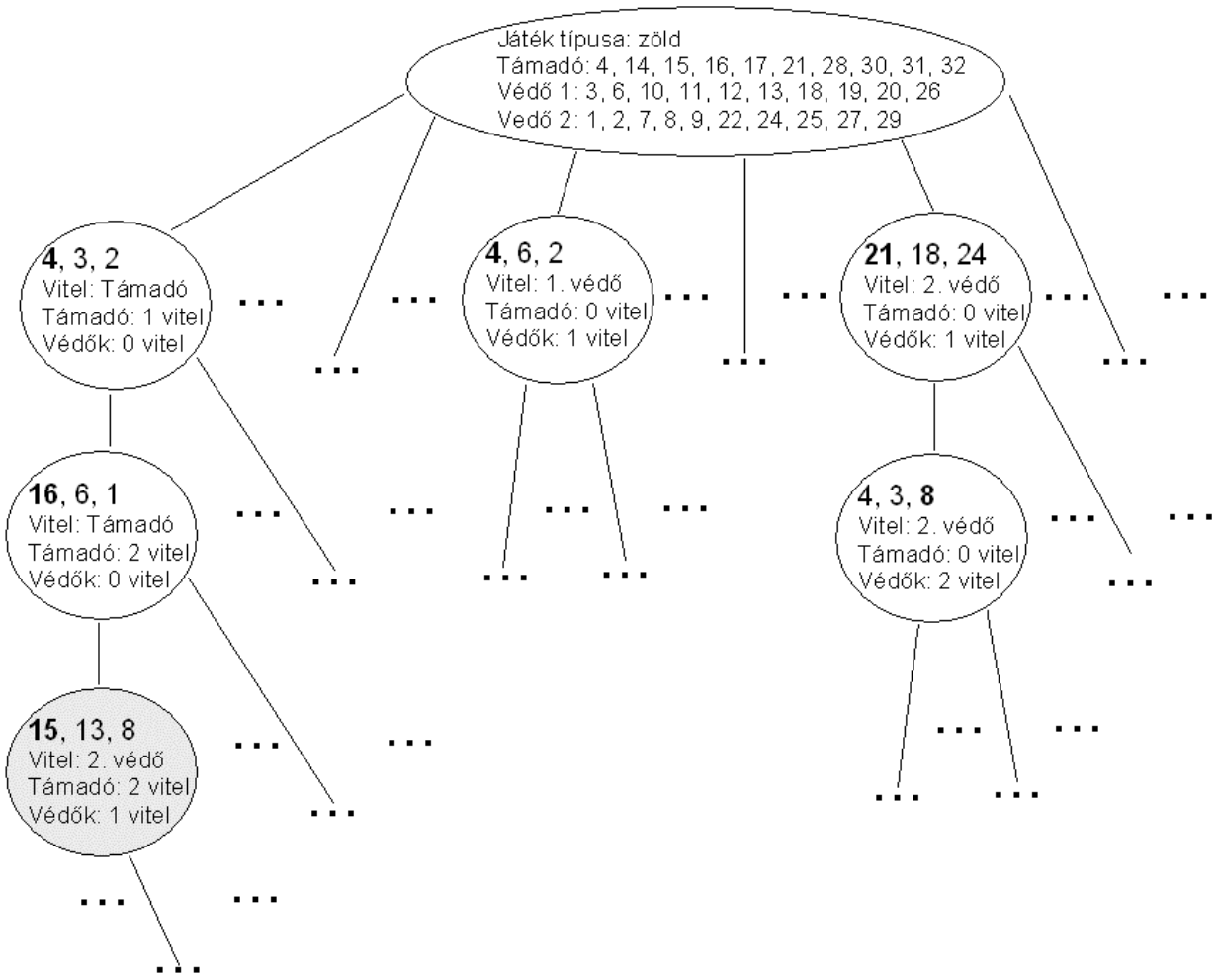
A játékot teljes információjú játéknak tekintjük: egy adott lap-leosztásra implementáljuk a játékfa alapú algoritmust, úgy, hogy mind a 30 kártya helyét ismertnek tekintjük. Véletlenszerűen generálunk különböző leosztásokat, és a legtöbb esetben jó eredményként meghatározott döntés szerint lépünk. A különböző leosztások véletlenszerű legenerálásánál

rulettkerék módszert alkalmazhatunk: a különböző „leosztás-típusok” valószínűségét egyrészt a licit körből vett információk, másrészt a játék szempontjából releváns különbségek figyelembevételével határozhatjuk meg.

Megjegyzendő, hogy a tesztek elvégzésénél külön tárgyaljuk a robotnak egy olyan verzióját, amelyik „csal”, abból a szempontból, hogy a játékfát a tényleges aktuális leosztásnak megfelelően építi fel („meglesve” a többi játékos lapjait).

A preferansz játék esetében alkalmazott játékfa „alapverziójának” szerkezetét a 3. ábra szemlélteti.

A gyökér csúcsban egy adott leosztás kártyái láthatóak, valamint a játék típusának meghatározása. Esetünkben egy zöld játéknak megfelelő játékfát szemléltettünk, tehát az adu szín a zöld. A programban a kártyák számozása 1-től 32-ig történik, a színek, és a színeken belül a kártyák értékeinek növekvő sorrendjében (így az 1-es a zöld hetesnek a 32-es a makk ásznak felel meg). A további szinteken a csúcsokban megjelenő három szám a játékosok által letett kártyákat jelenti (támadó, 1. védő és 2. védő sorrendben). A kiemelt szám az aktuális helyzetben elsőként lehívott lapot jelöli. Ezen kívül további információk jelennek meg: kié az aktuális vitel, és mennyi a támadó és védő játékosok addigi viteleinek száma. Például az első ágon, a harmadik szinten megjelölt csomópont (az ábrán: szürke háttér) esetében a második védő játékos visz, mivel harmadikként adu (zöld) ászt (8) tett, a játékos sárga király (15) hívásába, amire előtte az 1. védő a sárga alsóval (13) tudott válaszolni.



3. ***ábra***: A preferansz játékfa

Természetesen a teljes játékfa legenerálása még ismert leosztás esetében sem lehetséges, hiszen a lehetséges csomópontok száma óriási. Ezért vágásokat kell alkalmaznunk. Hasonló esetekben a legelterjedtebb vágási módszer az alfa-beta vágás (alfa-beta pruning) [6] A PrefBot GT első verziója egy némiképp eltérő módszert alkalmaz: a vágásokat a játékszabályok, illetve emberi szakértők által megadott további heurisztikus szabályok alkalmazásával valósítja meg. Pontosan ezért az első verzió esetében a csomópontok értékeit (a játékosok nyereményét) egyszerűen az addig a pontig megvalósított vitelek számának alapján határozzuk meg, nem alkalmazunk bonyolultabb pontozási algoritmust. A játékfa méretének csökkentésére alkalmazott fontosabb szabályok:

- csak a játék szabályai szerint érvényes lépések kerülnek kiértékelésre (a "színre színt, vagy, ha nincs szín trompfot” szabály szerint)

- az egy kézben lévő, egymás melletti lapok részére csak egy új csomópont kerül kiértékelésre, mivel a játék eredményességének szempontjából nem releváns, hogy közülük adott helyzet melyiket hívjuk

- ha a védő játékosok ütéseinek száma elérte az ötöt, az illető ágon játékosként nem lépünk tovább, mivel a játékot azon az ágon már nem nyerhetjük meg

- ha a támadó ütéseinek száma eléri a hatot, az illető ágon már nem lépünk tovább, mivel mindenképpen nyertünk (természetesen a több ütés pozitívumot jelentene, de a robot gyorsasága esetünkben kritikusabb szempont)

A módszer kiegészíthető további, szakértők tapasztalatain alapuló, játék-specifikus szabályokkal.

A tesztek azt mutatták, hogy a fa mérete még a vágási szabályok alkalmazásával is túl nagy lehet, ezért további szabályok kerültek bevezetésre. Az algoritmust az első nyerő szituáció elérésekor leállítjuk, és a talált nyerő ág szerint lépünk (a megtalált nyerő állapot „fele” teszünk egy lépést), majd a következő lépéseknél mindig újra legeneráljuk a fát, és újra futtatjuk az algoritmust a ténylegesen kialakult helyzet függvényében (az ellenfelek lépései után előállt állapotból).

Mivel a MC módszer szerint több lehetséges leosztás esetében is vizsgálnunk kell a játékfát, és a felhasználók türelmét a program nem veheti túlzottan igénybe, a robot részére gondolkodási időkorlát is megszabható.

A tesztek azonban azt mutatják, hogy a fenti megkötések már túlságosan rontják a robot képességeit (az eredmények a dolgozat következő részében kerülnek bemutatásra), így jelenleg folyamatban van a módszer egy módosított változatának kidolgozása, továbbfejlesztése, teljesítményének javítása.

5. Kísérleti eredmények

A kísérleti eredmények alapjául szolgáló tesztek végrehajtása és kiértékelése külső szakértők bevonásával történt. A tesztelésre használt kliensalkalmazás lehetővé teszi, hogy a szakértő bármelyik robottól döntést kérhessen, majd saját tudása alapján (mit döntött volna ő az adott helyzetben) elemezze a robot döntését.

Három robotváltozat került letesztelésre: a szabály alapú robot, a játékfa alapú robot „csaló” változata (amely „meglesi” a többiek lapjait), és a véletlenszerű leosztásokat alkalmazó játékfa alapú robot. A tesztek elvégzésekor szem előtt volt tartva az elv, hogy a robotok nagyjából azonos számú döntés meghozatalára legyenek „kötelezve”, az aktuális szerep (támadó/védő) és az aktuális pozíció (hív/középen vagy hátul tesz) által meghatározott különböző esetekben. Összesen minden robotnak 270 döntést kellett meghoznia.

Az elért eredményeket az alábbi táblázat foglalja össze (2. Táblázat):

Robot típusa	**Jó döntések száma**	**Rossz döntések száma**	**Pontosság**
Rubint Prefbot szakértői rendszer	230	40	~85%
Prefbot GT, „cheater” játékfa alapú, „csaló”	196	74	~72%
Prefbot GT játékfa + MC	195	75	~72%

2. ***Táblázat***: a különböző robotváltozatok alkalmazásával elért kísérleti eredmények

Megjegyzendő, hogy a játékfát alkalmazó robotok esetében felső időkorlát volt bevezetve a robot gondolkodási idejére. Mivel bizonyos esetekben a játékfa alapú robot gondolkodási ideje még viszonylag nagy, a jelen fázisban nem volt arra lehetőség, hogy az MC módszert alkalmazó robot nagyon sok különböző leosztást vegyen figyelembe: minden esetben 10-10 különböző leosztásnak megfelelő játékfát generált a módszer.

Láthatólag a játékfa alapú robotok jóval gyengébben teljesítettek. Ebből azt a következtetést lehet levonni, hogy a jelenleg használt vágási feltételek túl erősek és elvesztődnek olyan ágak, amelyek a jobb eredményekhez vezetnének.

A szabályalapú rendszer ígéretesnek mutatkozik, és megjegyzendő az is, hogy a tesztelés során már sikerült beazonosítani kimaradt szabályokat, amelyek alkalmazása valószínűleg tovább növeli a robot pontosságát. Negatívum a szakértők azon megfigyelése, miszerint ha a robot téved, akkor általában nagyot téved: az érvényes megoldások közül, gyakran a lehető legrosszabbat választja. Ennek oka, hogy az ilyen helyzetekben az alkalmazható szabály

hiánya egy véletlenszerű döntést eredményez, de ezt a problémát orvosolhatja a szabályrendszer kiegészítése.

Szembetűnő a tény, hogy gyakorlatilag nincsen különbség a játékfát alkalmazó robotok teljesítményének tekintetében. Ennek egyik valószínűsíthető oka az lehet, hogy a robot gondolkodási idejére alkalmazott felső időkorlát negatív hatása egyformán befolyásolja a két módszert. Valószínűleg viszonylag nagy volt az olyan esetek száma, amikor a robot nem talált jó megoldást, és ilyenkor tulajdonképpen véletlenszerűen hozta meg döntését. Az MC módszert alkalmazó robot esetében a legenerált leosztások aránylag kis száma mellett az eredményt (a tényt, hogy ilyen formában is megközelítette „csaló” változatának pontosságát) akár ígéretesnek is lehetne tekinteni, de egyértelmű, hogy a módszer jelenlegi formája komoly módosításokra szorul, és a teszteredmények helyes értelmezése is további elemzési munkát igényel.

A játékfa alapú robot esetében a legnagyobb kihívásnak továbbra is a futási idő csökkentése tűnik. Az elképzelés szerint a gyakorlatban működő rendszer esetében a robottal kapcsolatos számítási tevékenységeket a bejelentkezett kliensek elosztják egymás között. Amikor szükség van a robotjátékos döntésére, a szerver elküldi a „feladatot” az összes csatlakozott klienshez (nem csak az érintett játékasztal klienseihez), azok egy-egy külön végrehajtási szálon belül elvégzik a szükséges számításokat, majd visszaküldik az eredményeket a szerverhez, és a szerver az összesített eredmények alapján hozza meg a döntést. Természetesen ehhez először még szükséges a robot „csaló” változatának tökéletesítésére, jelenleg ez jelent prioritást.

A továbbfejlesztése jelenleg folyamatban van, és ezzel párhuzamosan körvonalazódik néhány más lehetőség is, új, eltérő modelleken alapuló robotjátékosok kifejlesztésére is.

Következtetések és továbbfejlesztési lehetőségek

A dolgozat a preferansz kártyajáték erdélyi változatának on-line számítógépes változatát mutatta be. Sikerült megtervezni egy olyan szoftverarchitektúrát, amely lehetővé teszi az Internetes játékot, akár több száz csatlakozott kliens esetén is, és könnyen karbantartható, továbbfejleszthető. Sikerült kidolgozni két módszert robotjátékosok kifejlesztésére. A módszerek segítségével kifejlesztett robotjátékosok a kísérleti eredmények alapján

ígéretesnek bizonyultak. Ez főként a szabályalapú módszerrel megvalósított robotra érvényes, ami már jelen formájában is képes egy kezdő emberi játékos szintjén preferanszozni.

A szoftver továbbfejlesztésének terén a még hiányzó lehetőségek (pl. kontrás játékok) beépítése, a szabályrendszer apró módosításainak lehetővé tétele, a GUI továbbfejlesztése jelent prioritást.

A robotjátékosok esetében még nagyon sok tennivaló van:

- a szabályrendszer kiegészítése
- a játékfa alapú módszer módosítása
- az eddig még nem implementált játéktípusok bevezetése
- a számítási feladatok elvégzésének párhuzamosítása

A tervek között szerepel új módszerek kidolgozása is más típusú robotjátékosok kifejlesztésére: gépi tanulási módszereken és természetből inspirált modelleken alapuló robotjátékosokat is érdemes lenne kipróbálni. Reményeink szerint a továbbfejlesztett robotok hamarosan neurális hálózatokat és genetikus algoritmusokat alkalmazó társaikkal is „összemérhetik erejüket”. Továbbá hibrid megoldások alkalmazásával is meg kellene próbálkozni.

Reméljük, hogy a software hamarosan hozzáférhető lesz a preferansz kártyajátékot kedvelők számára, és a robotjátékosok segítségével valóban visszaadhat majd valamit a „prefi partik” hangulatából. És reméljük azt is, hogy ezzel a projekt hozzájárul majd egy magyar kártyajáték megőrzéséhez, népszerűsítéséhez.

Hivatkozások

1. Thierry Depaulis, (1998), Dissent on the Origin of *Preferansz/Preference*, *http://spotlightongames.com/genealogy/cards.html#dissent*
2. Richard Heli, (1998), Card Games of the Donauschwaben in 18th- and 19th-century Hungary, http://spotlightongames.com/genealogy/cards.html

3. Matthew L. Ginsberg, (2001), GIB: Imperfect Information in a Computationally Challenging Game, Journal of Artificial Intelligence Research, 14, 303–358
4. Trustin Lee, Ashish Paliwal, (2009), Apache MINA Developer Guide, http://mina.apache.org/developer-guide.html
5. Chris Nikolopoulos, (1997), Expert Systems: Introduction to First and Second Generation and Hybrid Knowledge Based Systems, CRC Press.
6. Stuart J. Russel, Peter Norvig, Panem könyvkiadó, (2003), Mesterséges Intelligencia Modern megközelítésben