****

**Kauno technologijos universitetas**

Informatikos fakultetas

**Modulis „Tiriamasis projektas 1“  
Projektas: Dirbtinio intelekto metodų taikymas strateginiams žaidimams**

Projektavimo metodologijos ir technologijų analizė

|  |  |
| --- | --- |
|  |  |
| **IFM 2/2 gr. Tadas Laurinaitis**  Studentas / Studentė | (parašas) (data) |
|  |  |
| **Dr. Tomas Blažauskas**  Projekto vadovas | (parašas) (data) |
|  |  |
| **Lekt. Virginija Limanauskienė**  Dėstytoja | (parašas) (data) |
|  |  |

**Kaunas, 2022**

Turinys

[Paveikslėlių sąrašas 3](#_Toc123143692)

[Santrumpų ir terminų sąrašas 4](#_Toc123143693)

[Įvadas 5](#_Toc123143694)

[1. Tikslas 6](#_Toc123143695)

[2. Srities apžvalga ir egzistuojantys sprendimai 7](#_Toc123143696)

[3. Egzistuojančių rinkoje strateginiu žaidimų dirbtinio intelekto naudojimo analizė 14](#_Toc123143697)

[4. Išvados 18](#_Toc123143698)

[5. Literatūros sąrašas 19](#_Toc123143699)

Paveikslėlių sąrašas

[pav. 1 Deterministinio dirbtinio intelekto veikimo principas 8](#_Toc123143051)

[pav. 2 Personažo gyvybės atvaizduotos fuzzy grafu [10] 10](#_Toc123143052)

[pav. 3 Sprendimų medžio pavyzdys [14] 11](#_Toc123143053)

[pav. 4 Konvoliucinis neuroninis tinklas mokosi žaisti žaidimą „*Mario“* [16] 12](#_Toc123143054)

[pav. 5 Strateginio žaidimo „Sid Meier‘s Civilization V“ pagrindinio žaidimo lango vaizdas [17] 14](#_Toc123143055)

[pav. 6 Elementari Blackboard sistemos schema [21] 15](#_Toc123143056)

[pav. 7 Žaidimo „*Age of Empires III*“ pagrindinio žaidimo lango vaizdas, kuriame matomi skirtingi pastatai bei padaliniai [22]. 16](#_Toc123143057)

[pav. 8 Primityvus valdymo elgsenų medis su dviem šakomis [25] 17](#_Toc123143058)

Santrumpų ir terminų sąrašas

**Santrumpos:**

NPC (angl. *Non-playable character*) – kompiuterinio žaidimo veikėjas, kurį valdo kompiuteris.

AI (angl. *Artificial Intelligence*) – dirbtinis intelektas.

Įvadas

Dokumentas yra Programų sistemų inžinerijos magistrantūros disciplinos „Tiriamasis projektas 1“ ataskaita. Dokumento paskirtis apibūdinti tyrimo tikslus, apibendrinti atliktą literatūros analizę, pasirengti projekto reikalavimų specifikavimui, projektavimui, susipažinti su užsakymo taikymo sritimi, pasauliniais pasiekimais taikomojoje srityje.

Projektas – Dirbtinio intelekto metodų taikymas strateginiams žaidimams.

Dirbtinis intelektas padeda sukurti labiau reaguojančius, prisitaikančius ir sudėtingesnius strateginius žaidimus. Dirbtinio intelekto dėka, strateginiai žaidimai sukuria geresnę patirtį savo žaidėjams kurdami gyvenimiškus situacinius pokyčius žaidimo progrese bei suteikdami jaudulio jausmą [1]. Didėjantis strateginių žaidimų dirbtinio intelekto sudėtingumas užtikrina, kad žaidėjai labiau įsijaus į patį žaidimą, o pats žaidimas jiems greitai neatsibos. Šiuo metu yra daug strateginių žaidimų, kuriuose taikomi įvairūs algoritmai priešininkų ar sąjungininkų elgsenos modeliavimui, kelio radimui, duomenų gavybai, procedūriniam turinio generavimui ir žaidėjo patirties modeliavimui. Bėda ta, kad daugelis sukurtų algoritmų yra apsaugoti juos kuriančių kompanijų ar korporacijų ir nėra viešai prieinami.

Raktiniai žodžiai – Dirbtinis intelektas, strateginius žaidimus, strateginiai žaidimai, žaidėjams, dirbtinio intelekto sudėtingumas, modeliavimui, algoritmų.

# Tikslas

Šio darbo tikslas yra ištirti esamus dirbtinio intelekto algoritmus ir metodikas naudojamas kompiuterinių žaidimų strategijai realizuoti, pasiūlyti patobulinimus šiems algoritmams ir metodikoms ir įvertinti jų efektyvumą sukuriant algoritmus ir metodikas realizuojantį žaidimą.

# Srities apžvalga ir egzistuojantys sprendimai

Strateginiai žaidimai yra ideali aplinka dirbtinio intelekto galimybių tyrinėjimui, kadangi bendrai, žaidimai suteikia uždarą aplinką su tam tikromis, kiekvienam žaidimui specifinėmis taisyklėmis, kurioje gali būti išbandoma ir įvertinama plati aibė problemų sprendimų būdų ir technikų, prieš pritaikant šiuos būdus ir technikas sprendžiant realaus pasaulio problemas. Pirmas dirbtinio intelekto pritaikymas žaidime buvo 1951 metais sukurtame, matematiniame strateginiame žaidime „*Nim*“ [2], kurio tikslas buvo dviems žaidėjams vienam paskui kitą nuiminėti daiktus iš dviejų krūvų. Šiame žaidime dirbtinis intelektas sugebėdavo reguliariai laimėti prieš žmones. Tais pačiais metais Mančesterio universitete buvo sukurti du kompiuteriniai žaidimai, kurie taip pat naudojo dirbtinį intelektą – šaškės ir šachmatai. Per ateinančius dešimtmečius dirbtinio intelekto algoritmai naudojami šiuose žaidimuose buvo tobulinami, o šio tobulinimo kulminacija buvo pasiekta 1997 metais, kai *IBM Deep Blue* kompiuteris nugalėjo tuometinį šachmatų „*Grandmaster*“ titulą laikantį Garry Kasparov [3]. Dar praeito amžiaus aštuntajame dešimtmetyje dirbtinis intelektas tapo integralia kompiuterinių žaidimų dizaino dalimi. 1978 metais sukurtas „*Space Invaders*“ demonstravo vis sunkėjančius lygius su skirtingais judėjimo modeliais, priklausančiais nuo žaidėjo judėjimo, naudojantis elementariu dirbtiniu intelektu saugomų judesių modelių pavidalu [4]. Nuo to laiko dirbtinis intelektas buvo bandomas pritaikyti daugumos žanrų žaidimams. Dirbtinis intelektas sporto žaidimuose pasirodė dar nuo praeito amžiaus devinto dešimtmečio, išleidus tokius žaidimus, kaip „*Earl Weaver Baseball*“, „*Madden Football*“ ir „*Tony La Russa Baseball*“, kuriuose priešininkai bandė atkartoti realių trenerių ir vadovų vadovavimo stilius ir naudojamas taktikas, o vėlesniuose sporto žaidimuose netgi atsirado galimybės sukurti dirbtinio intelekto priešininkus naudojant žaidėjo nustatytus kintamuosius. Formalūs dirbtinio intelekto įrankiai, tokie kaip baigtinių būsenų mašinos, buvo pradedami naudoti dar 1990-taisiais įvairiuose, ypač naujuose žaidimų žanruose. Dirbtinis intelektas buvo pradedamas naudoti realaus laiko strateginiuose žaidimuose kelio radimo problemoms spręsti, atlikti realaus laiko sprendimus ar netgi planuoti kompiuterinio priešininko ekonomiką. Tačiau pirmųjų žaidimų dirbtinis intelektas nebuvo tobulas – pvz. 1990 Žaidimo „*Herzog Zwei*“ kelio radimas buvo praktiškai neveiksnus ir naudojo paprastą trijų baigtinių būsenų mašiną padalinių kontrolei, o 1992 žaidimo „*Dune II*“ dirbtinis intelektas sukčiaudavo, kad gautų nesąžiningą persvarą prieš žaidėją. Laikui bėgant, o dirbtinio intelekto metodams ir algoritmams tampant labiau rafinuotiems, didžioji dalis šių problemų buvo išspręsta, o vėlesni žaidimai pradėjo naudoti „*bottom-up*“ (liet. Iš apačios į viršų) dirbtinio intelekto metodus, kurie sugeba įvertinti žaidėjo veiksmus ir pagal juos nusprendžia tolesnius veiksmus [5]. Šiuo metu, kompiuteriniuose žaidimuose, o tuo pačiu ir strateginiuose žaidimuose naudojamas dirbtinis intelektas dažniausiai yra skirstomas į dvi rūšis – deterministinį (angl. *Deterministic*) ir nedeterministinį (angl. *Nondeterministic*).

## Deterministinis dirbtinis intelektas

Diagram

Description automatically generated

pav. 1 Deterministinio dirbtinio intelekto veikimo principas

Deterministinio dirbtinio intelekto elgesys ir veikimas yra konkretus ir nuspėjamas, nėra neaiškumo. Deterministiniai dirbtinio intelekto metodai yra kompiuterinių žaidimų duona. Šie algoritmai yra nuspėjami, greiti, lengvai implementuojami, lengvai suprantami ir lengvai ištestuojami. Deterministiniai metodai nors ir naudingi, tačiau ant žaidimo kūrėjo užkrauna naštą – žaidimo kūrėjas turi žinoti, numatyti ir implementuoti visus įmanomus veiksmus. Taip pat, deterministiniai metodai neapima mokymosi ir tobulėjimo, todėl žaidėjui šiek tiek pažaidus žaidimą, jie tampa nuspėjami, ko pasekoje sutrumpėja žaidimo gyvavimo ciklas. Paprastas deterministinio intelekto pavyzdys yra persekiojimo algoritmas. Parašytas algoritmas nurodytų personažui, kad jam reikia judėti link tikslinio taško x ir y koordinačių ašimis, kol personažo x ir y koordinatės sutaps su tiksline vieta [6]. 1 pav. matoma paprasta deterministinio dirbtinio intelekto reprezentacija, kur parodoma, jog iš Q1 galimi perėjimai į Q2 ir Q3, priklausomai nuo to ar Q1 reikšmė yra 0 ar 1.

## Nedeterministinis dirbtinis intelektas

Priešingai nei deterministiniuose dirbtinio intelekto metoduose, nedeterministiniuose metoduose yra tam tikras neaiškumo laipsnis, o elgesys ir veikimas yra nenuspėjamas (nenuspėjamumo laipsnis labai priklauso nuo panaudoto dirbtinio intelekto metodo ir jo supratimo lygio). Nedeterministiniai metodai taip pat gali mokytis ir ekstrapoliuoti patys, skatinti atsirandantį naują elgesį, arba elgesį, kuris atsiranda be aiškių nurodymų. Žaidimų kūrėjai tradiciškai žiūrėdavo į nedeterministinius metodus kaip į nepatikimus, nors paskutiniais metais šis požiūris pradėjo keistis. Panaudojus nedeterministinį metodą jį sunku ištestuoti ir užtikrinti, jog neatsiras nenorima elgsena, todėl esant trumpam žaidimo kūrimo ciklui, sunku garantuoti jog žaidimas bus pilnai ištestuotas ir atitiks kokybės standartus, kurių tikisi potencialūs žaidėjai. Puikus nedeterministinio metodo pavyzdys būtų kompiuterio valdomo priešininko mokymasis ir prisitaikymas prie žaidėjo kovos taktikų, o tokį prisitaikymą galima būtų pasiekti naudojantis neuroniniu tinklu, Bayeso technika arba genetiniu algoritmu [7].

## Nusistovėjęs dirbtinis intelektas žaidimuose

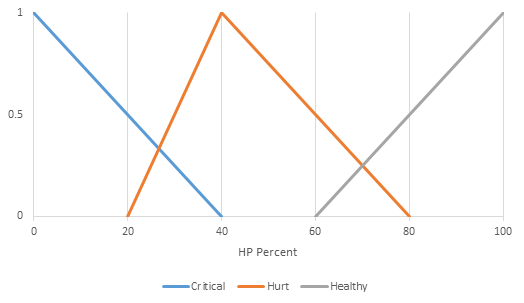
Dažniausiai žaidimuose suteikti dirbtiniam intelektui žmogišką intelektą nėra pagrindinis kurėjų tikslas. Dažnu atveju rašomas kodas būna skirtas kontroliuoti personažą, kuris nėra žmogus, kaip pvz. robotas, drakonas ar katė. Nors dažnu atveju, dirbtinis intelektas žaidimuose pasitelkiamas išspręsti įvairias sudėtingas problemas, kartais kuriami veikėjai, kurie nepasižymi labai protingomis tendencijomis taip pat yra tikslas, nes tai duoda žaidimui turiniui įvairovės. Taip pat dirbtinis intelektas pasitelkiamas kaip įrankis duoti ne žaidėjo valdomiems personažams skirtingas asmenybes, emocijas ar polinkius – pavaizduoti juos laimingus, išsigandusius, suirzusius ir t.t.

Dar viena iš labiausiai paplitusių dirbtinio intelekto metodikų žaidimuose yra sukčiavimas. Sukčiavimas yra labai įprastas būdas, padedantis kompiuterio valdomam žaidėjui suteikti pranašumą prieš protingus tikrus žaidėjus. Ši metodika pasireiškia įvairiausiais būdais, tačiau vienas iš pavyzdžių būtų kariniame strateginiame žaidime kompiuterio valdomam žaidėjui visos informacijos susijusios su tikru žaidėju suteikimas – tikro žaidėjo bazės vieta, valdomų personažų vietos ir kiekiai, jų tipai ir t.t. Tačiau kompiuterio valdomo veikėjo sukčiavimas gali būti ir blogas dalykas – jeigu tikram žaidėjui yra aišku, jog kompiuteris sukčiauja, tikram žaidėjui taps nuobodu ir jam atrodys, jog visos jo dedamos pastangos yra beprasmės. Taip pat nesubalansuotas kompiuterio sukčiavimas gali kompiuterio valdomam žaidėjui suteikti per daug galios, ko pasekoje žaidimas taps neįmanomas, todėl ir šiuo, ir praeitu atveju, kompiuterio valdomo žaidėjo sukčiavimas turi būti subalansuotas taip, kad tikram žaidėjui būtų sudaromas toks iššūkis, kad žaidimas atrodytų įdomus ir smagus [8].

## Taikomi metodai, jų veikimo principai ir apribojimai

Šiuolaikiniuose žaidimuose naudojamas platus spektras dirbtinio intelekto metodų - paprastos baigtinių būsenų mašinos ir *fuzzy* logika, įvairūs kelio radimo algoritmai, ekspertų sistemos, sprendimų medžiai, gilūs neuroniniai tinklai, genetiniai algoritmai, paskatinimo principu veikiantys mokymosi algoritmai ir daug kitų. Šiame poskyryje apžvelgsime viena deterministinį metodą – fuzzy logika ir du nedeterministinius metodus – sprendimų medžius ir giliuosius neurononinius tinklus.

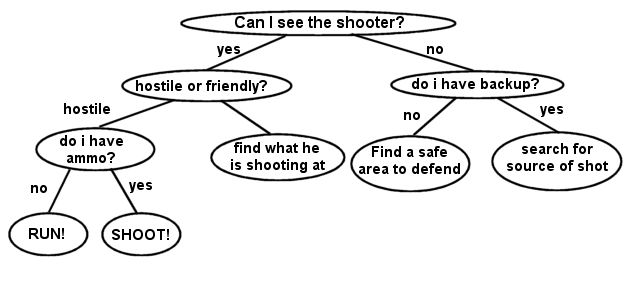
### Fuzzy logika



pav. 2 Personažo gyvybės atvaizduotos fuzzy grafu [10]

*Fuzzy* logika yra įprastinės logikos plėtinys, kur objekto priklausomybės laipsnis aibėje nėra apibrėžtas sveikaisiais skaičiais 0 (objektas aibei nepriklauso) ir 1 (objektas aibei priklauso), bet gali būti bet kuris skaičius iš intervalo nuo 0 iki 1 [9]. *Fuzzy* logika dažniausiai yra naudojama *fuzzy* sistemos forma, o ją sudaro fuzzy kintamieji, *fuzzy* taisyklės ir *fuzzy* išvadų variklis. Naudojantis *fuzzy* logika, kintamieji yra paverčiami į *fuzzy* kintamuosius naudojantis narystės funkcijų rinkinį pagal galimą jų verčių diapazoną. Narystės funkcijos priskiria aiškią kintamojo reikšmę *fuzzy* etiketei su tiesos laipsnius, kuris dažniausiai yra tarp 0 ir 1. Pavyzdžiui, kaip matome 2 pav., personažo gyvybės žaidime gali būti padalintos į tris dalis – *Critical*, *Hurt* ir *Healthy*, kaip matoma paveikslėlyje aukščiau. Kintamieji gali būti traktuojami kaip įvestis, pavyzdžiui personažo parametrai, arba išvestis. *Fuzzy* taisyklės gali būti aprašomos naudojantis aprašytais *fuzzy* kintamaisiais ar jų aibe. Taisyklės dažniausiai yra jeigu – tai forma, o jų gramatika panaši į *boolean* logiką. Po to kai taisyklės yra aprašytos, *fuzzy* išvadų variklis naudojamas, kad gauti išvadas apie esamų kintamųjų reikšmes pagal sukurtas taisykles, ko pasekoje išvesties kintamiesiems suteikiamos *fuzzy* reikšmės. Galiausiai, *fuzzy* kintamieji gali būti atverčiami atgal į paprastus kintamuosius. Kaip ir dauguma kitų dirbtinio intelekto technikų, *fuzzy* logika buvo testuojama žaidimuose, norint sukurti paprastą dizainą, su kombinuotą su inteligentiškais agentais. *Fuzzy* logika patenkina šiuos reikalavimus, kadangi ji yra pakankamai paprasta naudoti, primena tikrą kalbą ir turi patogią ir greitą dizaino metodologiją. *Fuzzy* logika gali būti naudojama žaidimo kompiuterio valdomų veikėjų sprendimų priėmimo procese, kaip pavyzdžiui daikto ar ginklo pasirinkimas arba kitų veikėjų valdymas, tai pat rizikų vertinimui. *Fuzzy* logikos naudojimas žaidimų kūrėjams atneša daug teigiamų dalykų, kadangi jos naudojimas reikalauji tik paprastos logikos supratimo, todėl tiek *fuzzy* taisyklės, tiek *fuzzy* kintamieji gali būti kuriami visų, dalyvaujančių žaidimo kūrimo procese. Tradicinis kompiuterio valdomų žaidėjų sprendimų priėmimas gali būti kiek per daug staigus naudojant tradicinius metodus, todėl *fuzzy* yra puikus kelias padaryti šiuos sprendimus labiau nuosekliais. Tačiau kaip ir visa kita, *fuzzy* logika nėra tobula – ji reikalauja visiškai teisingų įvesties ir išvesties kintamųjų ribų nustatymo, o sistemose su šimtais skirtingų žaidimo agentų veikiančių vienu metu, gali suprastinti žaidimo veikimą, kadangi šimtai taisyklių patikrinimų veiks kiekvieną sekundę [11].

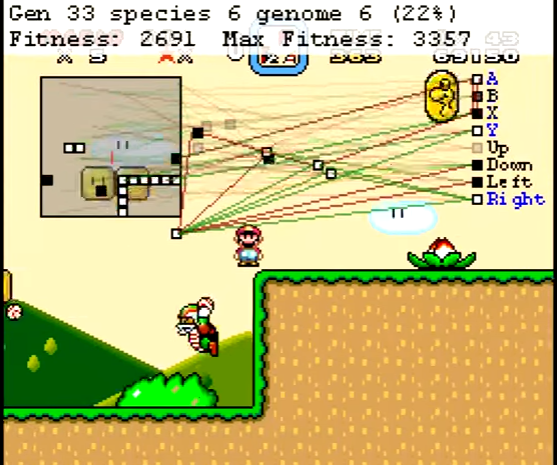
### Sprendimų medžiai



pav. 3 Sprendimų medžio pavyzdys [14]

Sprendimų medžiai, tai prižiūrimo mašininio mokymosi metodai, kurie gali būti apmokomi atlikti klasifikaciją ir regresiją. Tai yra vieni iš paprasčiausių mašininio mokymosi algoritmų, naudojamų žaidimų kūrime. Jie padeda įvertinti kintamųjų reikšmių svarbą, nustatydami tam tikras pasirinkimų taisykles atsižvelgdami į duomenų charakteristikas. Sprendimų medžiai yra lengvai suprantami ir interpretuojami, o pasirinkimai neturėtų būti sunkiai įvertinami. Žaidimų dizaino stadijoje, naudojamos sprendimų lentelės [13]. Sprendimų medžiai yra sudaryti iš šakninio lapo, kuris dažniausiai yra situacija, kurioje žaidėjas/veikėjas yra iš pradžių, ir lapų, kurie reprezentuoja kiekvieną tolimesnį pasirinkimą. Kiekvienas lapas dažniausiai turi ribotą kiekį lapų, arba jų neturi išvis. Sprendimų medžiu yra laikoma, kai daugiau nei vienas pasirinkimas yra duodamas tikram žaidėjui ir kai žaidėjas kažkurį iš jų pasirenka, jam parodoma dar pasirinkimų, kurie susiję su praeitu jo pasirinkimu. Sprendimų medžiai leidžia daug skirtingų pasirinkimų ir jų kombinacijų, iš kurių kiekviena yra unikali ir atskirta viena nuo kitos. Sprendimų medžiai leidžia žaidėjui susikurti savo istoriją ar išeigas priklausomai nuo žaidėjo padarytų pasirinkimų. Vienintelis skirtumas tarp sprendimų medžių ir visiškos laisvės žaidėjui yra tai, kad pasirinkimų rezultatai yra iš anksto žinomi. Modernūs kompiuteriniai žaidimai dažniausiai turi sprendimų medžius, apie kuriuos žaidėjas nieko nežino. Kompiuteriniai žaidimai dažniausiai turi tam tikras užuominas, kurios priveda prie tam tikro rezultato ir pasako žaidėjui koks tas rezultatas bus dar prieš padarant pasirinkimą. Sprendimų medžiai duoda žaidėjui įžvalgų, kas jos laukia žaidime, bei duoda užuominų kuris kelias yra teisingas. Pavyzdys – žaidime „*Star Wars Jedi: Fallen Order*“ žaidėjui žaidžiant žaidimą duodamos įžvalgos, kokia būtų ateitis, jeigu tam tikros aplinkybės susiklostytų. Taip pat, žaidime „*The Witcher 3: Wild Hunt*“, žaidimas turi skirtingas pabaigas priklausomai nuo žaidėjo padarytų veiksmų – įvykdytų misijų, neskaitant pagrindinės misijos, kiekio ir pasirinkimų, padarytų vykdant misijas [12]. 3 pav. matomas paslėptas sprendimų medis, kuriuo kompiuterio valdomas personažas gali priimti tam tikrus sprendimus, priklausomai nuo medžio šakos, kurioje jis šiuo metu yra, lygio.

### Gilieji neuroniniai tinklai



pav. 4 Konvoliucinis neuroninis tinklas mokosi žaisti žaidimą „*Mario“* [16]

Dirbtiniai neuroniniai tinklai yra algoritmai, kurių struktūra yra šiek tiek panaši į žmogaus smegenų struktūrą, ir kurie gali išmokti įvairias charakteristikas iš apmokymui skirtų duomenų ir modeliuoti net labai sudėtingas tikro pasaulio ar kompiuterinio žaidimo situacijas. Palyginus su klasikiniais dirbtinio intelekto metodais, neuroniniai tinklai gali pereiti tam tikras spragas žaidimo dizaine. Neuroniniai tinklai yra prisitaikantys ir sugeba pasitaisyti priklausomai nuo besikeičiančių žaidimo sąlygų realiu laiku. Atitinkami mokymo metodai gali būti naudojami priklausomai nuo to kokį žaidimą ir kokį neuroninį tinklą žaidimo kūrėjas bando realizuoti. Dirbtinio intelekto agentai, naudojantys giliuosius neuroninius tinklus strateginiuose žaidimuose gali greitai pakeisti savo žaidimo strategijas, kad neatsiliktų nuo tikrų žaidėjų, kadangi jie turi savybę mokytis ir tobulėti. Gilieji neuroniniai tinklai yra puikus būdas užtikrinti, kad žaidimas išliks sunkus netgi po ilgo žaidimo laiko. Gilieji neuroniniai tinklai šiuo metu yra labai populiarūs ir paklausūs kaip žaidimo agentų dizaino įrankis. Gilieji neuroniniai tinklai naudoja kelis neuroninių tinklų sluoksnius, kurie naudojantis žaidimo suteikiamais įvesties duomenimis tobulėja ir mokosi žaisti. Kontroliuodami vieną ar kelis žaidimo agentus, gilieji neuroniniai tinklai pasiekia geresnius rezultatus kompiuterio kontroliuojamo žaidėjo rezultatų pasiekime, nei kiti dirbtinio intelekto metodai, kas leidžia jiems būti tiek kompiuterio kontroliuojamais žaidėjais, tiek žaidimo aplinkos valdytojais [13]. Gilieji neuroniniai tinklai mokantis gali nekreipti dėmesio į nereikšmingus ir tik žmogui skirtus veiksnius – oro sąlygas žaidime, besikeičiančias spalvas grafinėje sąsajoje ar aplinkoje, todėl gali greitai pasiekti įspūdingus rezultatus [15]. 4 pav. matomas konvoliucinis neuroninis tinklas, kuris mokosi žaisti žaidimą „*Mario*“, naudodamasis valdymo komandomis kaip įvesties duomenimis, ir artėjimu prie tikslo kaip paskatinimu.

# Egzistuojančių rinkoje strateginiu žaidimų dirbtinio intelekto naudojimo analizė

Šiuo metu rinkoje egzistuoja labai didelis kiekis įvairiausio tipo strateginių žaidimų – nuo paprasčiausių žaidimų, tokių kaip šaškės, iki kompleksiškų, iš ne vienos dirbtinio intelekto sistemos sudarytų grandiozinės strategijos žaidimų, tokių kaip „*Sid Meier‘s Civilization V*“ ar „*Crusader Kings II*“. Žaidimai naudoja dirbtinio intelekto metodus, kad padidintų žaidimo patrauklumą potencialiems žaidėjams, kad sukeltų tikro, „gyvo“ ir protingo priešininko jausmą ir kad užtikrintų, jog žaidimą galima būtų pereiti ne vieną kartą. Šiam tikslui pasiekti, žaidimų kūrėjai ieško vis naujesnių ir labiau priimtinų dirbtinio intelekto metodų, tačiau ne visada renkasi patį naujausią, geresnį, tačiau mažiau ištirtą ir ištestuotą metodą, kadangi tai padidina kuriamo žaidimo tiek piniginius tiek laiko kaštus. Dažnu atveju liekama prie galbūt senesnių, tačiau labiau ištirtų ir laiko patikrintų metodų, kadangi jie siūlo labiau nuspėjamą kompiuterinių žaidimų kūrimo procesą ir lengviau nuspėjamus tiek piniginius tiek laiko kaštus. Šiame skyriuje atlikta dviejų pasirinktų, vis dar aktualių ir populiarių strateginių žaidimų ir juose naudojamų dirbtinio intelekto metodų analizė – „*Sid Meier‘s Civilization V*“ ir „*Age of Empires III*“.

## Sid Meier‘s Civilization V



pav. 5 Strateginio žaidimo „Sid Meier‘s Civilization V“ pagrindinio žaidimo lango vaizdas [17]

„*Sid Meier‘s Civilization V*“ yra strateginis kompiuterinis žaidimas, sukurtas *Firaxis Games*, kombinuojantis tiek ėjimais grįstą žaidimo principą, tiek realaus laiko strategiją. Žaidimo tikslas yra nuo senų iki naujų laikų pastatyti savo civilizaciją ir pasiekti vieną iš galimų pergalių tipų naudojantis mokslu, plėtimusi, diplomatija, tyrinėjimu, ekonominiu vystymusi arba kariuomene. Žaidėjas valdo vieną iš 43 galimų visų laikų civilizacijų, ir rungtyniauja su kompiuterio valdomomis civilizacijomis arba kitais tikrais žaidėjais tinklo rėžimu. Žaidime figūruoja skirtingos eros, kurios eina žaidėjui tobulinant savo civilizaciją. Žaidimas pastatytas naudojantis nauju žaidimų varikliu, ir naudoja šešiakampius laukelius žaidimo žemėlapio objektams [18]. Žaidimo naudojamas dirbtinis intelektas susideda iš kelių skirtingų dalių, kurių visuma vadinasi *Blackboard* sistema [19]. Kartu su *Blackboard* sistema, žaidimas naudoja sukčiavimą (2.3 poskyris šiame dokumente). Šios sistemos dalys sudaro *Blackboard* visumą:

1. Karo dirbtinis intelektas – atsakingas už kareivių valdymą, naujų kareivių paruošimą, kareivių prioriteto nustatymą, skirtingų kareivių tipų parinkimą atitinkamiems veiksmams.
2. Prekiavimo dirbtinis intelektas – atsakingas už prekiavimą su kitais kompiuterio ir tikrais žaidėjais, resursų vystymą, laukelių gerinimą.
3. Mokslo dirbtinis intelektas – atsakingas už mokslo civilizacijoje plėtojimą, mokslinės šakos pasirinkimus.
4. Tyrinėjimo dirbtinis intelektas – atsakingas už žemėlapio naršymą, žvalgų siuntimą.

### Blackboard sistemos žaidimuose

*Blackboard* sistema yra vienas iš dirbtinio intelekto realizavimo žaidimuose būdų. Tai metodas, kuris susideda iš žinomų faktų bazės, dar kitaip vadinamos lenta, kuri yra iteratyviai atnaujinama grupės žinių šaltinių, pradedant nuo problemos specifikacijos ir baigiant problemos sprendimu. Kiekvienas iš šaltinių atnaujina lentoje esamą informaciją galimais ne pilnais sprendimais, kai žinių šaltinio vidiniai apribojimai pasiekia lentos būseną [20]. *Blackboard* sistema padeda atsieti skirtingas funkcijas, taip sukuriant solidžią architektūrą su lengva implementacija ir palaikymu.

Diagram

Description automatically generated

pav. 6 Elementari Blackboard sistemos schema [21]

*Blackboard* sistema yra žinučių siuntimo ir duomenų sistema. Žaidimo metu, užkraunamos lentos, kurios turi sąrašą duomenų, nuo kurių priklausomai iškviečiami įvykiai, kuriuos lentą stebinčios sistemos gali panaudoti keičiant duomenis. Lentoje kiekviena eilutė turi unikalų Id, kuris naudojamas prieiti prie atitinkamų duomenų arba iškviesti atitinkamą įvykį. Lenta laiko duomenis atskirais kintamaisiais, o jie patys yra indeksuojami, kartu su jų raktais, todėl lenta veikia kaip kontroleris. Nors *Blackboard* sistema yra puikus būdas atsieti funkcijas ir pagreitinti skirtingų dirbtinio intelekto sistemų valdymą, tai nėra magiškas atsakymas į visus žaidimo reikalavimus ar problemas [21].

## Age of Empires III

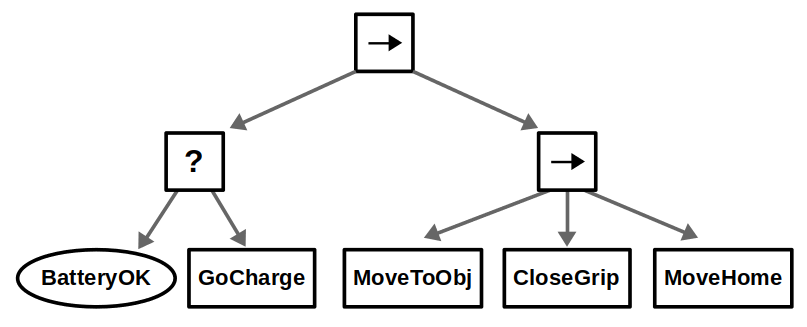


pav. 7 Žaidimo „*Age of Empires III*“ pagrindinio žaidimo lango vaizdas, kuriame matomi skirtingi pastatai bei padaliniai [22].

„*Age of Empires III*“ yra realaus laiko strateginis kompiuterinis žaidimas sukurtas *Microsoft*. Žaidime vaizduojama Amerikų kolonizacija, o žaidėjas gali pasirinkti vieną iš 14 civilizacijų už kurias jis gali žaisti. Žaidimas atvedė į strateginių žaidimų žanrą, tokių kaip pagrindinis miestas ir puikiai kombinuoja realaus laiko strategiją su rolių žaidimo atributais. Žaidėjo tikslas pasiekti pergalę prieš kompiuterio valdomus arba realius žaidėjus, statant savo miestą, išgaunant resursus, apmokant kareivius ir statant pastatus, įtvirtinimus ir vystant ekonomiką. Žaidimas suskirstytas į įvairius amžius, kurie eina žaidėjui tobulėjant ir tobulinant savo miestą [23]. Dirbtinis intelektas žaidime valdo kompiuterio valdomus žaidėjus, jų įsakymus ir priešų ir draugų padalinių sąveiką. Dirbtinis intelektas leidžia kompiuterio žaidėjams turėti skirtingus prioritetus, statyti įmantrias gynybines bazes ir jas pastoviai gerinti. Dirbtinis intelektas kompiuterio žaidėjams taip pat suteikia skirtingas taktikas ir požiūrius, priklausomai nuo to, už kurią civilizaciją jis žaidžia [24]. Žaidimo dirbtinis intelektas pagrinde naudoja elgsenų medžius įkomponuotus kartu su sprendimų medžiais (2.4.2 poskyris šiame dokumente) bei sukčiavimą (2.3 poskyris šiame dokumente).

### Elgsenų medžiai

Elgsenų medžiai šiuo metu yra dominuojanti technika, naudojama žaidimų kūrėjų kaip dirbtinio intelekto metodas jų kuriamuose žaidimuose. Ši technika pagrinde buvo sukurta kaip patobulinimas baigtinių būsenų mašinoms, kadangi šių pagrindinis trūkumas yra, jog esant šimtams būsenų, baigtinių būsenų mašinos tampa vis sunkiau ir sunkiau testuojamos. Pagrindinė idėja – elgsenos yra vykdomos arba po vieną, arba kelios paraleliai, o sprendimas, kurią elgseną vykdyti, atliekamas kas tam tikrą laiko tarpą, pavyzdžiui kas sekundę. Elgsenų medžiai taip pat leidžiam pakeisti elgsenų vykdymo seką, jeigu paskutinę sekundę buvo priimtas sprendimas vykdyti elgseną, kurios nėra sekoje. Elgsenų medžiai suteikia karkasą lengviau suprantamoms ir lengviau skaitomoms dirbtinio intelekto sistemoms, o vizualiai reprezentuoti medžiai leidžia lengvai juos testuoti [25]. Tačiau kaip ir visa kita, elgsenų medžiai turi savo trūkumų – labai dideli elgsenų medžiai turi labai didelius vertinimų kaštus žaidimo veikimo greičiui, ir nors elgsenų medžiai yra geras metodas elgsenų organizavimui, jie nesuteikia dirbtiniam intelektui labiau optimizuoto sprendimų atlikimo metodo.



pav. 8 Primityvus valdymo elgsenų medis su dviem šakomis [25]

# Išvados

1. Buvo atlikta įvairių literatūros šaltinių, susijusių su magistrinio darbo tema, analizė.
2. Išanalizavus dirbtinio intelekto naudojimą kompiuteriniuose žaidimuose pastebėtas ryški pažanga šioje srityje – per paskutinius 70 metų dirbtinis intelektas kompiuteriniuose žaidimuose patobulėjo nuo paprastų statinių metodų iki sudėtingų neuroninių tinklų ir tapo neatskiriama kompiuterinių žaidimų dalimi.
3. Pastebėta, kad dirbtinis intelektas kompiuteriniuose žaidimuose išskiriamas į dvi rūšis, o šios rūšys turi daug skirtingų porūšių, kurie tiek pavieniai, tiek kartu su kitais naudojami kompiuteriniuose žaidimuose.
4. Atliekant esamų sprendimų analizę, išsiaiškinta, jog tiek populiarūs, tiek nelabai populiarūs žaidimai naudoja dirbtinį intelektą vienokia ar kitokia forma ir padeda kompiuterio valdomiems žaidėjams imituoti tikrus žaidėjus ir tikriems žaidėjams užtikrinti įdomiai praleistą laiką.
5. Pastebėta tendencija, kad didžioji dauguma net ir nesenai išleistų kompiuterinių žaidimų vis dar naudoja kiek pasenusius, tačiau per laiką išdirbtus dirbtinio intelekto metodus, o tik maža dalis naujų kompiuterinių žaidimų bando įtraukti sudėtingesnius ir ne tiek išdirbtus dirbtinio intelekto metodus, dėl augančios žaidimo kainos ir testavimo laiko.

# Literatūros sąrašas

[1] J. Dsouza, „*AI in Gaming*“ [Tinkle]. Nuoroda: <https://www.engati.com/blog/ai-in-gaming> [kreiptasi 26 11 2022]

[2] „*History of AI Use in Video Game Design*“ [Tinkle]. Nuoroda: <https://bigdataanalyticsnews.com/history-of-artificial-intelligence-in-video-games/> [kreiptasi 18 11 2022]

[3] A. Schulz, „*25 years ago: Deep Blue beats Kasparov*“ [Tinkle]. Nuoroda: <https://en.chessbase.com/post/25-years-ago-deep-blue-beats-kasparov> [kreiptasi 26 11 2022]

[4] S. Rizzo „*Imagined Thinking Machines: Artificial Intelligence in Video Games*“, Full Sail University, bal. 2, 2015 [Tinkle]. Nuoroda: <https://hub.fullsail.edu/articles/imagined-thinking-machines-artificial-intelligence-in-video-games> [Kreiptasi 26 11 2022]

[5] M. Trends, „*The Evolution of Artificial Intelligence in Video Games*“, Analytics Insight, 12 29 2020 [Tinkle]. Nuoroda: <https://www.analyticsinsight.net/the-evolution-of-artificial-intelligence-in-video-games/> [Kreiptasi 26 11 2022]

[6] G. Seeman, D. M. Bourg „*Chapter 1. Introduction to Game AI*“, OREILLY [Tinkle]. Nuoroda: <https://www.oreilly.com/library/view/ai-for-game/0596005555/ch01.html#:~:text=Game%20AI%20techniques%20generally%20come%20in%20two%20flavors%3A%20deterministic%20and%20nondeterministic.&text=Deterministic%20behavior%20or%20performance%20is,is%20a%20simple%20chasing%20algorithm>. [Kreiptasi 26 11 2022]

[7] S. Pathak, „*Artificial Intelligence and Machine Learning in Game*“, Medium, 11 21 2021 [Tinkle]. Nuoroda: <https://medium.com/geekculture/artificial-intelligence-and-machine-learning-in-game-33ae7310e30a> [Kreiptasi 26 11 2022]

[8] N. Ahamed, „*AI Cheating in Games*“, Medium, 09 30 2022 [Tinkle]. Nuoroda: <https://medium.com/mlearning-ai/ai-cheating-in-games-583e333677ce#:~:text=Cheating%20in%20the%20context%20of,have%20in%20the%20same%20circumstance>. [Kreiptasi 26 11 2022]

[9] G. Valiulis, „Skysčio lygio valdymas taikant *fuzzy* logiką“, 09 02 2006 [Tinkle]. Nuoroda: <http://www.elektronika.lt/straipsniai/elektronika/3168/skyscio-lygio-valdymas-taikant-fuzzy-logika/> [Kreiptasi 27 12 2022]

[10] K. Chaudhari, „*Implementing fuzzy logic to bring AI characters alive in Unity based 3D games*“, Packt hub, 01 06 2018 [Tinkle]. Nuoroda: https://hub.packtpub.com/fuzzy-logic-ai-characters-unity-3d-games/

[11] M. Pirovano, „*The use of Fuzzy Logic for Artificial Intelligence in Games*“, University of Milano, 12 07 2012 [Tinkle]. Nuoroda: <http://www.michelepirovano.com/pdf/fuzzy_ai_in_games.pdf> [Kreiptasi 27 12 2022]

[12] B. Antonette, „*Decision Tress in Video Games*“, Medium, 11 12 2019 [Tinkle]. Nuoroda: <https://medium.com/@antoneb/decision-trees-in-video-games-3ea3f251f96e#:~:text=As%20mentioned%20earlier%20decision%20trees,not%20even%20be%20aware%20of>. [Kreiptasi 28 12 2022]

[13] E. Eliaçık, „*AI in gaming: A complete guide*“, Data Conomy, 28 09 2022 [Tinkle]. Nuoroda: <https://dataconomy.com/2022/04/artificial-intelligence-games/#Decision_trees> [Kreiptasi 28 12 2022]

[14] R. Penton, „*Trees Part 1*“, Gamedev [Tinkle]. Nuoroda: http://archive.gamedev.net/archive/reference/programming/features/trees1/page4.html [Kreiptasi 28 12 2022]

[15] L. Cross „*What Neural Networks Playing Video Games Teach Us About Our Own Brains*“, CalTech, 07 01 2021 [Tinkle]. Nuoroda: <https://www.caltech.edu/about/news/neural-networks-playing-video-games-teach-us-about-our-own-brains> [Kreiptasi 28 12 2022]

[16] „*Convolutional Network for Game*“ [Tinkle]. Nuoroda: <https://wiki.tum.de/display/lfdv/Convolutional+Neural+Network+for+Game> [Kreiptasi 28 12 2022]

[17] „*The Greatest Game Of Civilization V No One Played*“ [Tinkle]. Nuoroda: <https://civilization.com/civilization-5/news/entries/the-greatest-game-of-civilization-v-no-one-played/> [Kreiptasi 28 12 2022]

[18] Wikipedia – „*Civilization V*“ [Tinkle]. Nuoroda: <https://en.wikipedia.org/wiki/Civilization_V> [Kreiptasi 28 12 2022]

[19] Reddit – „*How exactly does Civ v‘s AI work?*“, 06 02 2016 [Tinkle]. Nuoroda: <https://www.reddit.com/r/civ/comments/44g8id/how_exactly_does_civ_vs_ai_work/> [Kreiptasi 28 12 2022]

[20] Wikipedia – „*Blackboard system*“ [Tinkle]. Nuoroda: <https://en.wikipedia.org/wiki/Blackboard_system#:~:text=A%20blackboard%20system%20is%20an,and%20ending%20with%20a%20solution>. [Kreiptasi 28 12 2022]

[21] B. Poli, „*Creating Decoupled Features: The Blackboard System*“, Game Developer, 19 06 2018 [Tinkle]. Nuoroda: <https://www.gamedeveloper.com/design/creating-decoupled-features-the-blackboard-system> [Kreiptasi 28 12 2022]

[22] M. Rorie, „*Age of Empires III Walkthrough*“, GameSpot, 25 01 2006 [Tinkle]. Nuoroda: https://www.gamespot.com/articles/age-of-empires-iii-walkthrough/1100-6136278/ [Kreiptasi 28 12 2022]

[23] Wikipedia – „*Age of Empires III*“ [Tinkle]. Nuoroda: <https://en.wikipedia.org/wiki/Age_of_Empires_III> [Kreiptasi 28 12 2022]

[24] Age Of Empires Fandom Wikipedia - „*Artificial Inteligence*“ [Tinkle]. Nuoroda: <https://ageofempires.fandom.com/wiki/Artificial_intelligence#Age_of_Empires_III> [Kreiptasi 28 12 2022]

[25] J. Rasmussen, „*Are Behavior Trees a Thing of the Past?*“, Game Developer, 27 04 2016 [Tinkle]. Nuoroda: <https://www.gamedeveloper.com/programming/are-behavior-trees-a-thing-of-the-past-> [Kreiptasi 28 12 2022]