FIATAL MŰSZAKIAK TUDOMÁNYOS ÜLÉSSZAKA

Kolozsvár, 2004. március 26-27.

A MESTERSÉGES INTELLIGENCIA LABIRINTUSA

Görög Levente, Máthé Zsolt, Komáromi Lóránd, Szilágyi Sándor Miklós

Abstract

In this study we present a specific method for developing intelligent decision-making agents by using different combinations of AI (Artificial Intelligence) algorithms. So far these agents are suitable most for playing *Labyrinth*, a board game invented and elaborated by Ravensburger, but their structure and decision-making mechanism make them capable to adapt to changing environments and changing rules. We suppose there's a general pattern for developing AI players of logic-based games of most kind - presenting a general genetic algorithm and validating the pruning of the games-states containing search tree might be considered attempts to prove our hypothesis.

Összefoglalás

Ebben a tanulmányban egy AI algoritmusok kombinációján alapuló módszert mutatunk be intelligens döntéshozó ágensek kifejlesztésére. Ezen ágensek elsősorban a Ravensburger által megalkotott *Labirintus* nevű táblajáték játszására alkalmasak, azonban struktúrájuk- és működési mechanizmusuknak köszönhetően képessé vál(hat)nak változó környezethez illetve szabályokhoz is alkalmazkodni. Feltételezzük, hogy létezik egy általános minta a legtöbb logikai játék AI játékosainak fejlesztésére - ezt a hipotézisünket próbáljuk igazolni egy általános genetikai algoritmus bemutatásával, illetve a játékalapokat tartalmazó keresési fa megnyesésének validációjával.

Bevezetés

Alaposabb szakismereteket igénylő, jól elszigetelt, és algoritmikus megoldásokat (egyelőre) nem ismerő feladatok számítógépes implementációja immár követelményként jelentkezik - egyre sürgetőbb módon - a XXI. századi számítástechnikában. Ahogyan az automatizálásra váró problémák komplexitása növekszik, úgy válik egyre nyilvánvalóbbá a tény, hogy a klasszikus értelemben vett racionalizáló algoritmusok nem alkalmasak az ilyen jellegű feladatok pseudo-leírására, illetve megoldására. A jól specifikált problémákra megírt *egyszerű reflexszerű ágensek*, akárcsak az ezeket támogató *explicit keresőtáblák* funkcionális használata egy lassú erodálási folyamat során jut majd el a morális kiöregedéshez [7]. Egyáltalán - vetődik fel a kérdés - lehet-e egy megfelelően bonyolult feladatot a hagyományos értelemben *jól* specifikálni, például olyan feladatokat, amelyekkel az intelligens entitások nap mint nap szembesülnek? Kérdéses, hogy megérthetjük-e, milyen

mechanizmusok alapján érzékelünk, tanulunk, emlékezünk és következtetünk, illetve, amennyiben igen, meg tudjuk-e valósítani ennek a megértésnek egy adekvát leírását és ezáltal valamiféle automata rendszerbe történő integrálását: „...annak tanulmányozása, hogy hogyan lehet a számítógéppel olyan dolgokat művelni, amiben pillanatnyilag az emberek a jobbak” (Rich és Knight, 1991). Dolgozatunkban ezekre a kérdésekre szeretnénk (lehetséges) válaszokat adni, illetve további kérdéseket megfogalmazni, mindezt egy konkrét, még fejlesztésben lévő alkalmazásunk tükrében.

A fentiekre reflektálva nyilvánvaló, hogy olyan módszereket keresünk, amelyek plauzibilis távlatokat nyitnak meg intelligens viselkedés modellezésére. Először azonban megemlítjük a már létező ilyen jellegű, validált eljárásokat:

1. **Evolúciós algoritmusok (EA):** a genetika és természetes szelekció fejlődéstani vetületére alapozott adaptív algoritmusok [2].
2. **Neurális hálózatok (NN):** Az agy információ-feldolgozási mechanizmusának analógiájára létrehozott rendszerek, amely felépítésükben gráfra hasonlítanak, neuronokat csomópontokkal, míg kapcsolatokat élekkel reprezentálva [6].

A fent bemutatott módszerek nem tekinthetők homogén eljárásoknak. Reprezentációtól függően lényeges eltérés mutatkozhat implementálásukban, olyannyira, hogy csupán az alapelv marad közös. A módszerek *gyűjtőneve* a **mesterséges intelligencia (AI)**, azaz összefoglalásként:

1. A nem létező algoritmus helyett a probléma megoldását heurisztikus kereséssel* támogatjuk
2. Létrehozuk a heurisztikus keresésnek megfelelő rendszerarchitektúrát (intelligens ágensek)
3. Gyors, empirikus prototípust készítünk - ezzel kísérletezve javítjuk a feladat specifikációját

Egyéni módszerek és technikák

Módszereinket egy konkrét alkalmazás [3] révén fogalmazzuk meg. Az illető program fejlesztése során

1. ábra

folyton általánosítási problémákkal küszködve elhatároztuk, hogy osztott rendszert alakítunk ki, amely a 1. ábrán látható struktúra szerint működik. Ennek az innovatív rendszernek létezik egy kézzelfogható előnye más rendszerekhez képest: nem környezetfüggő, ezért megváltozott viszonyokhoz is képes adaptálódni. Működési elve egyszerű, AI módszerek

összekapcsolásában rejlik. A [3] alkalmazásra reflektálva, a játék következő n állapotait egy keresési fában szeretnénk tárolni, azonban, ha csak az n nem egy végzetesen kis érték, nincsen erre lehetőség (a

* A heurisztikus keresés lényege az exponenciális bonyolultságú és kimerítéses keresés ágainak megnyesése a lényegtelen keresési irányok figyelmen kívül való hagyásával, az exponenciális bonyolultság polinomiálisa való visszavezetése.

játék elágazási tényezője 39 [3], és minden egyes levélcsomópontban útkereséssel szembesülünk^{**}). A keresési fán alfabéta nyesést [6] kell végeznünk az állapottér szűkítése végett. Ezt a hagyományos Greedy módszerrel valósítjuk meg^{***}, de lévén ez mohó algoritmus, heurisztikus keresés esetében kérdéses a hatékonysága, azaz validálni kellett (1. táblázat). A nyesés validációját 3-as és 4-es

Mélység	Továbbvitt állapotok	Optimális nyerési aránya
3	10%	82%
	1%	92%
	1‰	99%
4	10%	69%
	1%	77%
	1‰	85%

1.táblázat

mélységben végeztük: a programot önmaga ellen játsztattuk teljes, illetve megnyesett keresési részfával. A táblázat első oszlopa a bejárt mélységet tartalmazza, a másodikban azt tároljuk, hogy a nyesés után hány százaléka (ezreléke) maradt meg a teljes keresési fának, míg az utolsó oszlop a teljes keresési részfával dolgozó algoritmusnak a megnyesett keresési részfával operálóhoz viszonyított nyerési arányát fejezi ki szintén százalékosan. 3-as mélységben a táblázat minden egyes sorában szereplő szám adatok 100 játék átlagaként értelmezendők, míg 4-es mélységben 10 játék után számítottunk átlagokat. Azon hipotézisünket, hogy nyesési módszerünk hatékony, igazolni látszanak ezek az adatok, hiszen 4-es mélységben a keresési részfa csupán 10%-át megtartva a program 31%-osan teljesített a megnyesetlen keresési részfával operáló gépi ellenfele ellen. A keresés mélységkorlátjának megállapítását (3 illetve 4) nem explicit módon végeztük, hanem genetikusan algoritmus segítségével. Adott mélységig keresve, mérhetjük a keresés relatív hatékonyságát, illetve mérhetjük az adott mélységhez szükséges keresési időt. A hatékonyság (E) és az idő (t) közös nevezőre hozásával (a két tényező súlyozása genetikusan algoritmus segítségével) mérhetővé válik a hasznosság (U). Ezután csupán az $f(E, t) = U$ függvény maximumát kell megállapítani. Magáról az explicit genetikusan algoritmusról itt csak annyit, hogy a megállási fitness-kritérium (életerő objektívum) a konvergáló sorok mintájára a következőképpen van definiálva: $|X_n - X_{n+1}| < \epsilon$. Ez annyit tesz, hogy a populáció két legdominánsabb tagjának fitness-

mélységben végeztük: a programot önmaga ellen játsztattuk teljes, illetve megnyesett keresési részfával. A táblázat első oszlopa a bejárt mélységet tartalmazza, a másodikban azt tároljuk, hogy a nyesés után hány százaléka (ezreléke) maradt meg a teljes keresési fának, míg az utolsó oszlop a teljes keresési részfával dolgozó algoritmusnak a megnyesett keresési részfával operálóhoz viszonyított nyerési arányát fejezi ki szintén

^{**} A legrovidebb út-kereső algoritmus külön fontosságot rejteget a kiértékelő függvényben. Mivel ez a leggyakrabban használt módszer az új irányok keresésére és az állapotok heurisztikus kiértékelésére, hangsúlyt kellett fektetnünk arra, hogy komplexitása a lehető legkisebb legyen, és a lehető legkevesebb, de ugyanakkor a leggyorsabban [1] elérhető memóriát használja. A legrovidebb út-keresés implementálható a legegyszerűbb mélységi bejárásos próbálgatásokkal (backtracking vagy rekurzió), mi azonban a Depth-First-tel szemben a Breath-First bejárást preferáltuk (az előbbi exponenciális komplexitása végett), illetve egészen pontosan az ezen az eljárás alapuló Lee algoritmust [5], amely a szakirodalomban még Fill néven is ismert. Depth-First: egy gráfot úgy járunk be mélységben, hogy vesszük a kezdeti csomópontot és utána az első olyan pontot, amely érintkezik vele. Bejárjuk azt és az ahhoz tartozó pontokat, majd vesszük a következő szomszédját a kezdeti pontnak. Ez nyilvánvaló, hogy mélységi, és rekurzívan fogható fel egy exponenciális komplexitású algoritmus (O(xⁿ), ahol x a pontok szomszédjainak száma (átlagosan), n pedig a pontok száma). Breath-First: próbáljuk meg a vektorelemű vektort (azaz mátrixot) úgy elképzelni, mint egy irányítatlan gráfot, ahol akkor van el két (csomópont) között, ha azok szomszédok. Ezt az algoritmust alkalmazva tulajdonképpen egy szabályozott, feltételes hullámozást [4] végzünk a vektortömbön, melynek eredménye éppen a legrovidebb út a kezdőpont és a többi pont között. így nyilvánvalóan javítunk az alprogram futási idején, lévén ennek az eljárásnak a komplexitása O(m+n), ahol m az élek száma, illetve n a csomópontok száma.

^{***} Adott szint (mélység) levélcsomópontjainak azok hasznossága szerint csökkenő sorrendben megyünk végig, majd egy bizonyos értéknél kisebb hasznosságú csomópontok alatti ágakat lenyessük. A levélcsomópontok hasznosságát a [3]-ban bemutatott módszerrel állapítjuk meg.

mutatója megközelíti egymást (különbségük abszolút értéke kisebb, mint az s hibakorlát), vagyis nincsen már lehetőség számottevő fejlődésre. Az általunk használt genetikus algoritmus részletesen a [3]-ban van bemutatva, itt csupán folyamatábráját illusztráljuk (2. ábra).

2. ábra

Következtetések, célok

Az AI módszerek egy lehetséges jövőképet vetítenek elő a számítástechnikában. Megfelelően nagy állapottér esetében a lehető leghatékonyabbak az ismert algoritmusok közül-, a kisebb terek esetében ez még függőben lévő kérdés. Céljaink közé tartozik az ezzel kapcsolatban felmerülő problémáink során megfogalmazott hipotéziseink tesztelése, egy feladat-specifikációtól független, általános genetikus algoritmus megfogalmazása, valamint pszeudo-nyelven vezérelt interkommunikációs ágensek működtetése a megoldandó feladat specifikálására.

Hivatkozások

1. Todd M. Austin, Dionisios Pnevmatikatos, Gurindar Sohi: **Streamlining data cache access with fast address calculation**, May 1995, ACM SIGARCH Computer Architecture News, Proceedings of the 22nd annual international symposium on Computer architecture, Volume 23, Issue 2
2. Thomas Back: **Evolutionary Algorithms in Theory and Practice**, New York - Oxford, Oxford University Press, 1996
3. Görög Levente, Komáromi Lóránd, Máthé Zsolt, Szilágyi Sándor Miklós: **Az őrült labirintus**, 2004 febr.
4. Xiong Ji-Guang: **An Algorithm for Path Connections Using Asynchronous Diffracted Waves**, JACM(2004)
5. C.Y.Lee: **An algorithm for path connections and its applications**, IRE Trans. EC-10, P. 346 (Sept. 1961).
6. Simon Haykin: **Neural Networks. A Comprehensive Foundation**, Prentice Hall PTR, Upper Saddle River, NJ, 1994
7. Stuart J. Russell and Peter Norvig: **Mesterséges Intelligencia modern megközelítésben**. Panem - Prentice Hall, Inc. 2000

Görög Levente-Károly / egyetemista II, év

Sapientia egyetem / Icland 68, jud. Mures 4296

Tel.: 0742606039; E-mail: goroglev@yahoo.com