Универзитет у Београду
МАТЕМАТИЧКИ ФАКУЛТЕТ

Мастер рад

Имплементација аутоматских стратегија у видео играма заснованих на еволутивним алгоритмима

Ментор

др Филип Марић

Кандидат

Драгослав Стојчић

1005/2012

Београд, мај 2015. година

У тренутку када се избрише линија између људи и машина,
линија између људи и Бога ће постати нејасна

Желео бих да се захвалим професорима Математичког факултета, Универзитета у Београду, на високом квалитету знања које преносе током година. Захваљујем се члановима комисије на сугестијама и помоћи при изради рада.

Посебно бих желео да се захвалим за подршку др Филипу Марићу. Такође бих желео да изразим захвалност својој породици и пријатељима на подршци и стрпљењу.

Сажетак

Тема овог рада је имплементација аутоматског одабира стратегија у видео играма, заснованог на еволутивним алгоритмима, као и могућност побољшања система вештачке интелигенције (ВИ) у комерцијалним рачунарским играма применом тих еволутивних алгоритама.

У раду је разматрано како се побољшања игре могу постићи применом техника машинског учења. Разматрају се појмови динамичног и статичног учења, као и технике статичног скриптовања. Фокус је на еволутивним техникама које доприносе побољшању ВИ против аутоматских противника, са посебним осврtом на ВИ у стратегијама у реалном времену (СРВ играма). Бирано је репрезентативно окружење и примењени су еволутивни алгоритми како би се, кренувши од задатог скупа полазних, открили нове тактике тј. стратегије.

За потребе експеримента одабрана је игра Warcraft III са WorldEditor-ом и Јавом као алатом, тј. језиком за експериментално истраживање. Дефинисана је функција прилагођености Ф, која свакој стратегији придружује вредност у опсегу [0,1]. Дизајнирана су два генетска оператора за еволуцију стратегије, укрштање и мутација, а као механизам селекције уведена је и употребљена турнирска селекција. Код поставке експеримента коришћене су статичне скрипте ВИ које су уграђене у самој игри. Величина популације током експеримента била је 20, а критеријум заустављања је постављен тако да се еволуција завршава када функција прилагођености Ф, за најбољу изграђену стратегију у тренутној популацији, достигне вредност 0,8. У случају да се не нађе такво решење, критеријум заустављања је постављен на 10 генерација. Свака стратегија се процењује тако што се унесе у игру која се онда покреће и одређује се коначни резултат. Ради лакшег тестирања, брзина протока игре је повећана у односу на нормалну брзину игре, тестеру је откривена мапа, а време партије је ограничено на сат времена, када се посматра време у игри. Ипак, време процене квалитета сваке стратегије је значајно, што представља један од основних проблема за спровођење експеримента са већим бројем генерација и јединки.

Резултати ефикасности експеримента представљени су табеларно. Приказани су ефекти примене оператора укрштања и мутације. Добијени резултати су показали да се одређене тактике побољшавају све док се не постигне одређени врхунац, а ефикасност затим креће да опада до одређене тачке, након чега почиње са поновним растом, тј. креће се око локалног максимума.

Може се претпоставити да су наведени резултати задовољавајућег квалитета, да примена еволутивних алгоритама доводи до побољшања квалитета стратегија, али да је даљи рад усмерен ка побољшању неизбежан.

Кључне речи: Вештачка интелигенција, еволутивни алгоритми, комерцијалне видео игрице, стратегије у реалном времену, машинско учење, интелигентне машине

Abstract

The theme of this work is the implementation of automatic strategy selection in video games based on evolutionary algorithms, as well, as the possibility of improving artificial intellignce (AI) in commercial computer games.

In this paper we discusse how improvements can be acheved by using machine learning techniques. We discusse the terms of dynamic and static learning, as well as techniques of static scripting. The focus is set on evolutionary techniques that contribute to improving the AI against other opponents, with special reference to the AI in RTS (real-time strategy) video games. A representative enviroment was selected and new tactics were discovered by using evolutionary algorithms.

For the experimental setup the folowing tools for the experimental research were selected: Warcraft III, WorldEditor and Java. Fitness function F, in range [0,1] was used. Two genetic operator were designed, mutation and state crossover, and tournament selection as the selection mechanism. For the testing purposes, the games static AI scripts were used. The population size was set to 20, and the stop criteria was set to 0.8 of the fitness function value. In case the fitness function doesn't reach the 0.8 score, a stop critera of 10 generations is set. The game speed was set to 25 times normal game speed, the map was revealed to the tester, and the playing time was set to a maximum of one hour game time.

The results of the experiment were presented in tabel form – crossover and mutations. The results have showed that some tactics get better until a certain maximum is reache, then their efficiency starts to fall to a certain point and then strart to rise again, in other words it reaches a local maximum.

It can be assumed that these results have a satisfactory quality and that they show that there is room for further research conserning AI improvement related to game.

Keywords: Artificial inteligence, evolutionary algorithms, commercial video games, real time strategies, machine learning, inteligente machines

Садржај

ПРЕДГОВОР

Цео живот играм видео игрице. Филм „Терминатор“ је пробудио моје интересовање за вештачку интелигенцију и паметне машине које могу да парирају људима у процесу размишљања и доношења одлука, а у некој скоријој будућности да их и превазиђу. Ово интересовање довело је до тога да упишем студије информатике.

При избору теме за мастер рад определио сам се да искомбинујем две ствари које волим: видео-игре и вештачку интелигенцију. Тема мог мастер рада је: „***Имплементација аутоматских стратегија у видео-играма заснованим на еволутивним алгоритмима***“. Идеја рада је имплементација и примена еволутивних алгоритама у комерцијалним видео играма у циљу побољшања способности рачунара да победи противника и да створи осећај игре против стварне особе.

1. УВОД

У овом поглављу објашњава се централна тема мастер рада. Секција 1.1. садржи информације о развоју вештачке интелигенције у рачунарским играма; у секцији 1.2. говори се о примени скрипти у видео-играма; у секцији 1.3 говори се о могућностима побољшања видео-игара скриптама; у секцији 1.4 се спомињу радови са сличном темом; у секцији 1.5 се уводи проблем аутоматског конструисања стратегије која управља играчима (противницима) контролисаним од стране рачунара у рачунарској игри и о приступу решавања тог проблема и коначно секција 1.6 садржи преглед остатка тезе.

1.1 Развој вештачке интелигенције у рачунарским играма

Видео-игре су електронске игре које подразумевају људску интеракцију са корисничким интерфејсом на основу којег се кориснику приказује визуелна информација (слика, тј. анимација) на видео уређају као што је телевизор или монитор. Електронски рачунарски системи, који се користе за играње видео-игара, су познати као платформе (персонални рачунари и конзоле). Видео игре, као и већина других облика медија, могу се сврстати у жанрове на основу многих фактора као што су метод играња, циљеви игре, начина интеракције корисника са игром итд. Примери неких жанрова игара су **пуцачке игре** (енгл. **shooter games**) у којима играч контролише једног карактера који користи оружје из којег се испаљују пројектили (пушке, ракете...) или **игре улога** (енгл. **role-playing games (RPG)**) које црпу идеје из класичне серије Dungeons & Dragons игара и у којима играч преузима улогу једног или више „авантуриста“ који су специјализовани за специфичне вештине (борба прса у прса или коришћење магија) и помоћу њих напредује кроз унапред дефинисану причу. Ова два жанра имају подкатегорије **у првом** и **у трећем лицу** (у првом лицу се камера налази у очима карактера контролисаног од стране играча, док се у трећем лицу камера налази на рамену или иза леђа карактера контролисаног од стране играча). **Стратешке игре**, тј. **стратегије** (енгл. **strategy games**) су облик игара које захтевају пажљиво размишљање и планирање како би се постигла победа. У овом жанру играчу је дат Божји поглед на свет игре и могућност посредне контроле јединица под његовом командом [1].

Од настанка рачунарских игара, **вештачка интелигенција (ВИ)** је њихов саставни и неопходни део, било да су то игре за једног играча или игре на мрежи за велики број играча. ВИ је задужена за управљање објектима у игри и може се рећи да често обухвата све, осим **визуелног** (тзв. графике) и **звучног** дела (тзв. звука) игре. У индустрији која се бави стварањем видео-игара, ВИ подразумева интеракцију између људског играча и

играча контролисаних од стране рачунара, проналажење оптималних потеза коришћењем егзактних алгоритама претраге или техника машинског учења итд. Тренутно, у стварању видео игара приоритет има примена ефикасних алгоритама ВИ, како би се постигло да играчи који играју игре имају илузију да се њихови противници (играчи које рачунар контролише) понашају веома слично правим људским играчима, што игру чини много занимљивијом. Индустрија која се бави видео-играма зна да софистицирана ВИ доприноси побољшању вредности игре, повећава ниво забаве који игра пружа, а самим тим утиче и на повећање прихода. Многе видео-игре су тражене управо због квалитетне ВИ (на пример, Black & White, The Sims, Far Cry).

У прошлости, програмери су се углавном концентрисали на звук и графику у игри. Данас се тај став мења. ВИ већ у почетном дизајнирању игре има једнаку вредност као и графика и звук. Ово се дешава због тога што играчима постаје незанимљива игра против противника чије се понашање лако може предвидети, па програмери морају да уложе напор да сагледају могућност примене напреднијих техника, које ће корисницима игара пружити искуство које желе. Следећи степен у развоју игара биће стварање ликова који се понашају што реалније и који могу што боље да се прилагођавају ситуацијама у којима се налазе током играња игре.

Достигнућа у развоју рачунарских игара користе се и у другим областима. Програмери који раде у области војне науке и тактике схватају да, поред забаве, рачунарске игре могу бити корисне и за обуку војника и симулацију одређених ситуација и тактика. Међутим, да би овакав вид обуке био ефикасан и дао потребне резултате, потребно је да симулатори противника користе напредне технике ВИ које би што реалније симулирале људско понашање и начин размишљања [2].

Интерактивне рачунарске игре постају све атрактивније и за истраживања у другим областима у којима би се применио високи ниво људског понашања применом ВИ.

Једно од основних питања је то, да ли је могуће побољшати компоненту ВИ у комерцијалним рачунарским играма применом техника машинског учења (пре свега еволутивних алгоритама)? У оквиру овог мастер рада разматраће се само процес доношења одлука противника у игри (тј. одлука играча које контролише рачунар) и технике машинског учења биће примењене на проблем проналажења што боље стратегије на основу које аутоматски вођени играчи играју.

1.2 Примена скрипти у видео играма

Већина игара користи скрипте скоро за све имплементације ВИ. **Скрипте** садрже упутства по којима поступају аутоматски вођени играчи. Може се рећи да је скрипта структурирана

репрезентација низа догађаја који могу наступити током једне партије у оквиру неке рачунарске игре, као и одговора аутоматски контролисаних играча на те догађаје. Догађаји који сачињавају скрипту се одвијају секвенцијално, један након другог, под одређеним условима. Скрипте имају четири главне предности:

- разумљиве су,
- лако се примењују,
- лако се проширују,
- могу их употребљавати и они који нису програмери.

За писање скрипти се користе скриптни језици. **Скриптни језик** је програмски језик чији се програмски код извршава интерпретирањем и који се користи као допуна постојећег компилираног програма, писаног на вишем програмском језику. Најчешће коришћени скрипт језици у видео-игрmaма су Lua, QuakeC, Unrealscript, Python итд. [3] Скрипте су углавном статичне (једном написана скрипта се користи у више партија и играчи који играју на основу те скрипте понашају се увек на исти начин), са тенденцијом да буду обимне и сложене. Због статичности, скрипте садрже одређене слабости, тзв. „рупе“, које доводе до тога да људски играч може лако да порази наводно тешке аутоматске противнике. Такође, зато што су статичне, скрипте не могу да се баве непредвиђеном тактиком играча (човека), а људски играчи са друге стране, после одређеног времена, могу да уоче шаблон у понашању аутоматских противника који играју на основу статичних скрипти и да на тај начин стекну предност над њима. Једна од теза овога рада је то да се овакви недостаци могу отклонити применом техника машинског учења. На тај начин би се побољшао квалитет ВИ која контролише аутоматске играче. Машинско учење у рачунарским играма може бити динамично или статично, у зависности од тога када се одвија учење (динамично учење је учење током играња против различитих људских противника, чије се понашање мења и не може се унапред предвидети, док се статично учење одвија током играња против рачунарских скрипти које увек играју на исти начин) о чему ће бити речи у наредним поглављима.

1.2.1. Динамично учење

Динамично учење омогућава да се ВИ прилагоди ситуацији током играња игре. Овај облик учења омогућава противницима, којима управља рачунар, да током игре аутоматски поправе недостатке у скриптама којима је њихова игра вођена. Током овог облика учења, експлоатише се понашање човека и на тај начин се аутоматски играчи прилагођавају људским тактикама игре. Динамично учење може бити **под надзором** или **без надзора**. Динамично учење под надзором подразумева да играч (човек) оцени колико је ВИ успешна, а то онемогућава аутоматско прилагођавање. Динамично учење без надзора подразумева да систем аутоматски врши оцену своје успешности током игре и да се на

основу тога аутоматски прилагођава. Да би се динамично учење применило у пракси, оно мора бити брзо и ефикасно.

Динамични скриптинг је техника динамичног учења без надзора. Пожељно је да ВИ у играма буде таква да аутоматски контролисани играчи играју подједнако ефикасно у свим ситуацијама које могу наступити током једне партије, тј. да је ВИ испрограмирана без икаквих посебних претпоставки о неким специфичним ситуацијама које наступају током игре. Да би ВИ задовољавала овају услов, потребно је дубоко познавање целокупног домена игре, односно, свих правила игре и потенцијалних ситуација које током игре могу наступити. Обично се динамичке скрипте одвијају на основу неких фиксираних правила (правила описују могуће ситуације у игри и могуће одговоре на њих), међутим, техникама учења одређује се приоритет примене тих правила (правила се обично наводе са неким тежинама, а током учења врши се одређивање оптималних вредности тих тежина, како би се постигла што ефикаснија игра).

Техника динамичног скриптинга је погодна, јер испуњава наредна четири потребна захтева [4]:

- рачунски је брза, јер захтева само извлачење правила из базе правила и ажурирање њихове тежине једном по игри,
- ефектна је, јер се сва правила у бази правила заснивају на познавању домена игре,
- робусна је, јер правила не бивају уклоњена одмах када су добила негативну оцену,
- прилагодљива је, јер може брзо да се прилагоди статичном скриптингу (тренирању против фиксираних противника) или промени тактике.

Динамично учење се може реализовати и помоћу стабла одлучивања. Учење применом стабла одлучивања је метод апроксимације дискретних циљних функција у коме се научена функција представља у виду стабла. Сваком чвору стабла одговара тест неког атрибута инстанце, а гране које излазе из чвора различитим вредностима тог атрибута. Листовима одговарају вредности циљне функције. Инстанце су описане вредностима својих атрибута. Класификују се полазећи од корена, спуштајући се низ грану која одговара вредности тестираног атрибута инстанце коју класификује. Класа се додељује инстанци када се дође до листа [5].

1.2.2. Статично учење

Статично учење подразумева побољшање ВИ у игри, без учешћа човека, тренирањем против фиксираних, аутоматски вођених противника. Код статичног учења програм, по завршетку партије, оцењује игру ВИ и применом алгоритама машинског учења покушава да је побољша. Адаптивне технологије (на пример, еволутивни алгоритми) омогућавају програмерима технике који могу да допринесу оптимизацији детаља ВИ током развоја рачунарске игре. Оптимизација ВИ је по правилу тежак проблем и важна фаза током развоја једне рачунарске игре. У комерцијалним играма постоји стотине параметара који утичу на стил игре аутоматских играча. Тестирање сваке комбинације је немогућ задатак, посебно ако се има у виду обично кратак временски период подешавања, који је на располагању програмерима ВИ.

Примена статичног учења омогућава стварање нових стратегија и тактика аутоматски вођених противника. Такође, овај облик учења се може комбиновати и са динамичним учењем. У том случају, статично учење побољшава динамични процес откривајући нове стратегије и тактике које се могу додати динамичном скрипту и на тај начин може допринети ефикасности у суочавању са тактикама играча (човека), које програмер није предвидео [4].

1.3 Могућности побољшања видео игара скриптама

Помоћу технике статичног скриптовања могуће је генерисати одређени број скрипти за које се показује да су ефикасне у игри против одређеног типа играча. Техникама динамичког скриптовања, могуће је све те скрипте комбиновати и објединити у стратегију која ће бити ефикасна против већег броја разнородних типова играча јер ће динамично научени системи бити у стању да препознају карактеристике стила игре играча против којих тренутно играју и да у бази статично генерисаних скрипти пронађу и покрену оне које играју ефикасно против играча са баш таквим стилом игре. Међутим, ово још увек није остварено у пракси. То доводи до уочавања следећег проблема:

- *У којој мери статична техника учења може да се користи за динамичне скрипте и побољшање базе правила, а све у циљу побољшања ВИ у рачунарским играма?*

Многе технике машинског учења погодне су за статично учење, јер нису у противречности са четири претходно изнета захтева. Пожељно је фокусирати се на

еволутивне технике које доприносе побољшању ВИ против аутоматски вођених противника. Када се правилно примењује, еволутивни алгоритам може да са бави сложеним окружењима, као што су рачунарске игре.

Најкомплекснији вид ВИ налази се у СРВ играма (Стратегија у реалном времену). Динамични скрипт се показао као добар за игре у трећем лицу, али не и за СРВ игре, па због тога треба размотрити следећа питања:

- Да ли је могуће осмислити и реализовати еволутивни алгоритам за откривање нове тактике и стратегије у СРВ играма?
- Да ли ће и у којој мери новооткривене тактике и стратегије побољшати перформансе противничког играча?

1.4 Радови на сличну тему

Интересовање на ову тему је велико и постоји одређен број радова у којима се покушало са имплементацијом еволутивних алгоритама ради побољшања ВИ у рачунарским играма. Већина радова је показала добре резултате. Један од примера је мастер теза Ченга [6]. Као игра на којој се врши експеримент коришћена је игра **Wargus** која је испрограмирана коришћењем платформе за развој игара **Stratagus.** Wargus је бесплатна обрада игре Warcraft 2. Тестирања су базирана на игри против скриптова “Soldiers Rush” и “Knights Rush”, који су били доступни у Wargus-у. Ченг је показао да је у сваком експерименту његов програм веома брзо дошао до јаких тактика, којима се постиже висока стопа победе [6].

Са напретком игара мењају се и параметри релевантни за експеримент, па је неминовно поновно спровођење истраживања на новим играма, што је један од циљева ове тезе. Овим истраживањима би се испитало да ли је и даље могуће побољшање ВИ, у промењеном окружењу, коришћењем еволутивних алгоритама.

1.5 Циљ тезе

Циљ тезе је да се покаже да је применом статичког учења и еволутивних алгоритама могуће аутоматски конструисати ефикасне стратегије на основу којих играју играчи којима управља рачунар у савременим стратешким играма у реалном времену. Да би се то показало, потребно је реализовати следеће кораке:

1. избор репрезентативне игре и алата за реализацију експеримената,
2. имплементација система који применом статичног учења, помоћу еволутивних алгоритама, открива нове стратегије и тактике у СРВ играма, које су ефикасније од полазног скупа ручно конструисаних стратегија.

1.6 Преглед остатка тезе

Остатак тезе је обухватио следеће: у поглављу 2 разматрају се различите технике ВИ у комерцијалним рачунарским играма, релевантним за ову тему. У поглављу 3 се се разматра истраживање циља, односно одабир репрезентативног окружења, у поглављу 4 разматра се примена еволутивних алгоритама у изабраним играма, како би открили нове тактике и стратегије. У поглављу 5 је обрађен коначни циљ, и објашњено је како су примењене статичне тактике и стратегије. У том поглављу су такође приказани резултати изведеног експеримента.

2. Технике ВИ у рачунарским играма

Секција 2.1 садржи приказ технике ВИ базиране на правилима, која су релевантна за ово истраживање. У секцији 2.2 разматране су различите врсте машинског учења и објашњено је који су изазови да би се овакво учење могло реализовати у рачунарским играма. У секцији 2.3 приказан је увод у еволутивне алгоритме.

2.1 Технике у играма које се заснивају на ВИ

Приступи за избор технологије за ВИ, који се базирају на избору правила, су широко прихваћени и састоје се од статичног и динамичног дела. Скрипте се имплементирају у скрипт језицима. Неке игре за управљање ВИ користе скрипт језике као што су Bioware NWscript, UnrealScript или LUA.

Неки принципи на којима се заснива избор технологије за развој ВИ су:

- сви принципи се преузимају од познатих програмских парадигми,
- дизајн на основу правила је генерално предвидив, а самим тим је и лак за тестирање и отклањање грешака,
- већина програмера нема довољно знања о комплекснијим технологијама ВИ и зато их не користе.

Скрипте садрже низ предефинисаних правила које противник извршава у задатим ситуацијама. Неки противнички потези зависе од времена, а неки од ситуације. Ако скрипт неком играчу налаже да крене у напад када му величина војске достигне, на пример 10 војника, он ће сваки пут нападати *само* када има тачно 10 војника. Овакву тактику играч може експлоатисати тако што ће противнику увек убити 10-ог војника и самим тим противник га никада неће напасти, нема потребе да му уништава целу војску. Због тога скрипте не омогућавају противнику да се прилагоди тактици играча, већ слепо прате команде. Врло брзо, чим се открију шаблони у игри, победа постаје тривијална.

Тренутно доминантна техника која се базира на правилима је писање скрипта које се заснивају на симулацији рада коначних аутомата којима се представљају стања у програму и акције које је у одређеним условима потребно извршити. Овом програмском техником се поједностављују одређени сложени задаци и такве скрипте се користе за контролу игре од споља. Скрипте прате предефинисане акције, које се примењују у одређеним ситуацијама. Ситуације се представљају чворовима, а гранама између тих чворова.

Коначни аутомат је математички модел који се користи за дизајнирање компјутерских програма и секвенцијалних логичких кола. Формална дефиниција би била да се коначни

аутомат над коначном азбуком $\sum$ састоји од коначног скупа Q, који се назива скуп стања, скупа $I \subset Q$ почетних (или иницијалних) стања, скупа $F \subset Q$ завршних (или финалних) стања и скупа $\Delta \subset Q \times \sum \times Q$ који се назива релација прелаза [7]. Коначан аутомат садржи коначан број стања и у датом тренутку може бити само у једном од тих стања. Свако стање има своје понашање и сопствени покретач. Покретачи узрокују прелаз из једног стања у друго. Стратегије описане коначним аутоматима у рачунарским играма користе се чешће од било које друге технике ВИ зато што су аутомати једноставни за програмирање, једноставни за разумевање и користе се за широку класу проблема. Недостатак примене аутомата је да често долази до предвидљивости у игри. Велики број одговора може бити генерисан на дати скуп симулација и на тај начин може постати непредвидив за понашање [8].

Скрипте и коначни аутомати су детерминистичке технике ВИ које захтевају да програмер предвиди све аспекте понашања играча. Дакле, програмер мора да предвиди сва могућа стања и ситуације и одреди одговарајуће акције у сваком стању, тј. ситуацији. Већина рачунарских игара има на стотине различитих параметара и сценарија који утичу на понашање ВИ. Техникама машинског учења се претражује простор могућих потеза у рачунарским играма и ефикасно траже успешне комбинације.

2.2 Машинско учење у играма

Машинско учење је дисциплина која се бави изградњом прилагодљивих рачунарских система који су способни да побољшавају своје перформансе користећи информације из искуства [5]. Процес учења у играма генерално подразумева прилагођавање понашања аутоматски вођеног играча током игре у циљу побољшања његових перформанси. Треба имати на уму да се термини динамични и статични, који се користе у секцији 1.2, односе на време када се одвија учење, тј. током играња против људи или током само-игре, а не на то како се учење постиже.

Постоје разлике између директног и индиректног учења, тј. директне и индиректне адаптације:

- ***Индиректна адаптација***: Индиректна адаптација настаје када се алтернативе за одређене аспекте понашања праве на основу статистике током игре. Одлуку о томе које статистике се прате и како се оне тумаче, тј. како се на основу њих врше неопходне промене у понашању аутоматски контролисаних играча, доносе дизајнери ВИ. Механизам учења је на тај начин ограничен на извлачење информација из игре и не утиче директно на промену понашања. Индиректна адаптација је ефикасна, јер њена широка употреба претходног знања чини

механизам учења једноставним, високо ефикасаним и лако контролисаним, подложним тестирању и потврди. Техника "Динамичне поставке тежине", која се користи у Max Payne 2, је пример индиректног учења у рачунарским играма. Тежина противника се мења током игре и што је играч ефикаснији повећава се и сама тежина игре.

- ***Директна Адаптација***: Директна адаптација се односи на оптимизационе алгоритме учења којима ВИ директно мења понашање на основу процене својих перформанси у свету игре. У основи, алгоритми за учење траже оне вредности параметара ВИ који нуде најбоље перформансе, односно траже најбоље понашање. Директна адаптација, генерално, није ефикасна и тешко се контролише, што је чини тешком за отклањање грешака. Такође, тешко је утврдити одговарајући учинак функције прилагођености. Међутим, битна предност директне адаптације је та да не ограничава понашање противника и захтева минимални утицај човека. Блек показује да се директно учење успешно може применити у рачунарским играма [9].

Компаније које се баве стварањем рачунарских игара су опрезне у коришћењу техника машинског учења из следећих разлога:

- ови системи могу да науче погрешне лекције,
- често су врло захтевни у смислу времена обраде,
- могу бити тешки за подешавања којима се постижу жељени резултати.

Ипак, машинско учење може да постави већи изазов, а то је оно што играчи желе. Алгоритми машинског учења могу се користити за прилагођавање игре условима који се не могу предвидети пре пласирања на тржиште, као што су посебни стилови и склоности појединих играча. Када се користи правилно, машинско учење може да помогне да игре буду још издржљивије и отпорније на играчке подухвате. Машинским учењем може да се промени начин на који се игре играју, тако што се играчи подстичу да не усавршавају само једну технику, већ да континуирано трагају за новим стратегијама како би поразили ВИ. Програмери могу да користе технике машинског учења за генерисање софистицираних ВИ и пре пласирања игре [9].

Може се закључити, да учење треба да буде наредни велики помак у рачунарским играма. Програмери се удаљавају од „тврдог кодирања“, које се заснива на правилима окренутим флексибилнијим ВИ базираним на адаптивним технологијама. То су, на пример, стабла одлучивања, неуронске мреже и еволутивни алгоритми.

2.3 Увод у еволутивне алгоритме у видео играма

Еволутивни алгоритми су широк назив за групу алгоритама за оптимизацију и претраживање, који се заснивају на принципу биолошке еволуције. Они обухватају генетске алгоритме**,** класификационе системе и генетско програмирање**.**

Генетски алгоритми су фамилија општих хеуристичких алгоритама глобалне претраге заснованих на Дарвиновој теорији еволуције [10]. Они припадају широј групи еволуционих алгоритама који користе технике инспирисане еволуционом биологијом. Сматра се да је модерне генетске алгоритме увео Џон Холанд [11]. Генетски алгоритми мотивисани су природним процесом еволуције. У природи, еволуција је процес у којем јединке које су најбоље прилагођене околини преживљавају и остављају потомство које је најчешће исто тако или боље прилагођено околини.

Генетски алгоритми се имплементирају као рачунарска симулација у којој популација апстрактно репрезентованих **јединки** које су кандидати за решење оптимизационог проблема, треба да се постепено приближава бољим решењима, тј. да садржи све боље и боље јединке. Технике оптимизације почињу од једног потенцијалног решења за дати проблем и онда постепено прилагођавају ово решење да би се постигао оптимум. Генетски алгоритми користе популацију решења, кодираних на неки начин (најечшће бинарно) и називају се **хромозоми**. Сваки ген у хромозому представља променљиве или аспект решења. Хромозомима у популацији се додељује вредност функције прилагођености. **Функција прилагођености** указује на то колико је ово решење успешно у решавању проблема у односу на друга решења у популацији. За стварање нових решења еволутивни алготирам примењује **генетске операторе** над постојећим решењима. Генетски оператори којима треба само један родитељ се називају оператори **мутације**, док се оператори са два или више родитеља називају оператори **укрштања.**

- **Репрезентација јединки**

Репрезентацију јединки треба одабрати тако да сваки хромозом описује једно могуће решење разматраног проблема и да се за такву репрезентацију једноставно могу дефинисати коректни генетски оператори. Генетски оператори морају бити тако дефинисани да се њиховом применом не могу добити јединке које не представљају могућа решења. Правилан одабир репрезентације може допринети ефикасности алгоритма. Најчешћа и најуспешније коришћена репрезентација јединки је помоћу низа битова. Сваки бит се назива ген.

Кодирање умногоме зависи од проблема и не постоји јединствено кодирање које ће задовољити сваки проблем.

- **Функција циља и функција прилагођености**

Функција циља пресликава одређени догађај или вредност једне или више променљивих у реалан број. Тај број представља „цену“ која се везује за тај догађај. Функција циља може бити функција губитка или функција добитка. Код генетских алгоритама се користи одређени тип функције циља, функција прилагођености.

Функција прилагођености представља оцену квалитета јединке. Погодан избор функције прилагођености је од изузетне важности за ефикасност алгоритма. Што је вредност функције прилагођености већа, већа је вероватноћа да се та једника користи за генерисање нове генерације. Током извршавања алгоритма генеришу се нове генерације, при чему се за сваку јединку у новој генерацији изнова рачуна функција прилагођености. Наравно, неће све јединке у новој генерацији бити промењене, већ ће неке остати исте. Зато се дешава да ће алгоритам за јединке које се нису промениле изнова рачунати вредност функције прилагођености коју је већ једном израчунао. Овај проблем се решава **кеширањем** функције прилагођености. Већ израчунате вредности функције прилагођености се памте и за непромењене јединке преносе у нову генерацију. Тиме се смањује број рачунања и повећава ефикасност и брзина рада алгоритма.

- **Генетски оператори**

Мутација узима један хромозом као родитеља и умеће, брише или мења гене како би настало дете (нов хромозом). Улога мутације у генетским алгоритмима је да спречи да јединке у популацији постану сувише сличне и да помогне у обнављању изгубљеног генетског материјала.

Генетски оператори којима треба више родитеља се зову **укрштања**. У укрштању учествују најчешће две јединке које се називају родитељи. Резултат укрштања је једна нова јединка или две нове јединке које зовемо деца. Очекивано је да деца наслеђују прилагођеност родитеља, па чак и да имају и бољу прилагођеност. У једној варијанти довољно је изабрати тачке прекида и прекомбиновати низове битова, дете од једне тачке прекида до следеће наслеђује гене од једног родитеља, а наредни део од другог. Укрштање може бити једнопозиционо, двопозиционо, вишепозиционо и униформно [12].

Да би се изабрали родитељи за генетске операторе, примењује се механизам селекције. Селекција обично има за циљ да са већом вероватноћом одабере јединке које су боље прилагођене, рачунајући да ће од боље прилагођених родитеља настати боље прилагођена деца. Постоји неколико често коришћених облика селекције.

- **Рулетска селекција** је процес селекције у коме веће шансе да учествују имају прилагођеније јединке. Вероватноћа да јединка буде изабрана је једнака количнику вредности њене функције прилагођености и суми функција прилагођености осталих јединки. У рулетској селекцији могуће је да једна јединка буде више пута изабрана да учествује у следећој генерацији и репродукцији.

- **Турнирска селекција** подразумева турнире у којима веће шансе за победу имају прилагођеније јединке. За један турнир се случајним избором бира ***k*** јединки из популације, а након тога се сортирају по вредности функције прилагођености и из тог низа се бира једна јединка са вероватноћом ***p***. Јединкама које су једном изабране може се забранити учешће у даљим турнирима.

Производња нових јединки, процес еволуције, наставља се све док се не постигне неки унапред дефинисани циљ. Нова решења мењају постојећу, или се уносе у нову популацију. Резултат тог процеса је популација јединки које се постепено прилагођавају ограничењима свог дигиталног окружења, тј. развијају се током времена. Најспособнија јединка у крајњој популацији се сматра траженим решењем проблема [5].

Algoritam: Opšti genetski algoritam

Ulaz: —
Izlaz: najkvalitetnija jedinka u tekućoj populaciji

1. Generiši početnu populaciju potencijalnih rešenja;
2. Izračunaj prilagođenost svake jedinke u populaciji;
3. Izvršavaj sledeću petlju sve dok nije zadovoljen uslov zaustavljanja:
   - Izaberi iz populacije skup jedinki za reprodukciju;
   - Generiši novu generaciju, primenivši nad izabranim jedinkama proces reprodukcije (tj. genetske operatore ukrštanje i mutaciju);
   - Izračunaj prilagođenost novogenerisanih jedinki;
   - Zameni najlošije jedinke u populaciji novogenerisanim jedinkama.
4. vrati najkvalitetniju jedinku u tekućoj populaciji

Слика 1 : Општи генетски алгоритам [5]

Еволутивни алгоритми су робусне методе претраживања, односно, добро се примењују у различитим окружењима и на различитим проблемима. Њима се ефикасно врши претраживање у великим, комплексним, или лоше дефинисаним просторима претраживања. Када се одреде одговарајућа репрезентација и функција прилагођености, еволутивни алгоритам може бити моћно средство за решавање проблема са великим бројем променљивих и хаотичним везама међу њима, попут рачунарске игре. Колико је аутору овог текста познато, еволутивни алгоритми нису често коришћени у комерцијалним рачунарским играма које се играју на мрежи. Програмери су запоставили еволутивне алгоритме јер имају тенденцију да буду скупи и да производе неефикасано понашање. Други недостатак је тај што не гарантују да ће пронаћи добро решење, чак ни осредње. Међутим, еволутивни алгоритми се спорадично користе у једноставнијим рачунарским играма које се играју у локалу [13].

3. Стратешке игре у реалном времену

У овом поглављу се разматрају стратешке игре у реалном времену и избор игре погодне за даље експерименте. У секцији 3.1 објашњавају се основе стратегије у реалном времену, у секцији 3.2 се описује ВИ у стратегијама у реалном времену. У секцији 3.3 су приказани критеријуми за избор окружења. У секцији 3.4. представљен је Warcraft III,а у секцији 3.5. је закључак.

3.1 Стратегије у реалном времену

Данашње **стратешке игре у реалном времену (СРВ)** су једноставне војне симулације које захтевају од играча да контролише армије (армије се састоје од различитих типова јединица), и да порази све супротстављене снаге (њих може контролисати било рачунар, било други играчи који истовремено играју игру преко рачунарске мреже). У већини стратегија у реалном времену, кључ за победу лежи у ефикасном прикупљању и управљању ресурсима, као и одговарајућој расподели ових средстава током различитих елемената игре. Типични елементи стратегија у реалном времену подразумевају изградњу објеката, истраживање нових технологија и борбу. Dune 2 (слика 2) се сматра за прву СРВ игру. Назив „стратегије у реалном времену“ је осмислио Вествудов Брет Спери. У почетку су само хтели да класификују ову игру као ратну или као стратешку, али Спери је сматрао да би то могло да одбије играче због огромне сложености у конвенционалним ратним и стратешким играма. По Сперију су пре 1992. године, ратне игре и игре стратегије биле веома специјализовани жанрови и најбоље их је назвати стратегије у реалном времену, јер оне то заправо и јесу. Термин „реално време“ у називу ове врсте игара указује на чињеницу да време у игри тече унапред одређеном брзином и да играч нема времена да пуно размишља током игре већ мора да издаје налоге брзо и често. Ипак, многи играчи СРВ се не слажу са употребом речи „стратегија“ у овим играма, тврдећи да стратегије у реалном времену нису ништа друго него јефтина имитација потезних игара. Ова тврдња проистиче из тенденције да побеђује играч који даје налоге бржим темпом [14].

Од настанка Dune 2, објављено је доста нових стратегија у реалном времену. Компанија Blizzard је 1994. године објавила игру Warcraft, стратегију у реалном времену постављену у домену фантазије. Његови наставци, Warcraft II (1995. године) и Warcraft III (1999. године), постигли су један од највећих успеха које је овај жанр доживео. Играчи ову игру често играју пуно пута. И дан данас постоји подршка у оквиру Battle.net-а за ову игру и организују се светски турнири. Новије СРВ игре од Warcraft III, као што су Starcraft 2, су

подигле жанр на виши ниво, али углавном у смислу графике и звука (слика 2), а не у смислу механике играња.

Слика 2: Dune 2 (слика лево) је била прва СРВ игра икад, Starcraft 2: Hearth of the Swarm (слика десно) је најновија стратегија компаније Blizzzard.

3.2 Вештачка интелигенција у стратегијама у реалном времену

ВИ је одувек била веома важна карактеристика стратешких игара. Чињеница је да стратешке игре не могу да се ослањају само на графику, већ је неопходна квалитетна ВИ, како би се задржала пажња играча.

Дизајнери ВИ у стратегијама у реалном времену морају да обрате пажњу на многе детаље, попут управљања ресурсима, детаљне анализе терена, моделовања игре противника, мапирања утицаја средине и још много тога. Додавање суптилних и комплексних алгоритама у ВИ играча којима управља рачунар у великој мери ће унапредити корисниково задовољство игром и његов осећај уживања у игри. ВИ треба да аутоматским играчима обезбеди што паметније и прилагодљивије понашање.

Планирање војног успеха у стратегијама у реалном времену може да се подели на две одвојене категорије: **стратегија** и **тактика**. Тактика покрива интеракције мањег обима, као што су извиђање бојног поља или заробљавања непријатељског града. За разлику од њих стратегије су свеобухватне. Генерално, најважнији стратешки принципи су:

- јединство команде (постојање једног централног вође),
- контрола циља (постојање ратног плана),
- флексибилност (могућност промене борбеног плана),
- економија силе (подела снага и ресурса на одговарајући начин),
- иницијатива и масовност у акцијама.

Неки програмери тврде да је добро структурирана вишеслојна ВИ комбинована са резоновањем усмереним ка циљу, већ спремна да се позабави неким од војних идеологија стварног света.

Ремзи предлаже **вишеслојан, хијерархијски оквир** (енгл. framework) вештачке интелигенције, где различитим нивоима ВИ управљају различите компоненте (тзв. **менаџери**) које управљају задацима ВИ на тим нивоима. Овакав оквир омогућава доношење „великих стратешких одлука“ на вишем нивоу, на коме постоје менаџери за извршење задатка. Игра Empire Earth компаније StainlessSteel Studios је вероватно игра са најуспешнијом ВИ до сада и она је заснована на вишеслојној ВИ [9].

Ова игра је распоредила задатке ВИ на следеће компоненте (менаџере):

- **менаџер градње:** одговоран је за постављање објеката и градова, одређујући при том места на којима је оптимално поставити грађевине одређене врсте, водећи при том рачуна о правилима игре тј. о томе где зграде могу или не могу бити постављене,
- **руководилац јединицама:** прати које јединице се тренирају у којим зградама, прати границу становништва и одређује приоритет захтева за изградњу и тренирање појединих јединица,
- **менаџер ресурса:** одговоран је за грађане који сакупљају ресурсе и одговара на захтеве руководилаца јединица и менаџера градње. Ова компонента је такође одговорна за ширење и есплоатисање нових ресурса и локација,
- **менаџер истраживања:** испитује технологије и бира их на основу њихове корисности и трошкова,
- **менаџер борбе:** одговоран је за управљање војним јединицама на бојном пољу, обучава јединице преко менаџера јединица и користи их у нападу или одбрани, где је потребно,
- **менаџер цивилизације:** координира између менаџера градње, менаџера јединица, менаџера ресурса и менаџера истраживања, управља експанзијом градова, одређује лимите потрошње, градње и унапређење јединица.

У стратегијама у реалном времену непрактично је размишљати унапред предубоко јер је сам број могућих потеза из сваког стања превелики. Бољи приступ је орјентација ка достижним подциљевима. Крајњи циљ стратегија у реалном времену треба да буде победа. Међутим, овај циљ је превише компликован да би се директно решавао. Решење је у томе да се испуњење тог главног подциља подели на испуњење једноставнијих

подциљева, чије постепено испуњавање доводи до испуњења крајњег циља. Подциљеви би могли бити, на пример, „прошири базу“ или „онемогући противнику прикупљање ресурса" [15].

3.3 Избор окружења за експеримент

У експерименталном делу ове тезе биће имплементиране технике машинског учења засноване на еволутивним алгоритмима у циљу изградње што боље стратегије игре у оквиру једне стратешке игре у реалном времену. У потрази за одговарајућим окружењем, тј. одговарајућом игром, треба узети у обзир следеће услове:

1) потребно је да окружење буде лако доступно и прилагодљиво,
2) окружење игре треба да садржи скриптинг језик, пожељно је да то буде са софистицираним интерфејсом за програмирање ВИ и могућношћу да подржи технике учења,
3) пожељно је да се експерименти у окружењу одвијају брзо,
4) окружење игре мора да буде репрезентативно, значи мора имати нетипичну ВИ.

Модерне комерцијалне игре су погодне за истраживања у погледу њихових реалних услова и нетипичне ВИ. Нажалост, већина игара не дозвољава било какву промену над игром и њеним кодом. Ипак, неке комерцијалне рачунарске игре укључују алатке за измену игара и изградњу ВИ, наравно са одређеним ограничењима.

Warcraft III, као једна од таквих игара, има моћан алат за промену игре, изградњу мапа, сценарија и ВИ. Додатни разлог за избор ове игре је симболичке природе, јер је то једна од најуспешнијих стратегија у реалном времену. За саму имплементацију еволутивног алгоритма коришћен је програмски језик Java и Net Beans окружење.

3.4 Warcraft III и WorldEditor

Warcraft III се игра на мапама различитих димензија. Терен на мапама чине елементи попут река, планина или стена. Мапа је почетно покривена црном маском, што значи да се изглед неког дела мапе не може видети док се исти не истражи. Такође, делови мапе који су раније истражени, а тренутно нису у видокругу неке од играчевих јединица, покривени су „маглом рата“ - терен остаје видљив, али било које промене (померање јединица,

прављење/рушење зграда или дрвећа) нису видљиве. Играчи морају да граде насеља (базе) како би сакупљали ресурсе, бранили се од напада и правили јединице за борбу. Постоје три главна ресурса у игри: злато, дрвна грађа и храна. Прва два ресурса су потребна за грађење и прављење јединица, док храна ограничава максимални број јединица које играч може да поседује одједном.

Слика 3: снимак екрана игре Warcraft III

У претходним Warcraft играма, постојале су само две расе за играње, *Орџи* и *Људи*. Разлике између те две расе су биле чисто козметичке: свака јединица *Људи* је имала свој еквивалент код *Орака*. У игри Warcraft III су додате још две расе: *Немртви* и *Ноћни Вилењаци*. Такође, разлике међу расама су драстично повећане, тако да свака раса има своје предности и мане. На пример, *Људи* и *Орџи* су јачи током дана, док *Ноћни Вилењаци* ноћу постају невидљиви. Грађевине за различите расе се знатно разликују. *Људи* имају фарме, јефтине грађевине које обезбеђују мање хране од одговарајућих (скупљих) грађевина осталих раса. Код *Орка* то су „јазбине“, скупље и спорије за изградњу, али зато доносе више хране и могућност да радници *Орка* нападају непријатеље стрелама из „јазбина“. *Немртви* имају „Зигурате“, који могу да се надограде у одбрамбене куле. Коначно, *Ноћни Вилењаци* имају „месечеве бунаре“, који се могу користити за обнављање животне енергије и енергије потребне за коришћење магија пријатељских јединица.

Слика 4: расе у игри Warcraft III

У игри Warcraft III су додате и моћне јединице - хероји. За сваког убијеног непријатеља, хероји добијају поене искуства помоћу којих напредују и добијају нове магије, што уводи елементе стратегије у реалном времену. Поједини хероји поседују ауре - магије које имају трајни ефекат на све јединице у њиховом окружењу. Хероји могу да носе са собом „предмете“, који им повећавају вештине, штит и друге особине. Највиши ниво који херој може да достигне на нормалној мапи је десети, али је у прилагођеним мапама могуће повећати максимални ниво и до 10.000. На шестом нивоу, херој може да научи своју „ултимативну“ магију, која је јача од свих његових осталих магија. У претходном делу (Warcraft II) хероји су били само појачане верзије обичних јединица, са мало јачим оклопом и другим карактеристикама, или понекад са слабијим карактеристикама (на пример Гул'Дан). Warcraft III има хероје који су много јачи од обичних јединица. Играчи праве „олтаре“ помоћу којих могу да врате своје хероје из мртвих.

У игри се уводе нове јединице за сваку расу и неутралне банде, такозвани „крипови“ (жаргонски назив, настао од енглеског назива: *creeps*), рачунарски контролисане јединице које су непријатељске свим играчима. Крипови чувају кључне локације на мапи, као што су рудници или неутралне зграде. Када их убије, играч добија искуство и често неки предмет за хероја.

У Warcraft III такође се уводи и смена дана и ноћи. Осим естетског изгледа, смена дана и ноћи даје поједине предности/мане неким расама. Ноћу већина „крипова“ спава, па је ноћ безбеднија за истраживање мапе. Међутим, ноћу се смањује видокруг већини јединица

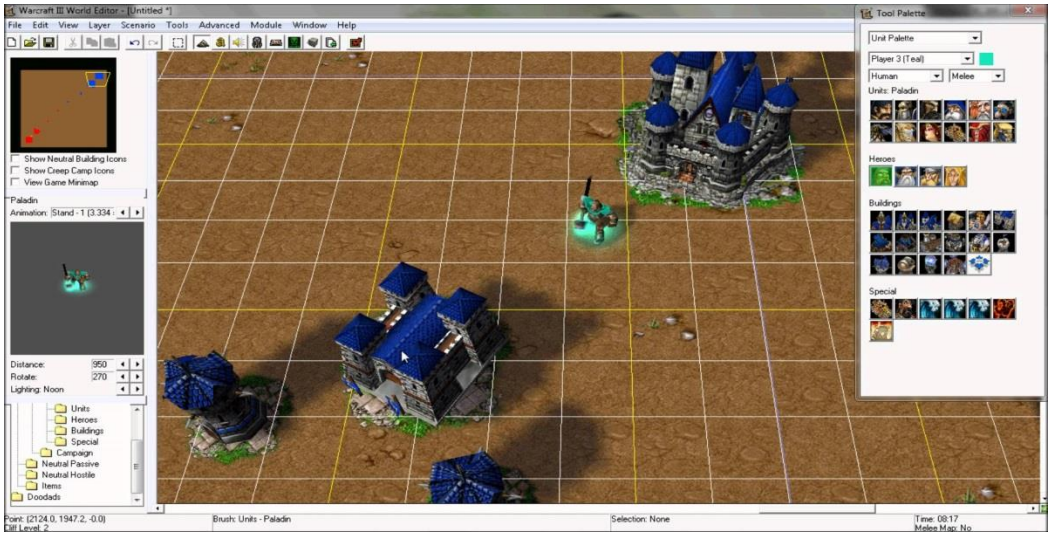
(осим код *ноћних вилењака*, који то спречавају одговарајућом надоградњом). Улогу у прецизности гађања има 3Д терен: јединице на висини имају бољи поглед и боље гађају од оних на нижем терену [16].

Ради једноставности у експерименту концентрисаћемо се само на расу *Људи*, а имплементација за остале расе је суштински иста и лако може да се изведе.

Warcraft III WorldEditor је алат за мењање игре који је уграђен у игру и представља њену стандардну компоненту. Он омогућава корисницима да креирају и мењају мапе са високим нивоом детаља и флексибилности. Алат је издељен на различите модуле који се користе за мењање различитих аспеката игре.

1. едитор земљишта: користи се за обраду мапе, додавање воде и дрвећа, зграда, јединица итд...,
2. едитор објеката: дозвољава промену већине атрибута објеката, као што су изглед и способности јединица, зграда и предмета и дозвољава креирање ових објеката,
3. едитор окидача: дозвољава креирање окидача који се активирају када се деси одређени догађај,
4. ВИ едитор: дозвољава кориснику креирање скриптова ВИ.

WorldEditor такође дозвољава тестирање креираних мапа и ВИ са убрзаним протоком времена, што олакшава тестирање тактика одабраних од стране нашег алгоритма. Ипак, нагласимо да иако се тестирање врши убрзаним темпом и даље је потребно незанемариво време да се свака од тактика истестира.



Слика 5: WorldEditor алат за мењање игре

3.5 Сажетак

У овом поглављу је представљена игра одабрана за даље експерименте. Одабрана је игра Warcraft III са WorldEditor-ом и Јавом као алатима за експериментално истраживање, пошто ова поставка задовољава све захтеве: WorldEditor је софистицирани алат и лако је променљив и проширив; цела игра (укључујући ВИ) је дефинисана у Lua језику, који је моћан скрипт језик (слика 5) и омогућава примену техника машинског учења, као што су динамични и еволутивни алгоритми; експерименти у алату су брзи, јер може да се бира брзина протока времена партије. Најзад, игра Warcraft III се још увек сматра репрезентативном у смислу играња.

```
//*************************************************************************
//*
//*  Building and Harvesting
//*
//*************************************************************************

//=========================================================================
// Specifies building priorities for workers
//=========================================================================
function BuildPriorities takes nothing returns nothing
    local integer mine = TownWithMine()
    call SetBuildAll( BUILD_UNIT, 1, 'ogre', -1 )
    call SetBuildAll( BUILD_UNIT, 1, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 2, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 3, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 4, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 5, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 6, 'opeo', -1 )
    call SetBuildAll( BUILD_UNIT, 1, 'obar', -1 )
    if (gCond_Barracks_2) then
        call SetBuildAll( BUILD_UNIT, 2, 'obar', -1 )
    endif
    if (gCond_Barracks_3) then
        call SetBuildAll( BUILD_UNIT, 3, 'obar', -1 )
    endif
    if (gCond_Barracks_4) then
        call SetBuildAll( BUILD_UNIT, 4, 'obar', -1 )
    endif
    if (gCond_Barracks_5) then
        call SetBuildAll( BUILD_UNIT, 5, 'obar', -1 )
    endif
    if (gCond_Barracks_6) then
```

Слика 6: Пример Lua кода

4. Машинско учење у реалном времену Стратешке игре

ВИ заснована на еволутивним алгоритмима, може да игнорише конвенционално војно размишљање засновано на пописивању свих могућих ситуација које могу наступити у игри и да "размишља ван граница". Претпоставка је, да када имитирамо природни процес опстанка најспособнијих и еволуције, можемо да откријемо неочекиване успешне стратегије и тактике које могу да надиграју уграђене ВИ. У секцији 4.1 и 4.2 говориће се о томе, како се еволутивни алгоритми могу применити на стратегије у реалном времену уопште и посебно на Warcraft III.

4.1 Еволутивни алгоритми који се примењују у играма стратегије у реалном времену

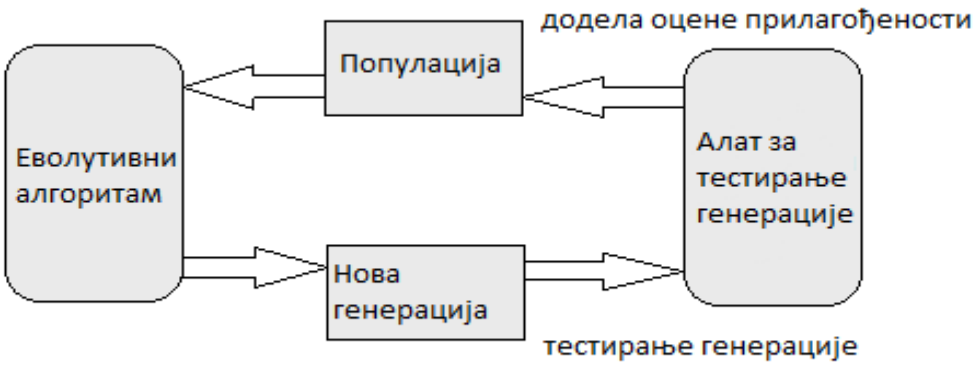
Основна питања приликом дизајнирања ВИ за стратегију у реалном времену која је заснована на еволутивним алгоритмима су начин кодирања стратегија (јединки) и начин оцењивања њихове успешности и један од основних доприноса овог рада је управо дефинисање репрезентације и функције прилагођености које успевају да изнедре стратегије доста боље од полазних.

Шема кодирања мора да буде у стању да представља евентуална квалитетна решења за проблем, а пожељно је да никада не представља неизводљива решења. Из тог разлога дајемо максималну слободу алгоритму у одабиру и параметаризацији правила, али га спречавамо да убаци илегална правила у оптицај. Ово се реализује помоћу стања која одговарају правилима игре која алгоритам може да бира.

Израда одговарајуће функције прилагођености је од суштинског значаја за ефикасан еволутивни алгоритам. У суштини, што је хромозом бољи у решавању конкретног проблема, добиће већи резултат применом функције прилагођености. Адекватно познавање домена игре, тј. познавање специфичности конкретне стратегије у реалном времену је пресудно приликом дефинисања добре функције прилагођености. У стратегијама у реалном времену проблем може бити описан као добијање битке против супротстављене војске на специфичној мапи. Наравно, победа треба да добије већи резултат функције прилагођености него пораз, али такође, и убедљива победа треба да добије већи резултат него тесна победа.

Још једна карактеристика која захтева посебну пажњу приликом дизајнирања алгоритма за одабир стратегија је величина популације. Врло битан део стратегије су грађевински приоритети (нпр. одређени редослед у изградњи зграда). Изворна популација треба да садржи довољно варијација у изградњи за тестирање различитих приступа изградњи царства и тражење оптималног решења. Уколико је изабрана популација превелика, еволуција може дуго да траје. Међутим, уколико је изабрана популација премала, алгоритам би могао да конвергира ка лошем решењу због недовољно узорака из простора за претраживање.

Алгоритам се иницијализује стварањем популације са фиксним бројем јединки (фиксним бројем потенцијалних решења, тј. фиксним бројем конкретних стратегија игре). Нова решења тада играју против статичне ВИ и њихов успех се мери. Када се популација напуни и свим хромозомима се доделе резултати функција прилагођености, успешна решења започињу процес размножавања. Алгоритам ће тада одабрати један од доступних генетских оператора, а затим ће алгоритмима селекције изабрати одговарајући број родитеља и креирати њихове потомке. Процес еволуције ће се наставити све док се не достигне критеријум заустављања, односно, када се пронађе решење проблема. Слика 5. илуструје шематски еволутивни процес примењен у СРВ игри.



Слика 7: Шема еволутивног процеса у СРВ-игри

4.2 Еволутивни алгоритам имплементиран у Warcraft III

Кодирање

Већ је наглашено да је кодирање јединики веома важна компонента генетског алгоритма. Након неколико покушаја, у овом раду смо се одлучили за следећу репрезентацију. Свака јединка, тј. сваки хромозом представља једну конкретну стратегију игре и може имати један од четири гена. То су ген за градњу, ген за хероје, ген за војску и ген за економију. У оквиру имплементације система, подаци о тренутној популацији се памте у текстуалном фајлу. Опис сваке засебне јединке заузима дест редова фајла. Између сваке две јединке стоји реч „next“ која која означава почетак нове јединке. На крају фајла стоји реч „end“ и означава крај фајла. Прикажимо сада пример описа једне јединке:

P *(војска, херој, ресурси, тотал, тотал противника, дужина трајања*

10;5;2;6;4;1200;1 *партије у секундама, индикатор победе: 1-победио, 0-изгубио)*
B
1:1;2:3;3:1;4:4;5:6 *(објекат/надоградња : количина за градњу)*
H
1;3;2 *(први херој : други херој : трећи херој)*
A
1:1:1:1:1:1:1:1:1 *(у овом примеру по један примерак од сваке јединице)*
E
5;5 *(број сељака на дрвима ; злату)*
next

Велика слова одређују који се ген налази у реду испод њих. ***P*** – поени освојени на основу учинка током партије, ***B*** – ген за градњу, ***H*** – ген за хероје, ***A*** – ген за армију, ***E*** – ген за економију. Тачка-запета (симбол **;**) раздваја одређене целине. Код поена су то различити типови поена, код градње су то целине које представљају шта се гради и у којој количини, код хероја то је број хероја, код армије раздваја чете. Унутар сваке чете има девет цифара раздвојених двотачком. Позиција цифре представља тип јединице, а сама цифра количину те јединице у чети. Има девет јединица (нпр. цифра на првој позицији преставља колико има пешадинаца). Код економије цифре представљају број сељака на задатку прикупљања одређеног ресоурса.

Евалуација

Функција прилагођености дефинсана у овом поглављу резултат је неколико покушаја. Основни циљ добре стратегије је да играч вођен том стратегијом оствари победу и што већи, а његови противници што мањи, број поена. Ако је играч вођен стратегијом у стању да дуго остане у игри, али на крају ипак изгуби, вероватно је да је тај хромозом близу

проналажења решења и мали број промена у генима може да доведе до победничке стратегије.

Дакле, за мерење успеха ВИ-је предлажемо наредну функцију прилагођености Ф, која даје вредност у опсегу [0,1] и која је дефинисана на следећи начин:

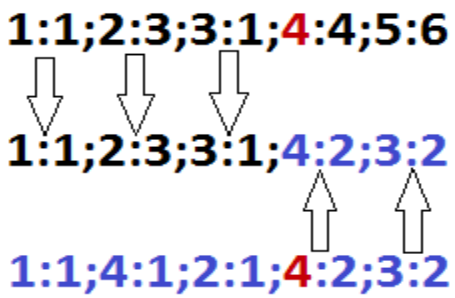
$$Ф = \begin{cases} \dfrac{\frac{пв}{ув} * вп + пх + пр}{т1 + т2} \\ \dfrac{вп + пх + пр}{т1 + т2} \end{cases}$$

У једначини, ВП представља поене за војску играча, ПХ поене за хероје, а ПР поене за ресурсе. Т1 представља укупан резултат играча, а Т2 укупан резултат противника. ПВ представља протекло време, односно време које је протекло пре него што је партија изгубљена од стране једног од играча. УВ представља крај циклуса, односно најдуже време трајања једне партије. Када партија дође до крајњег циклуса, а ниједна војска није била потпуно поражена, резултат се у том тренутку мери и партија се прекида. Фактор ПВ / УВ осигурава да се решењима која изгубе, али одиграју дугу партију, додели већи резултат прилагођености него губитничим решењима која одиграју кратку партију.

Генетски Оператори

Генетски оператори су често дизајнирани да се уклопе у специфичан проблем и облик хромозома. Дизајнирана су два генетска оператора за еволуцију јединке:

1. **Укрштање**: узима два родитеља и комбинује њихове гене. Гене за градњу комбинује тако што узима објекте од једног родитеља, до тренутка док један објекат није заједнички за оба родитеља, затим узима објекте од другог. На овај начин се обезбеђује да се не може узети неизвршив редослед градње. За остале гене узима насумично вредности из родитеља, с'тим да ипак даје малу предност родитељу са већим резултатом поена за тај одређени ген.



Слика 8: пример укрштања гена градње

2. **Мутација**: узима јединку и мења гене тако што им насумично додељује вредности из опсега дозвољених вредности. Све вредности имају подједнаке шансе да буду додељене. Из овог оператора је изузет ген за градњу, јер би његово насумично мењање могло лако да доведе до недозвољених комбинација и не би имало сврхе.

Шанса да се изврши један од ова два оператора је 50%.

Избор

Као механизам селекције уведена је турнирска селекција. Турнирска селекција је најпогоднија у овом случају зато што се:

1. лако спроводи и
2. највероватније ће изабрати добра решења.

Турнирска селекција подразумева да се узима N јединки, прави M група у којима се оне „такмиче“ једне против других и да се победници узимају за родитеље на које се примењују генетски оператори. Ако је вредност параметра N виша, селекција ће бити ужа, док ће мања вредност параметра N дати разноврснији избор. Ова метода ће највероватније изабрати добра решења и спречити брзо конвергирање. Обзиром да се користи мали број рачунских операција, овај механизам је такође рачунски брз.

Алгоритам случајним избором узима три јединке и од њих бира најбољу за родитеља који учествовује у репродукцији. Два родитеља се укрштају и резлтат (дете) се памти у најлошијој јединки. Родитељи и добијено дете се затим изузимају из даљег процеса репродукције за текућу генерацију.

Критеријум заустављања

Ако резултат функције прилагођености прелази жељену вредност, решење је пронађено. У алгоритму, резултат функције прилагођености већи од 0,7 готово сигурно представља убедљиву победу. Како не постоји гаранција да ће алгоритам пронаћи такво решење, мора му се ограничити број генерација. Када се достигне један од критеријума за заустављање, бележи се најбоље решење и евентуално се поново покреће са новом популацијом.

5. Експеримент

5.1 Поставке експеримента

У експерименту се користе статичне скрипте ВИ које су уграђене у самој игри. То су ***Лако*** и ***Нормално***. Квалитет аутоматски генерисаних стратегија врши се у борби против тих статичних скрипти. Постоји и скрипта **Тешко** (енг. **Insane**) која представља највећу тежину, али та скрипта вара (има убрзану градњу, убрзано прикупља ресурсе и нема лимит на број јединица које може да направи) како би била тешка за победу, па из тих разлога није уврштена у експеримент. Величина популације је 20. У теоријски идеалном случају апсолутна победа би имала ефикасност 1. Међутим, ово је немогуће, јер би подразумевало да противник не направи ни једну једину јединицу. Због тога је критеријум заустављања постављен на 0,8. Критерујум заустављања у случају да се не нађе решење је постављен на 10 генерација. Почетна генерација је креирана коришћењем популарних тактика објављених од стране играча и објављених на популарном сајту http://battlenet.org. Већина тактика су балансиране, у смислу да нису ни прејаке, ни преслабе. Такође су убачене две јаке и две брзопотезне тактике. Јаке тактике су у старту имале ефикасност од око 0.56, док је осталим тактикама ефикасност варилала између 0.16 и 0.50. За тестирање је коришћена средња мапа (2vs2), како би играчи имали времена да се развију, а да до сукоба дође у догледно време. Ради лакшег тестирања, брзина протока игре је повећана на 25 пута нормалне брзине, тестеру је откривена мапа и време партије је ограничено на сат времена, када се посматра време у игри. Нагласимо да је, и поред овог убрзања, потребно неколико минута да би се проценио квалитет сваке јединке. Додатни проблем представља то што се свака јединка мора у игру укључити уз интервенцију човека који учитава тактику и покреће убрзану симулацију, прикупља резултате и врши њихово сумирање у коначну вредност функције прилагођености.

5.2 Резултати

Резултати су показали да свака нова генерација има повећање између 0.02 до 0.07 код јединки које су еволуиране.

Укрштањем су добијане како јаче, тако и слабије јединке од родитељских. На пример, спајањем две јаке јединке, 0.52 и 0.60, добијена је јединка са ефикасношћу од 0.57. Спајањем слабе и јаке јединке, 0.23 и 0.50, добијена је тактика са ефикасношћу 0.31. У првом случају јединке су саме по себи биле поприлично јаке, па је нова тактика имала ефикасност ближу ефикасности јаче, док је у другом случају добијена ефикасност ближа

слабијој.

У случају мутације ефикасност је често подизана. На пример, јаким јединкама (нпр. јединки код које је прилагођеност износила 0.60), ефикасност је подигнута за 0.03 (0.63). У случају слабих тактика ефикасност је знатно више подигнута (0.33 на 0.54), јер су јаке тактике близу идеалног решења, простор за побољшање је мали, тако да долази до малих промена. Код слабијх тактика је другачије. Мале промене доносе доносе велика побољшања и остварују доста боље резултате.

После пет генерација пронађена је тактика са ефикасношћу од 0.70. Ова тактика је релативно добра и у већини случајева доноси победу.

Да би се мало разјаснила природа примене генетских оператора у овом контексту, у наредним табелама приказани су резултати појединих укрштања, односно појединих мутација вршених током експеримента.

УКРШТАЊЕ				
P1	P2	Ф(P1)	Ф(P2)	Ф(Д)
T5	T3	0.6	0.24	0.3
T7	T18	0.11	0.52	0.33
T10	T16	0.24	0.64	0.6
T5	T13	0.6	0.32	0.39
T11	T0	0.64	0.59	0.6

Табела 1. : Резултати појединих укрштања

МУТАЦИЈА		
P1	Ф(P1)	Ф(M)
T18	0.16	0.62
T0	0.5	0.59
T2	0.64	0.36
T16	0.64	0.69
T6	0.64	0.71
T12	0.35	0.43
T2	0.36	0.65
T14	0.31	0.33
T4	0.24	0.56
T17	0.29	0.75

Табела 2. : Резултати појединих мутација

5.3 Дискусија

Током процеса еволуције уочени су одређени шаблони у развоју. Преферира се развој већег броја јединица на почетку игре и мањи број доступних јединица на крају игре. У почетку се прави што више пешадинаца и артиљераца, да би се касније један део њих заменио јачим јединицама, као што су на пример коњаници и бомбардери. Такође, преферира се градња јединица за подршку, тј. свештеника који исцељују рањене јединице. Повећан акценат се ставља на развој појачања за јединице и објекте. Архимаг је у већини јаких тактика изабран као први херој.

Посматрањем тока игре може се уочити да је акценат на затворенијем стилу игре, који се огледа у томе да се развија база и држи одбрана, све док се не достигне жељени ниво снаге потребан да се противник порази коначним нападом.

Поједини играчи можда ову тактику сматрају лошом, али треба имати у виду да постоје и играчи којима одговара такав стил игре. Победничке тактике које су креиране су такве да играч који нема просечно предзнање о томе против кога игра, добија добар отпор, а самим тим и забавну партију.

Током експеримента је уочено да се одређене тактике побољшавају све док се не постигне одређени врхунац. Њена ефикасност затим креће да опада до одређене тачке, након чега почиње са поновним растом, тј. креће се око локалног максимума.

6. Закључак и идеје за будућност

Основни циљ рада је усмерен ка томе да као крајњи резултат пружи одговор на два питања: да ли је могуће осмислити и реализовати еволутивни алгоритам за откривање нове тактике и стратегије у СВР играма и да ли ће и у којој мери новооткривене тактике и стратегије побољшати перформансе аутоматски вођених противничких играча?

Да бисмо пронашли одговор на задата питања, поставили смо следеће подциљеве:

1) Избор репрезентативне игре и алата за реализацију експеримента

Као репрезентативна игра изабран је Warcraft III, једна од најбољих СРВ игара, која поред своје симболичке вредности поседује и моћан алат, World Editor, који омогућава креирање нових и уношење измена у постојеће скрипте.

2) Имплементација система који применом статичног учења помоћу еволутивних алгоритама, открива нове стратегије и тактике у СВР играма, које су ефикасније од полазног скупа ручно конструисаних стратегија.

Креиран је програм коришћењем концепта генетских алгоритама, који применом укрштања и мутације мења постојеће и креира нове стратегије.

Имајући у виду да су задати циљеви постигнути, и да програм успешно мења и креира стратегије игре, може се рећи да је одговор на прво питање потврдан. Могуће је осмислити еволутивни алгоритам за креирање нових, успешних стратегија у СВР играма. Такође, како се експериментом показало да је дошло до значајног побољшања полазног скупа стратегија, одговор на друго питање је да су стратегије креиране од стране програма учиниле игру аутоматски вођеног противничког играча мање предвидивом и тиме повећали његове шансе за победу.

На основу резултата који су добијени може се закључити да је експеримент успешан, јер је добијено побољшање почетне генерације тактика. Ако се имају у виду радови који су рађени на исту или сличну тему, овим радом је показано да се еволутивни алгоритми могу применити и на новијим играма. Међутим, све је још увек у фази експеримента, на коме је потребно радити и даље како би се дошло до коначне реализације.

Једна од отежавајућих околности при извођењу експеримента је само тестирање и оцењивање добијених стратегија (израчунавање њихове функције прилагођености). За сваку стратегију је морао ручно да се покрене тест и да се запишу резултати. Такође је било потребно и сачекати док се партија одигра, која (у убрзаном режиму) траје од 5 до 10

минута. То је фактор који је значајно ограничио обим експеримента и због којега је било могуће креирати само 10 генерација јединки.

Након овог експеримента идеја за неки будући рад на овом истраживању била би да се у експеримент уврсти више игара са различитим тактикама, или да се експеримент спроведе тако што ће се користити играч као учесник. На тај начин би се добило више референци, али би због саме природе експеримента и промењљивости цео процес дуже трајао. Такође би била пожељна и имплементација аутоматског система пуштања и тестирања тактика. На тај начин би се у реално кратком времену и без присустава испитивача могао покренути велики број генерација.

Треба имати у виду и то да је компанијама које стварају игре приоритет профит који остварују, а како је за овакву врсту истраживања потребно време, можда ће се пре определити за технике које већ користе, иако се њима не постиже потенцијал који би те игре могле имати. Међутим, врло брзо ће доћи до крајње тачке експлоатације, па ће сходно жељи играча за нечим новим и бољим, морати да се окрену побољшањима.

Имајући у виду да савремена технологија напредује великом брзином и да постоје игре са графиком која брише линију између виртуелног и реалног, као и то да се развијају технологије попут Oculus Rifta, које нам дозвољавају да „уронимо“ у свет игре, сасвим је извесно да је рад усмерен ка даљем побољшању вештачке интелигенције неизбежан.

7. Додатак

Детаљан опис јединица, објеката и истраживања и њихово кодирање у програму.

ОБЈЕКТИ		
објекат	код	Намена
Town Hall	1	Главна зграда, користи се за тренирање сељака
Keep	2	Надоградња главне зграде, отвара нове објекте и истраживања
Castle	3	Последња надоградња главне зграде, отвара све и истраживања
Barracks	4	Служи за тренирање пешадинаца, артиљераца и коњаника
Lumbermill	5	Користи се за одлагање дрва и истраживање
Blacksmith	6	Користи се за истраживање оружија и оклопа јединица
Arcane Sentry	7	Тренира свештенике и чаробнице
Workshop	8	Тренира јединице за рушење објеката
Gryphon Aviary	9	Тренира јахаче грифона
Farm	10	Повећава капацитет јединица које могу да се праве
Altar	11	Призива хероје
Tower	12	Извиђача кула
Guard Tower	13	Кула са стрелама
Cannon Tower	14	Кула са топовима

Табела 3.

ЈЕДИНИЦЕ		
војник	код	Намена
Footman	16	Основна јединица, пешадинац
Rifleman	17	Артиљерац, ефикасан против летећих јединица
Knight	18	Коњаник, веома јака јединица
Sorceress	19	Чаробница, може да успори непријатеља, постане невидљива и претвори непријатеља у овцу
Priest	20	Свештеник, лечи рањене јединице и скида зле чаролије
Gyrocopter	21	Летећа машина, ефикасна у истраживању мапе и против летећих јединица
Mortar Team	22	Топџија, спора и рањива јединица али веома ефикасна против објеката
Steam Tank	23	Возило са јаким штитом и одлично против објеката. Слабо против осталих јединица
Gryphon Rider	24	Јахач грифона, летећа јединица јака против објеката и осталих јединица

Табела 4.

ИСТРАЖИВАЊА		
Истраживање	код	Намена
Defend	25	Повечава отпорност пешадинаца на оштре нападе
Long Rifles	26	Повећава даљину пуцања артиљераца
Animal War Traning	27	Повећава животну енергију коњаника и јахача грифона
Iron Plating	28	Појачава оклоп јединица за блиску борбу
Iron Forged Swords	29	Појачава напад јединица за блиску борбу
Studed Leather Armor	30	Појачава оклоп јединица за борбу на даљину
Black Gunpowder	31	Појачава напад јединица за борбу на даљину
Steel Plating	32	Појачава оклоп јединица за блиску борбу
Steel Forged Swords	33	Појачава напад јединица за блиску борбу
Reinforced Leather Armor	34	Појачава оклоп јединица за борбу на даљину
Refined Gunpowder	35	Појачава напад јединица за борбу на даљину
Mithril Plating	36	Појачава оклоп јединица за блиску борбу
Mithril Forged Swords	37	Појачава напад јединица за блиску борбу
Dragonhide Armor	38	Појачава оклоп јединица за борбу на даљину
Imbuned Gunpowder	39	Појачава напад јединица за борбу на даљину
Improved Lumber Harvesting	40	Повећава количину дрва које сељаци прикупљају
Improved Masonery	41	Повећава отпорност објеката на нападе
Sorceress Adept Training	42	Повећава све атрибуте чаробнице и даје јој магију невидљивости
Priest Adept Traning	43	Повећава све атрибуте свештеника
Flying Machine Bombs	44	Омогућава летећим машинама да нападају копнене јединице
Flare	45	Открива део мапе
Magic Sentry	46	Даје кулама способност да виде невидљиве јединице
Sorceress Master Training	47	Повећава све атрибуте чаробнице и даје јој да претвара непријатеље у овце
Priest Master Traning	48	Повећава све атрибуте свештеника
Advanced Masonery	49	Повећава отпорност објеката на нападе
Imbuned Masonery	50	Повећава отпорност објеката на нападе
Advanced Lumber Harvesting	51	Повећава количину дрва које сељаци прикупљају
Storm Hammers	52	Напад јахача грифона удара више јединица

Табела 5.

8. Литература

1. **Rollings A. and Adams E.,** *Andrew Rollings and Ernest Adams on Game Design,* New Riders Publishing, pp. 321 - 345, (2003), ISBN 1-59273-001-9.

2. **Laird J. E. Lent.,** *Using a Computer Game to Develop Advanced AI,* IEEE Computer Society Press, Vol. 34, pp. 70-75, (2001).

3. **Tozour P.,** The Perils of AI Scripting. *AI Game Programming Wisdom (ed. S. Rabin),* Charles River Media, pp. 541-547, (2004).

4. **Spronck P. Sprinkhuizen-Kyper I. and Postma E.,** *Enhancing the Performance of Dynamic Scripting in Computer Games (ed. M. Rauteberg),* Springer, IFIP International Federation for Information Processing. Vol. LINCS 3166, (2004)

5. **Јаничић П. and Николић М.,** *Вештачка интелигенција,* скрипта, Математички факултет, Београд, пп. 147-157, (2010)

6. **Ch'ng and Huat S.,** *Evolutionary programming for automatically generating strategies in real-time strategy games,* Master thesis, University Malasia, Sabah, (2010)

7. **Витас М. Д.,** *Преводиоци и Интерпретатори,* Математички факултет*, пп. 49-59*, (2004)

8. **Belzer J. Holzman G. A. and Kent A.,** *Encyclopedia of Computer Science and Technology,* CRC Press, Vol. 25, (1975), ISNB 0-8247-2275-2.

9. **Rabin S.,** *AI Game Programming Wisdom,* Charles River Media, pp. 557-556, (2002), ISBN 978-1584500773.

10. **Darvin C.,** *The origin of spicies,* Bantam Classics; Reissue edition, ISBN 978-0553214635, (1999)

11. **Holland J.,** *Adaptation in Natural and Artificial Systems. An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence,* The University of Michigan Press, ISBN 978-0262581110, (1992)

12. **Muhlenbein H.,** *How Genetic Algorithms Really Work: Mutation and Hillclimbing,* PPSN, Vol. 92, (1992)

13. **Rusell S. and Norvig P.,** *Artificial Intelligence A Modern Approach (3rd Edition),* Pearson, pp. 693-830, ISBN 978-0-13-604259-4, (2010)

14. **Geryk B.,** A History of Real-Time Strategy Games, *GameSpot,* [На мрежи]

15. **Rabin S.** *Ai Programming Wisdome 2,* Charles River Media, pp. 457-466, ISBN 978-1584502890, (2003)

16. **Blizzard Entertainment,** *Warcraft III Manual, (2002)*