

Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

DIPLOMOVÁ PRÁCE

Bot pro Starcraft: Broodwar



2015

Bc. Radomír Škrabal

Vedoucí práce: Mgr. Petr Osička,
Ph.D.

Studijní obor: Informatika, prezenční
forma

Bibliografické údaje

Autor: Bc. Radomír Škrabal
Název práce: Bot pro Starcraft: Broodwar
Typ práce: diplomová práce
Pracoviště: Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby: 2015
Studijní obor: Informatika, prezenční forma
Vedoucí práce: Mgr. Petr Osička, Ph.D.
Počet stran: 47
Přílohy: 1 CD/DVD
Jazyk práce: český

Bibliographic info

Author: Bc. Radomír Škrabal
Title: A Starcraft: Broodwar bot
Thesis type: master thesis
Department: Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense: 2015
Study field: Computer Science, full-time form
Supervisor: Mgr. Petr Osička, Ph.D.
Page count: 47
Supplements: 1 CD/DVD
Thesis language: Czech

Anotace

Práce pojednává o vývoji a architektuře bota Princess Of Blades pro strategickou hru Starcraft: Brood war. Stručně seznamuje čtenáře se hrou a některými dalšími podobnými projekty, rovněž se věnuje návrhu a implementaci bota a několika experimentům s jeho výkonem.

Synopsis

Thesis presents developement and architecture of Starcraft: Brood war bot Princess Of Blades. It introduces reader into Starcraft: Brood war and to several similar bot projects. It also explains Princess Of Blades desing and implementation and presents several experiments.

Klíčová slova: umělá inteligence, starcraft; bot

Keywords: artifical intelligence; starcraft; bot

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce

podpis autora

Obsah

1	Úvod	8
1.1	Motivace	8
1.2	Starcraft: Brood war	9
1.3	Další boti pro Starcraft:Broodwar	11
1.4	Některé pojmy ve Starcraft: Brood war	11
2	Návrh bota	15
2.1	Struktura bota	15
2.1.1	Příklady komunikace ve struktuře bota	16
2.2	Zvolené prostředky a technologie	19
3	Uživatelská příručka	20
3.1	Princess Of Blades	20
3.2	Instalace Princess Of Blades	20
3.3	Hraní Princess Of Blades proti vestavěné AI	21
3.4	Hraní Princess Of Blades proti člověku na jednom počítači	22
3.5	Hraní Princess Of Blades proti jiným botům	24
3.6	Nastavení Princess Of Blades	25
4	Implementované algoritmy	27
4.1	Bayesovská síť	27
4.1.1	Maximal-Likelihood Estimation (MLE)	28
4.1.2	Gibbsův sampler	28
4.1.3	Technická řešení	31
5	Implementační detaily	33
5.1	Reprezentace build orderů a související nástroje	33
5.2	Hierarchy velení a zasílání zpráv	34
5.3	Ekonomický hard-skript	35
5.4	Taktický hard-skript	37
5.5	Naivní micromanagement	38
5.6	Reprezentace znalostí o protivníkovi	38
6	Experimenty a výkon bota	39
6.1	Výkon proti vestavěné AI	39
6.1.1	Výtězná ZvP	40
6.1.2	Prohra v TvZ	41
6.2	Výkon proti jiným botům	42
6.2.1	Prohra proti MaasCraft	42
6.3	Výkon proti živým hráčům	43
6.3.1	Prohra proti Martinu Hořavovi	43

7 Závěr	45
7.1 Shrnutí	45
7.2 Budoucí práce	45
Seznam zkratk	46
Literatura	47

Seznam obrázků

1	Starcraft: Brood war	10
2	Struktura Princess Of Blades	16
3	Soubor bwapi.ini upravený pro hraní proti člověku na jednom počítači	23
4	Chaoslauchner připravený ke spuštění více instancí Starcraft: Brood war	24
5	Několik instancí Starcraft: Brood war běžících současně.	25
6	Bayesovská síť pro ZvZ matchup	28
7	Bayesovská síť pro ZvP matchup	29
8	Bayesovská síť pro ZvT matchup	29
9	Aplikace BOBuilder	35

Seznam tabulek

1	Vybrané jednotky Starcraft: Brood war	13
2	Výkon Princess Of Blades proti vestavěné AI , bez požití Bayesovské sítě	39
3	Výkon Princess Of Blades proti vestavěné AI , se zapnutou Bayesovskou sítí	40
6	Výkon Princess Of Blades proti botovi MaasCraft, se zapnutou Bayesovskou sítí	42
8	Výkon Princess Of Blades proti živým hráčům	43

Seznam zdrojových kódů

1	MLE v Princess Of Blades	30
2	Gibbsův sampler	30

1 Úvod

Strategické videohry odehrávající se v reálném čase, [Real-Time Strategy \(RTS\)](#), jsou jedním z vůbec nejkomplexnějších žánrů videoher vůbec. Jedná se o komplexní prostředí, kde musí hráč na základě neúplných informací a v omezeném čase rozhodovat o dalším strategickém postupu na nízké (příkazy jednotkám), střední (kompozice armády) i vysoké úrovni (postup v technologickém stromu), musí vyvažovat ekonomickou (těžba surovin) a vojenskou (útok, obrana, průzkum) složku své strategie a navíc musí aktivně reagovat na strategii protihráče. Vytvořit umělou inteligenci schopnou hrát takovou komplexní videohru je atraktivní a obtížný úkol.

Většina současných [RTS](#) používá coby počítačové protihráče pouze v zásadě jednoduché, pevně naskriptované chování obsahující v lepším případě několik předem připravených strategií, obvykle podpořených podváděním ze strany [AI](#). Lidský hráč je obvykle schopen během několika málo her odhalit a překonat vestavěnou strategii a počítač porazit, další hraní potom je pouze opakováním naučeného vzoru a není ani zábavné ani nijak přínosné.

Vytvoření bota pro hraní [RTS](#) na lidské úrovni je v současné době předmětem intenzivního výzkumu. Frameworkem pro takovýto výzkum byla v minulosti hra Wargus, v současnosti však převládá zaměření na komerční strategickou hru Starcraft: Brood war¹ s opensource knihovnou [A Brood War API \(BWAPI\)](#), pomocí které lze poměrně snadno vytvářet počítačové hráče pro Starcraft: Brood war.

Cílem této práce bylo vytvořit bota pro Starcraft: Brood war tak, aby měl vhodnou přesně definovanou a rozšiřitelnou strukturu, mohl posloužit jako základ pro další práci a jako framework pro testování [RTS AI](#) algoritmů a tyto vlastnosti na něm demonstrovat.

Struktura práce je následující: Kapitola 1 shrnuje úvod, motivaci a další boty pro Starcraft: Brood war. V kapitole 2 se budu zabývat obecným návrhem architektury bota, zvolenými prostředky, v kapitole 3 uživatelskou příručkou a v kapitole 4 algoritmy implementovanými v Princess Of Blades. Kapitola 5 popisuje implementační detaily bota. Kapitola 6 obsahuje experimenty a výkon bota v současné podobě a kapitola 7 shrnuje celou práci a navrhuje další směry vývoje Princess Of Blades.

1.1 Motivace

Využití současných poznatků z výzkumu oblastí [AI](#) a machine learning ve vytváření počítačových protivníků moderních [RTS](#) jsou minimální až nulové. Důvodů je několik, v první řadě se jedná o zvyšování nákladů na audiovizuální zpracování videoher jako takových, kde na moderní [AI](#) nezbývá místo v rozpočtu.

¹StarCraft®: Brood War® ©1998 Blizzard Entertainment, Inc. All rights reserved. StarCraft, Brood War and Blizzard Entertainment are trademarks or registered trademarks of Blizzard Entertainment, Inc. in the U.S. and/or other countries.

Dále k situaci jistě přispívá, že vývoj sofistikovaných počítačových protivníků je nákladný proces s nejistým výsledkem. Navíc případná řešení třetích stran na rozdíl od jiných běžných součástí videoher, jako je například fyzika nebo přehrávání multimédií, na trhu prakticky neexistuje.

V rámci výzkumu lze nalézt poměrně velké množství různých botů pro Starcraft: Brood war. Jedná se o projekty různé velikosti, vytvářené v rámci výzkumných pracovišť. Tyto projekty nezřídka implementují sofistikované algoritmy s dobrými výsledky, ale architektura samotných botů je často proprietární a jednoúčelová. Často (až na skutečně velké projekty jako např. UAlberta Bot[1]) jim chybí celkový rámec počítačem řízeného hráče, který by byl alespoň nějakým základním způsobem schopen spravovat i ty části hry, pro které není určeno ono sofistikované řešení.

Motivací k vytvoření Princess Of Blades byla tedy snaha navrhnout nějaký vhodný obecný rámec, který by rozdělával a alespoň naivním způsobem spravoval všechny aspekty RTS hry Starcraft: Brood war. Původní myšlenka byla rozdělit všechny úkony které hráč obvykle při hraní RTS provádí a hierarchicky je uspořádat do vzájemně komunikující infrastruktury. Inspirací byla v tomto mimo jiné právě architektura UAlberta Bot [1].

1.2 Starcraft: Brood war

Starcraft: Brood war je RTS videohra vydaná v roce 1998 firmou Blizzard Entertainment. V následující dekádě se tato hra etablovala jako kanonický vzor pro většinu RTS pozdějších generací a rovněž jako jeden z nejpobulárnějších a nejprestižnějších disciplín v progamingu (e-sportu). Za toto vděčí zejména svému designu postavenému na třech diametrálně odlišných hratelných stranách, jejich vyváženosti a množství různých hratelných strategií za každou z nich. S příchodem opensource knihovny BWAPI, pak nahradila starší Wargus na poli výzkumu umělé inteligence pro RTS.

Příběh Starcraft: Brood war se odehrává ve vzdálené budoucnosti, ke o nadvládu soupeří tři rasy:

- *Terran* Lidé, lidská rasa ve světě Starcraft: Brood war. Jedná se o výrazně defenzivně zaměřenou rasu s mocnou statickou obranou a unikátní schopností opravovat a léčit všechny typy svých jednotek.
- *Zerg* Mimosemská rasa vzdáleně inspirovaná vetřelcem. Na rozdíl od terranů jsou zergové výrazně zaměřeni na útok, velké počty levnějších jednotek a vysokou mobilitu a rychlost. Nedokážou opravovat své jednotky, ale ty se samy během času regenerují. Základny (s několika výjimkami) mohou stavět pouze na území, které předem asimilovali (creep). Na rozdíl od ostatních dvou ras své jednotky nevyrábějí v továrnách, ale mutují je z larev, které produkuje hlavní budova, která jich má v daném okamžiku omezený počet.
- *Protoss* Pokročilá rasa mimosemských telepatů. Jsou rovněž zaměřeni spíše na útok, ale spoléhají se hlavně na silné a drahé (a často poměrně pomalé)



Obrázek 1: Starcraft: Brood war

jednotky v omezených počtech. Všechny jejich jednotky disponují štíty, které se v čase obnovují, ale je-li štít vypotřebován a zasažena samotná jednotka, pak toto poškození opravit nelze. Jejich základny lze stavět pouze v okolí speciálních budov (pylon), které jim poskytují energii. Pylony lze stavět kdekoli, ale pokud nejsou umístěny redundantně představují zranitelnou část protosské infrastruktury.

Starcraft: Brood war byl více než dekádu jednou z vedoucích disciplín progamingu (e-sportu). Společně s dalšími videohrami (Quake 3: Arena, Counter-Strike a další) byl jedním z titulů, které pomohly se progamingu etablovat a získat masovou popularitu. V dnešní době již existují profesionální hráči i týmy (v žargonu e-sportu obvykle nazývané klany) financované výhradně od sponzorů a výher v turnajích. Samotný Starcraft: Brood war byl v e-sportu nahrazen až svým sequelem Starcraft II: Wing of Liberty² v roce 2010, do té doby neměl v rámci e-sportu RTS Starcraft: Brood war vážnější konkurenci.

²V současné době je používán samostatný datadisk/pokračování Heart of the Swarm a zanedlouho by měl být podobným způsobem vydán další s podtitulem Legacy of the void, nicméně všechny tři samostatně hratelné hry jsou považovány shodně za druhou generaci

1.3 Další boti pro Starcraft: Brood war

Botů pro Starcraft: Brood war založených na [BWAPI](#) existuje poměrně velké množství a není účelem zde vyjmenovat všechny. Zaměřím se zejména na projekty, které mě nějakým způsobem inspirovaly nebo ovlivnily návrh mého bota Princess Of Blades.

- *UAlberta Bot* Bot vyvíjený na kanadské University of Alberta je jistě jedním z technicky a designově nejpokročilejších botů pro Starcraft: Brood war vůbec. Zahrnuje v sobě velké množství pokročilých algoritmů (například Search-Based [AI](#) for Build Order Planning, Real-time simulation for Combat scenario Decisions, kompletní výčet a více detailů najdete na [\[1\]](#)). Rovněž se pravidelně umísťuje na předních příčkách turnajů botů ve Starcraft: Brood war. Právě jeho návrh byl velkou inspirací pro strukturu Princess Of Blades.
- *Overmind* Další z botů vyvíjený v akademickém prostředí tentokrát Univerzitou v Berkeley [\[2\]](#). Podobně jako UAlberta Bot implementuje sofistikované algoritmy a dosahuje špičkových výsledků v turnajích i v experimentech proti lidským hráčům. Oproti UAlberta Bot se však jeho návrh zdá poněkud neintuitivní.
- *EISBot* Vyvíjený organizací [Extensive Intelligence Studio \(EIS\)](#), Santa Cruz [\[3\]](#). Jedná se rovněž o bot s poměrně čistou a logickou strukturou, zaměřující se zejména Goal-Driven algoritmy a v poslední době i na reaktivní plánování.

Jak jsem již naznačil nejedná se zdaleka o vyčerpávající výčet z, dalších botů pro Starcraft: Brood war zmíním například Skynet, [Trobar Clus Starcraft \(TSC\)](#), KillerBot a [Artificial Intelligence Using Randomness \(AIUR\)](#).

1.4 Některé pojmy ve Starcraft: Brood war

V dalším textu jsou použity některé pojmy z prostředí okolo Starcraft: Brood war jejichž význam nemusí být na první pohled zřejmý. Tato kapitola je vysvětluje.

Budovy všech ras se ve Starcraft: Brood war dělí na *základní* (*basic structures*) a *pokročilé* (*advanced structures*). Rozdíl je čistě v uživatelském rozhraní, kde jsou jednotlivé skupiny budov v různých podmenu, nicméně toto rozdělení se používá při definici některých dalších pojmů.

Replay je zaznamenaný zápas ve Starcraft: Brood war. Replaje se zaznamenává přímo engine hry a ukládají se v proprietárním binárním formátu. Struktura tohoto formátu není známá a přehrávání v engine hry je jediný známý způsob jak z nich získávat data. Toto přehrávání je však omezeno maximální rychlostí engine.

Caster je typ jednotky která obvykle vůbec neútočí, ale má k dispozici několik speciálních ručně aktivovaných schopností určených pro podporu boje, které

spotřebovávají body energie. Casteři rasy Terran jsou *Medic* a *Science Vessel*, pro Zerg jsou to *Queen* a *Defiler* a pro Protoss jsou to *High Templar* a *Arbiter*. Za castery se obvykle nepovažují jednotky, které vyžadují alespoň jeden výzkum, aby mohly využívat svoje body energie (*Wraith*, *Ghost*, *Battlecruiser*, *Corsair*).

Zápas ve Starcraft: Brood war se dělí na tři části. *Early game*, což je začátek zápasu, obvykle zaměřený na budování ekonomiky, kdy hráči mají pouze menší počty základních jednotek. Za konec early game je obvykle považován moment, kdy první hráč dokončí první pokročilou budovu. *Mid game* následuje po early game a je to fáze, kdy se střetávají oba hráči o kontrolu nad mapou (zejména o expandy) se středně pokročilými jednotkami. Mnoho zápasů ve Starcraft: Brood war skončí už ve fázi mid game. *Late game* je pozdní fáze zápasu, kdy se některý z hráčů dostal na některý z vrcholů technologického stromu a vyznačuje se použitím masivního množství jednotek a vysoce pokročilých jednotek včetně casterů každé ze stran.

Build Order označuje pořadí v jakém hráč v zápase Starcraft: Brood war staví budovy a jednotky. Build Orderů je velké množství a navíc se stále upravují a vyvíjí, obecně lze build ordery dělit podle toho, pro kterou část zápasu jsou určeny. Build ordery určené pro early-game se někdy nazývají tvz. *opening*.

Hráč může ve Starcraft: Brood war provádět dva typy globálních vylepšení své armády. Jsou to *technologie* a *upgrady*. Rozdíl spočívá v tom, že technologie otevírají jednotkám nové schopnosti, zatímco upgrady nějakým způsobem posilují schopnosti stávající (rychlost, sílu, výdrž, zásobu energie apod.)

Suroviny používané ve Starcraft: Brood war jsou *minerály* a *plyn*. Minerály jsou základní surovinou, která je potřeba k výrobě všech jednotek a budov a nachází se na mapách v konečném (i když obvykle dost velkém) množství. Plyn je vedlejší surovina používaná pro výrobu pokročilých jednotek a budov, výzkum technologií a provádění upgradů. Obvykle čím je jednotka pokročilejší, tím více plynu potřebuje.

Dělník (worker) základní jednotka pro sběr surovin a stavbu budov. Na začátku zápasu ve Starcraft: Brood war má hráč obvykle čtyři. Jsou to *SCV* za Terrany, *Drone* za Zergy a *Probe* za Protossy.

Populační limit (populace) je třetí zdroj používaný ve Starcraft: Brood war. Omezuje maximální počet jednotek, které může hráč najednou vyrobit. Populaci zvyšují resource depoty a pak za každou rasu jedna další budova nebo jednotka (Terran - *Supply Depot*, Zerg - *Overlord*, Protoss - *Pylon*). Maximální hodnota populace je 200.

Resource depot je základní budova, kterou má hráč společně s několika dělníky na začátku zápasu. Umožňuje výrobu dalších dělníků, ukládání nasbíraných surovin a obvykle poskytuje i populační limit. Resource depot je *Command Center* pro Terrany, *Hatchery* pro Zergy a *Nexus* pro Protossy.

Expand je označení pro resource depot, umístěný tak, aby dovoloval rychlou těžbu ze vzdálených zdrojů minerálů nebo plynu. Budování a umístění expandů je důležitou součástí build orderů zaměřených na mid nebo late game.

Detector je schopnost jednotky odhalit neviditelné nepřátelské jednotky. Za

každou rasu ji má alespoň jedna obraná věž a jedna pohyblivá jednotka. Absence (nepřátelského) detektoru v určité fázi hry je základem mnoha strategií.

V následující tabulce vyjmenovávám některé důležité jednotky a budovy ve Starcraft: Brood war. Obzvláště se zaměřím na ty, které použiji v následujícím textu.

Tabulka 1: Vybrané jednotky Starcraft: Brood war

Název jednotky	Rasa	Popis
Command Center	Terran	Resource depot
SCV	Terran	Worker
Refinery	Terran	Terranská verze budovy pro těžbu plynu.
Barracks	Terran	Základní budova pro výrobu vojenských jednotek. Umí vyrábět jednotky Marine, Firebat, Medik a Ghost.
Marine	Terran	Základní bojová jednotka. Umí útočit na dálku i na létající cíle.
Firebat	Terran	Pokročilá pěchota pro boj na blízko. V moderní hře Starcraft: Brood war se nepoužívá.
Medic	Terran	Caster s nízkými technologickými požadavky. Používá se v kombinaci s marine.
Factory	Terran	Pokročilá továrna pro výrobu pozemních jednotek.
Machine shop	Terran	Vylepšení pro factory. Umožňuje vyrábět sledge tanky
Siedge Tank	Terran	Těžká mid game jenotka s mocným dalekonosným útokem. Základ velkého množství terranských strategií.
Starport	Terran	Továrna pro výrobu leteckých jednotek.
Science Vessel	Terran	Pokročilý terranský caster a pohyblivý detektor.
Hatchery	Zerg	Resource depot a budova produkující larvy a v důsledku i všechny jednotky.
Drone	Zerg	Worker
Overlord	Zerg	Jednotka poskytující populaci, pohyblivý detektor a po upgradu i transportní jednotka.
Extractor	Zerg	Zergská verze budovy pro těžbu plynu
Lair	Zerg	Vylepšení hatchery. Umožňuje pokračovat v technologickém stromu. Dokončení stavby lair se někdy nazývá „přechod na Lair tech“.
Hive	Zerg	Vylepčení lair, umožňuje postoupit k nejpokročilejším jednotkám. Obdobně jako u lair „přechod na Hive tech“.

Spawning Pool	Zerg	Základní budova umožňující výrobu zerglingů a pokračování v technologickém stromě.
Zergling	Zerg	Základní bojová jednotka. Má útok pouze na blízko. Z každé larvy se najednou rodí dva.
Hydralisk Den	Zerg	Budova umožňující produkci jednotky hydralisk a po vylepšení i lurker.
Hydralisk	Zerg	Základní zergský střelec. Útočí jak na pozemní tak na létající jednotky. Účinný proti velkým jednotkám, Je potřeba k výrobě lurkerů.
Lurker	Zerg	Pokročilá zergská střelecká jednotka. Umí útočit pouze když je zahrabaný, ale pak se nemůže přesouvat. Když je zahrabaný je neviditelný. Základ mnoha zergských strategií.
Spire	Zerg	Pokročilá zergská budova umožňující výrobu létajících jednotek.
Mutalisk	Zerg	Základní zergská létající jednotka. Jeho cena a vlastnosti z něj dělají jednu z nejpoužívanějších jednotek ve hře za zergy. Základ mnoha zergských strategií.
Nexus	Protoss	Resource depot
Probe	Protoss	Worker
Assimilator	Protoss	Protosská verze budovy pro těžbu plynu
Gateway	Protoss	Základní továrna na jednotky.
Zealot	Protoss	Základní bojová jednotka. Útočí pouze na blízko. Je dražší, ale silnější než základní jednotky ostatních stran.
Citadel of Adun	Protoss	Technologická budova. Důležitá pro postup v technologickém stromu.
Cybernetics Core	Protoss	Technologická budova. Důležitá pro postup v technologickém stromu. Umožňuje produkci dragoonů.
Dragoon	Protoss	Základní střelec.
Templar Archives	Protoss	Technologická budova. Důležitá pro postup v technologickém stromu. Umožňuje produkci high templarů.
High Templar	Protoss	Základní caster.
Stargate	Protoss	Továrna na letecké jednotky.
Corsair	Protoss	Pokročilá letecká jednotka, součást velkého množství protosských strategií.

2 Návrh bota

Původní záměr při vytváření bota Princess Of Blades byl navrhnout bota tak, aby bylo co nejsnazší a nejrychlejší v něm implementovat a zkoušet různé algoritmy, pro různé úkony spojené s hraním hry. Rovněž bylo žádoucí vytvořit bota tak, aby bylo jednotlivé algoritmy nebo skripty možno snadno (při zachování rozhraní) navzájem třeba i za chodu vyměnit. To logicky vedlo na desing, který nějakým způsobem používá většina moderních botů pro Starcraft: Brood war, tedy samostatné hierarchicky uspořádané agenty, specializované pro jednotlivé úkony hraní hry.

2.1 Struktura bota

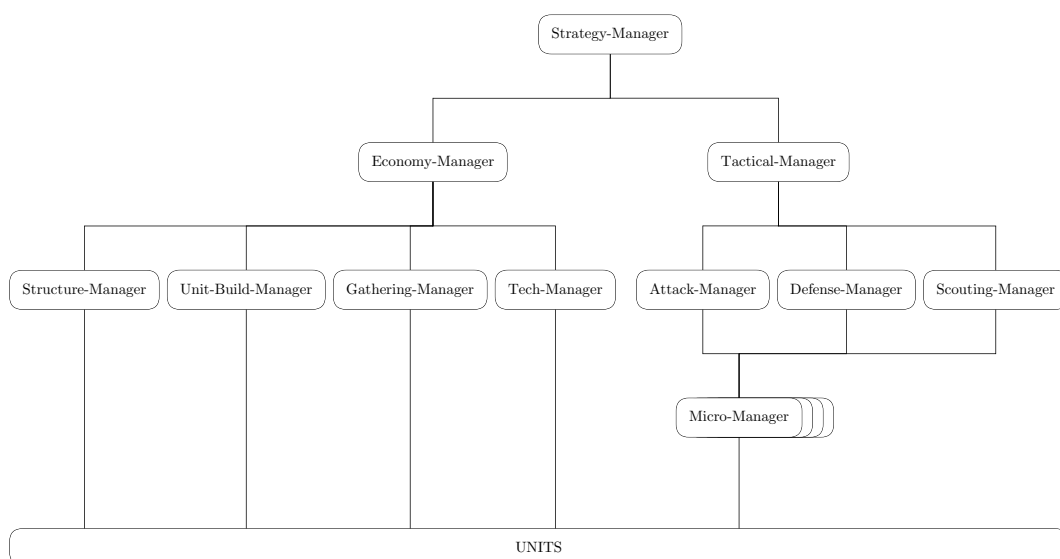
Jestliže jsme již provedli rozhodnutí, že bot založíme na agentech, musíme tyto agenty specifikovat a určit jim strukturu. Protože Starcraft: Brood war je v podstatě hra o velení armádě, použijeme armádní metaforu i pro popis struktury. Bot je rozdělen hierarchicky do jednotlivých stupňů velení od těch nejvyšších, kde se rozhoduje o globální strategii, až po ty nejnižší, kde se ovládají jednotlivé jednotky. Strukturu velení Princess Of Blades můžete vidět na obrázku 2.

Na nejvyšší úrovni velení figuruje pouze jeden agent a to *Strategy Manager*. Pokud se přidržíme metaforu vojenského velení, pak o něm lze uvažovat jako o nejvyšším veliteli nebo, chcete-li, ústřední vládě. Strategy Manager rozhoduje o celkové strategii (zejména build-orderu), o vyráběných jednotkách, a v důsledku i načasování útoků. Rovněž je centrálním uzlem, co se týče shromažďování informací o protivníkovi a bojišti, a poslední instancí při řešení požadavků podřízených agentů.

Na druhé úrovni nalezneme dva agenty specializované na dvě stěžejní části Starcraft: Brood war a to *Economy Manager* pro ekonomiku a *Tactical Manager* pro boj. Můžeme se na ně dívat jako na vysoké důstojníky armády (plukovníky nebo nižší generály), kteří velí každý své části armády. Economy Manager má přímý dohled nad výrobní frontou a využitím surovin a skrze podřízené agenty pak i na sběr surovin, výstavbu základny, výrobu jednotek a výzkum. Tactical Manager přímo rozhoduje o cílech útoků a skrze podřízené agenty pak kontroluje útok, obranu a průzkum.

Agenti třetí úrovně se dají rozdělit do dvou skupin podle toho jestli jsou podřízeny Economy Managerovi nebo Tactical Managerovi. Z pohledu hierarchie bychom je mohli nazvat důstojníky armády. V některých případech tito agenti přímo velí (ovládají) jednotlivé jednotky na bojišti (zejména v případě pod-ekonomických agentů), jindy ještě využívají následující poslední vrstvy.

Mezi pod-ekonomické agenty patří *Structure Manager* starající se o výstavbu základny, expandů a obraných věží. Dále *Unit Build Manager* řídící výrobu (a umístění výroby - kde se která jednotka vyrobí) jednotek, *Gathering Manager* zajišťuje těžbu surovin a maximální využití dělníků a *Tech Manager* spravuje laboratoře, výzkum a vylepšování jednotek.



Obrázek 2: Struktura Princess Of Blades

Pod-taktičtí agenti jsou tři a to *Attack Manager* určující složení armády pro útok a shromaždiště, *Defense Manager* zajišťující obranu základen a *Scouting Manager* provádějící průzkum.

Poslední vrstvou velení je potom agent nebo spíše rodina agentů zvaní *Micro Manager*. Jedná se o agenty, kteří přímo na nízké úrovni pracují s jednotkami a ovládají je. Ve vojenské metafoře bychom je nazvali poddůstojníky. Určují jednotkám cíle a priority, ale i to jak jich mají dosáhnout, které schopnosti použít, kdy útočit a kdy se přesouvat.

Ještě je nutné zmínit komunikaci mezi jednotlivými agenty. Komunikace se-shora dolů je ve formě rozkazů a agenti níže v hierarchii nemají možnost rozkazů neuposlechnout. Komunikace v opačném směru je méně častá a obvykle se jedná o různé druhy žádostí (žádost o přidělení jednotek na útok apod.), tyto požadavky pak posuzují přímo nadřízení agenti a je na nich, zda je vyřídí nebo zamítnou, popřípadě je mohou ponechat na vyšší instanci.

Pro konkrétní technické řešení jednotlivých agentů a komunikace mezi nimi viz. kapitola 5.

2.1.1 Příklady komunikace ve struktuře bota

Abych lépe osvětlil vzájemné vztahy a hierarchii použitou v Princess Of Blades, uvedu zde dva příkladny komplexnějších úkonů a rozhodnutí, na kterých musí vzájemně spolupracovat několik agentů tak, jak se provádějí v naivní implementaci. Konkrétně budu popisovat výrobu jednotky lurker pro ekonomickou část Princess Of Blades a útok nashromážděnými jednotkami v taktické části.

Lurker je jednotka strany zergů používaná v mnoha strategiích (viz. kapitola 1.4). Je specifická v tom, že na rozdíl od většiny ostatních jednotek se nevyrábí mutací přímo z larvy, ale mutuje se z jiné bojové jednotky - hydraliska. Nastává

tedy střet zájmů, kdy chceme aby byl hydalisk automaticky přiřazen do bojové části bota jako bojeschopná jednotka, ale zároveň chceme vyrábět lurkery a tak potřebujeme, aby kontrolu nad hydaliskem převzal Unit Build Manager. Tato situace se v naivní implementaci Princess Of Blades řeší následovně.

1. Strategy Manager rozhodne, že bude vyroben lurker.
2. Economy Manager zařadí lurkera do výrobní fronty. Jakmile je lurker na vrcholu fronty a hráč má dostatek surovin na jeho výrobu pošle požadavek na výrobu lurkera do Unit Build Manager.
3. Unit Build Manager zjistí, že lurker je jednotka která se mutuje z hydaliska. Zkontroluje jestliže má hydaliska k dispozici.
 - (a) Jestliže má k dispozici hydaliska, promění ho na lurkera.
 - (b) Jinak výrobu odloží a pošle svému nadřízenému zprávu `hydraRequest`.
4. Economy Manager přijme zprávu `hydraRequest`. Jelikož pro tuto zprávu nemá definovanou obsluhu přepoše ji svému nadřízenému.
5. Strategy Manager přijme `hydraRequest`. V reakci na to přikáže agentovi Tactical Manager uvolnit hydaliska.
6. Tactical Manager postupně přikazuje svým podřízeným agentům uvolnit hydaliska.
7. Podtaktičtí agenti přikazují svým pořízeným mikromanagerům uvolnit hydaliska.
8. Mikromanager prochází postupně všechny svoje jednotky a hledá jednotku typu hydalisk, která nemá žádné rozkazy. Jestliže takovou jednotku najde, uvolní ji a předá nadřízenému agentovi.
9. Pokud bylo hledání úspěšné, je postupně hydalisk předán až na úroveň agenta Strategy Manager. Pokud nebylo úspěšné, je tento stav signalizován.
 - (a) Jestliže byl hydalisk nalezen, předá se Economy Manageru a ten jej předá Unit Build Manageru a v další iteraci se proběhne krok [3a](#).
 - (b) Jestliže nebyl hydalisk nalezen je nutné jej vyrobit a zapamatuje si požadavek Unit Build Manager na hydaliska.
10. Strategy Manager rozhodne, že bude vyroben hydalisk.
11. Economy Manager zařadí hydaliska do výrobní fronty. Jakmile je hydalisk na vrcholu fronty a hráč má dostatek surovin na jeho výrobu, pošle požadavek na výrobu hydaliska do Unit Build Manager.

12. `Unit Build Manager` zjistí, že hydralisk je jednotka, která se mutuje z larvy. Zkontroluje jestli má larvu k dispozici.
 - (a) Jestliže má larvu k dispozici, použije ji k mutaci hydraliska.
 - (b) Jinak odsune požadavek na dobu, kdy bude mít k dispozici larvu³.
13. Je vytvořen hydralisk a předán agentovi `Starategy Manager`.
14. `Strategy Manager` zjistí, že má od agenta `Unit Build Manager` prioritní požadavek na hydraliska. Pak se chová, jako by byl hydralisk nalezen v bodě 9a.

Načasování a provedení útoku je netriviální akce, kterou musí každý bot hrávající *Starcraft: Brood war* zvládnout. Následuje popis, jak útok řeší naivní implementace *Princess Of Blades*.

1. `Strategy Manager` se periodicky dotazuje agenta `Tactical Manager`, jestli je připraven k útoku.
2. `Tactical Manager` je připraven k útoku, jestliže celková síla všech jednotek jeho podřízených agentů větší, než síla nejslabší známé základny protivníka.
3. Jestliže `Tactical Manager` odpoví na dotaz o připravenosti k útoku kladně, pak `Strategy Manager` vydá rozkaz k útoku.
4. `Tactical Manager` vybere cíl útoku (nejslabší známou nepřátelskou základnu) a předá jej s rozkazem k útoku agentu `Attack Manager`.
5. `Attack Manager` porovná sílu jednotek které má k dispozici, se silou základny na kterou má útočit.
 - (a) Jestliže má dostatek jednotek, shromáždí je a zaútočí.
 - (b) Jinak pošle zprávu `requestUnits` svému nadřízenému agentovi.
6. `Tactical Manager` přijme zprávu `requestUnits`. V reakci na ni si vyžádá všechno volné jednotky svých podagentů a přidělí je žadateli.
7. Jestliže má `Attack Manager` dostatek jednotek přejde na 5a, jinak opakuje 5b.

³Larvy se generují automaticky v budově hatchery

2.2 Zvolené prostředky a technologie

V této kapitole se věnuji prostředkům a knihovnám použitým při vytváření Princess Of Blades.

- *BWAPI*: Knihovna [BWAPI](#) je základním kamenem všech výše popsaných botů pro Starcraft: Brood war. Jedná se o komunitní opensource projekt, který na základě reverse engineeringu a injektování dll knihoven vytvořil funkční knihovnu a sadu nástrojů pro tvorbu botů pro Starcraft: Brood war. Samotná Princess Of Blades používá [BWAPI](#) verze 4.0, ale některé doprovodné projekty používají z technických důvodů starší verzi 3.7. Podrobnosti o [BWAPI](#) naleznete na jejích stránkách Google Code [\[4\]](#) nebo aktuálně na GitHub [\[5\]](#).
- *DLib*: Opensource knihovna DLib je obecná multiplatformní knihovna pro C++. V Princess Of Blades a doprovodných projektech se používá její reprezentace bayesovských sítí a operace nad nimi. Více o knihovně najdete na [\[6\]](#).
- *easyLogging++*: Multiplatformní opensource single header knihovna pro logování. Více na [\[7\]](#).
- *pugiXml*: Knihovna používaná v Princess Of Blades a doprovodných projektech pro parsování Xml souborů. Více na [\[8\]](#).

3 Uživatelská příručka

Tato kapitola je určena pro používání bota v jeho základní podobě. Obsahuje základní informace o Princess Of Blades, nastavení bota a návody, jak uvést bota do provozu v různých módech (bot proti vestavěné [AI](#), bot proti botovi, bot proti živému hráči na jednom počítači). Všechny následující informace a postupy byly otestovány na Windows 7 Profesional 64-bit, ale měly by fungovat na libovolné verzi operačního systému, který podporuje [BWAPI](#) 4.0. resp. Starcraft: Brood war 16.1.

Pro podrobnější informace o implementaci je k dispozici kapitola [5](#) a [4.1.3](#) nebo okomentované zdrojové kódy všech projektů, jenž jsou součástí této práce a detailní dokumentace ve formátu .pdf a html. Všechny tyto zdroje naleznete na přiloženém DVD.

3.1 Princess Of Blades

Princess Of Blades je bot založený na [BWAPI](#) 4.0 napsaný v jazyce C++ za použití několika externích knihoven (viz. [2.2](#)). Princess Of Blades je určen ke hraní kompletních zápasů v Starcraft: Brood war za rasu Zerg (za jiné rasy nebude fungovat správně, pokud vůbec) na libovolné zcela průchozí mapě⁴. Princess Of Blades v základu funguje ve dvou módech se zapnutou a bez zapnuté Baysovské kvalifikace protivníka. Princess Of Blades umožňuje snadné přidávání a testování nových strategií pro bota pomocí přidávání souborů build orderů do adekvátních složek (viz. [5.1](#)). V současné verzi obsahuje přes deset strategií proti každé hratelné straně.

3.2 Instalace Princess Of Blades

Instalace bota Princess Of Blades může být mírně komplikovaná. Pro úspěšnou instalaci doporučuji řídit se následujícími kroky.

1. Nainstalujte Starcraft: Brood war a updatujte jej na verzi 16.1⁵.
2. Nainstalujte [BWAPI](#) 4.0.
3. Necht je instalační složka Starcraft: Brood war <starcraft> a instalační složka [BWAPI](#) <bwapi>.
4. Do složky <starcraft>/bwapi-data/AI/ nakopírujte soubor PrincessOfBlades.dll.
5. Otevřete soubor <starcraft>/bwapi-data/bwapi.ini.

⁴Na mapách, kde nelze dosáhnout pozemními jednotkami expandy nebo nepřátelskou základnu, nelze správně fungování bota zaručit.

⁵Současné distribuce Starcraft: Brood war jsou již obvykle updatovány na tuto verzi hned po instalaci

- (a) Změňte hodnotu `ai` na
`bwapi-data/AI/PrincessOfBlades.dll`
(výsledek `ai = bwapi-data/AI/PrincessOfBlades.dll`).
- (b) Změňte hodnotu `race` na `Zerg` (výsledek `race = Zerg`).
- (c) Změňte hodnotu `game_type` na `FREE_FOR_ALL`
(výsledek `game_type = FREE_FOR_ALL`).
- (d) Alternativně můžete použít soubor `bwapi.ini` z příloženého DVD.

6. Tímto je Princess Of Blades nainstalován a připraven k hraní.

3.3 Hraní Princess Of Blades proti vestavěné AI

Toto je základní funkce Princess Of Blades, na které se testovaly všechny funkce. Princess Of Blades je schopna hrát proti libovolné straně zápas 1v1. Pokud jste se řídili instalačními pokyny (viz. 3.2), je Princess Of Blades k tomuto hraní již připravená. Proto zde jenom upřesním některá nastavení která se týkají BWAPI a její funkce menu automation.

- Veškerá nastavení se provádějí v ini souboru `<starcraft>/bwapi-data/bwapi.ini`.
- Položka `auto_menu` zapíná a nastavuje automatizaci menu Starcraft: Brood war. Pro hraní Princess Of Blades proti vestavěné AI se používá hodnota `SINGLE_PLAYER` (výsledek `auto_menu = SINGLE_PLAYER`).
- Položka `auto_restart` se používá pro automatické restartování hry po skončení zápasu, při experimentech je užitečné nastavovat na ON, ale pro běžné hraní je lepší ponechat defaultní OFF.
- Položka `map` specifikuje relativní cestu k mapě (branou od kořenového adresáře Starcraft: Brood war), kterou bude bot hrát. Defaultní hodnota zvolí náhodnou mapu ve složce `<starcraft>/maps/`.
- Položka `enemy_race` určuje rasu, proti které má bot hrát. Možnosti jsou `Terran`, `Protoss`, `Zerg`, `Random`, `RandomTP`, `RandomTZ`, `RandomPZ`. Nicméně podle mých zkušeností toto nastavení nefunguje vždy spolehlivě a je lepší nastavit přímo položky `enemy_race_#`.
- Položky `enemy_race_#` určují rasy jednotlivým hráčům, proti kterým má bot hrát. Hodnoty jsou stejné jako u položky `enemy_race`, s tím rozdílem, že ještě je možná hodnota `Default` s tím, že pak se výběr řídí právě podle položky `enemy_race`.

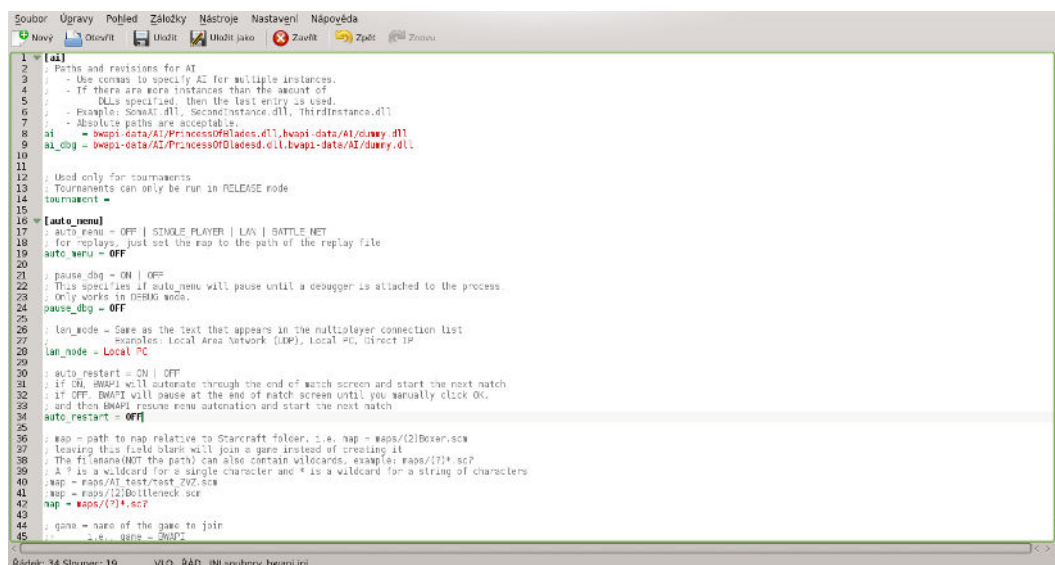
Pokud tedy je [BWAPI](#) nastaveno pro hraní proti vestavěné [AI](#) (`auto_menu` má hodnotu `SINGLE_PLAYER` a `race` má hodnotu `Zerg`), pak stačí spustit program `Chaoslauchner` (najdete ho v `<bwapi>/Chaoslauchner/-Chaoslauchner.exe`), zaškrtnou položku `BWAPI-Injector (1.16.1) RELEASE` a stisknout tlačítko `Start`. `Starcraft: Brood war` by se pak měl sám spustit a začít novou hru s nastavením ze souboru `<starcraft>/bwapi-data/-bwapi.ini`.

3.4 Hraní Princess Of Blades proti člověku na jednom počítači

Hraní botů proti člověku na jednom počítači není v [BWAPI](#) v základu přímo podporováno. Obvyklý postup by zahrnoval spuštění jedné instance `Starcraft: Brood war` na virtuálním stroji a hraní přes lokální síť virtuálního stroje. Nicméně pro potřeby testování a experimentů jsem vytvořil postup, který umožňuje hraní proti botům `Starcraft: Brood war` na jednom počítači, pomocí vestavěné funkčnosti [BWAPI](#).

[BWAPI](#) pro účely kompetitivního hraní botů přidává do základní nabídky multiplayeru `Starcraft: Brood war` položku `Local PC`, která umožňuje proti sobě hrát botům s takřka nulovou latencí na jednom počítači. Nicméně tato možnost je dostupná pouze, když je [BWAPI](#) injektovaná do `Starcraft: Brood war`, tedy pouze když je přítomný nějaký bot. Nasnadě bylo řešení použít běžného bota pro jednu instanci `Starcraft: Brood war` a pro druhou instanci (za kterou bude hrát člověk) použít hloupého bota, který nebude dělat nic. Respektive bude dělat jedinou věc a to, že povolí cheatovací flag `EnableUserInput`, aby jednotky ve hře mohli přijímat od hráče rozkazy. Následuje podrobný postup:

1. Nainstalujte `Princess Of Blades` podle pokynů v [3.2](#).
2. Spusťte `Multiple Instance Hack.bat` umístěný v `Starcraft/bwapi-data/`.
3. Upravte soubor `<starcraft>/bwapi-data/bwapi.ini`.
 - (a) V položce `ai` přidejte za hodnotu čárku a vložte text `bwapi-data/AI/dummy.dll`.
(výsledek:
`ai = bwapi-data/AI/PrincessOfBlades.dll,
bwapi-data/AI/dummy.dll`)
 - (b) Položku `auto_menu` nastavte na `OFF`.
(výsledek: `auto_menu = OFF`)
4. Z přiloženého DVD nakopírujte soubor `/bin/PrincessOfBlades/-dummy.dll` do složky `<starcraft>/bwapi-data/AI/`.

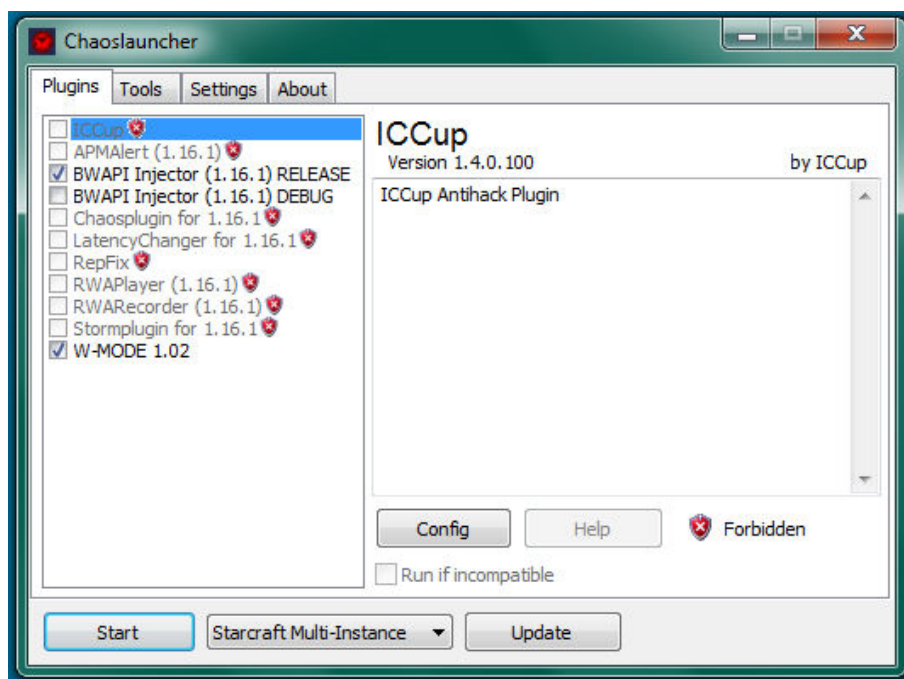


```
1 [ai]
2 ; Paths and revisions for AI
3 ; - Use commas to specify AI for multiple instances.
4 ; - If there are more instances than the amount of
5 ;   DLLs specified, then the last entry is used.
6 ; - Examples: Second.dll, SecondInstance.dll, ThirdInstance.dll
7 ; - Absolute paths are acceptable.
8 ;   ai = bwapi-data\AI\PrincessOfPalaces.dll,bwapi-data\AI\dummy.dll
9 ;   ai_obj = bwapi-data\AI\PrincessOfPalaces.dll,bwapi-data\AI\dummy.dll
10
11
12 ; Used only for tournaments
13 ; Tournaments can only be run in RELEASE mode
14 tournament =
15
16 [auto_menu]
17 ; auto_menu = OFF | SINGLE PLAYER | LAN | BATTLE NET
18 ; for replays, just set the map to the path of the replay file
19 auto_menu = OFF
20
21 ; pause_dbg = ON | OFF
22 ; This specifies if auto_menu will pause until a debugger is attached to the process.
23 ; Only works in DEBUG mode.
24 pause_dbg = OFF
25
26 ; lan_node = Same as the text that appears in the multiplayer connection list
27 ; examples: Local Area Network (LAN), Local PC, Direct IP
28
29 [lan_node]
30 lan_node = Local PC
31
32 ; auto_restart = ON | OFF
33 ; if ON, BWAPI will automate through the end of match screen and start the next match.
34 ; if OFF, BWAPI will pause at the end of match screen until you manually click OK,
35 ; and then BWAPI resumes new automation and start the next match
36 auto_restart = OFF
37
38 ; map = path to map relative to Starcraft folder, i.e. map = maps\[2]Boson.scm
39 ; leaving this field blank will join a game instead of creating it
40 ; The filename(NOT the path) can also contain wildcards, examples: maps\[??]*.sc?
41 ; A * is a wildcard for a single character and * is a wildcard for a string of characters
42 ; map = maps\AI\test\test_2002.scm
43 ; map = maps\[2]Bottleneck.scm
44 ; map = maps\[??]*.sc?
45
46 ; game = name of the game to join
47 ; i.e. game = BWAPI?
```

Obrázek 3: Soubor bwapi.ini upravený pro hraní proti člověku na jednom počítači

5. Spustě Chaoslauchner – MultiInstance umístěný v BWAPI/Chaoslauchner.
6. V Chaoslachneru zvolte ve spodním combo-boxu položku Starcraft – Multi-Instance.
7. Zaškrtněte v nabídce plug-inů položku BWAPI-Injector (1.16.1) RELEASE.
8. Zaškrtněte v nabídce plug-inů položku W-MODE 1.02.
9. Stiskněte tlačítko Start.
10. Objeví se okno Starcraft: Brood war, toto je instance ke bude hrát bot.
11. Pokud nechcete hrát v okně odškrtněte v nabídce plug-inů W-MODE 1.02⁶.
12. Stiskněte tlačítko Start.
13. Objeví se další okno Starcraft: Brood war. Zde budete hrát vy.
14. Ve vašem okně klikněte na Multiplayer→Expansion.
15. Z nabídky spojení vyberte Local PC (pokud se Local PC nezobrazuje znamená to, že není injektované BWAPI)-

⁶Nebudete-li chtít použít W-MODE, budete v následujících krocích muset mezi jednotlivými instancemi Starcraft: Brood war přepínat klávesovou zkratkou ALT+TAB



Obrázek 4: Chaoslauncher připravený ke spuštění více instancí Starcraft: Brood war

16. Vyberte (popř. vytvořte) profil a založte hru (Create Game), zvolte mapu a klikněte na Ok.
17. Přepněte se do okna bota.
18. Klikněte na Multiplayer→Expansion→Local PC→<jméno profilu>.
19. V nabídce byste měli vidět hru pod názvem profilu kterou jste založili.
20. Připojte se k ní a vyberte rasu Zerg (Princess Of Blades nebude za jinou rasu fungovat!).
21. Připněte se zpět do svého okna a klikněte na Ok, hra začne odpočítávat a spustí zápas.

3.5 Hraní Princess Of Blades proti jiným botům

Spuštění Princess Of Blades v zápasu proti jiným botům na jednom počítači je podobné jako hraní proti Princess Of Blades na jednom počítači. Pochopitelně přidávají se další problémy se spouštěním ostatních botů a navíc několik drobností, na které se ze pokusím upozornit.

Při spouštění her proti jiným botům můžete narazit na následující (a jiné) problémy:



Obrázek 5: Několik instancí Starcraft: Brood war běžících současně.

- Verze [BWAPI](#). Velká část botů dostupných na internetu je založena na starších verzích [BWAPI](#), zejména na [BWAPI 3.6](#). Tyhle verze jsou binárně nekompatibilní s [BWAPI 4.0](#) a nelze je spouštět z jednoho Chaoslachneru.
- Dávejte si pozor v jakém módu je bot kompilován (DEBUG, RELEASE), každý z nich má jinou knihovnu a injektuje se z Chaoslachneru odpovídajícím nastavením BWAPI-Injector (1.16.1) DEBUG resp. BWAPI-Injector (1.16.1) RELEASE-
- Chaoslachner - MultiInstance injektuje boty podle souboru `<starcraft>/bwapi-data/bwapi.ini`, podle nastavení `ai`. Tato položka může mít v hodnotě cestu k několika souborům oddělené čárkou (bez mezery!), přičemž do jednotlivých instancí Starcraft: Brood war se injektují postupně v pořadí v jakém jsou uvedeny.
- Pokud je v `<starcraft>/bwapi-data/bwapi.ini` uvedeno méně cest k botům, než je vytvořeno instancí, všechny takto nadbytečné instance budou injektovány posledním botem v pořadí

3.6 Nastavení Princess Of Blades

Nastavení Princess Of Blades jsou minimální, týkají se zejména Bayesovského prediktoru (viz. [4.1](#)) a všechny se nachází v konfiguračním souboru `<starcraft>/bwapi-data/bwapi.ini`. Nastavení jsou následující:

- Sekce `[options]`
 - Položka `bayesPredictor`, možné hodnoty jsou `true`, `false`. Tato položka ovlivňuje použití Bayesovského prediktoru při hraní bota Princess Of Blades, jestliže je nastavená na pravdu⁷ bude Bayesovský prediktor zapnutý, jinak bude vypnutý.
 - Položka `bayesComputationTime`, možné hodnoty jsou libovolná celá čísla větší než 0. Toto je položka ovlivňující kolik času má Bayesovský prediktor v každém snímku na výpočet. Na testovacím stroji⁸ se hodnota použitelná pro plynulé hraní, pohybovala mezi 7 a 15 milisekundami. Pokud máte s Princess Of Blades se zapnutou Bayesovskou predikcí problémy s výkonem, snižte tuto hodnotu (0 nebo záporné hodnoty způsobí, že se Princess Of Blades bude chovat jako by byl Bayesovský prediktor vypnutý). Pokud je Bayesovský prediktor vypnutý, nemá položka `bayesComputationTime` žádný vliv.

⁷Princess Of Blades ve svém nastavení používá zobecněnou pravdivost. Jakýkoli řetězec, který není řetězcem `false`, je považován za pravdu

⁸Intel(R) Core-i3 M330 2.13GHz, 4GB DDR3 RAM, Mobility Radeon HD 4500 Series

4 Implementované algoritmy

V této kapitole popíšu algoritmy implementované v Princess Of Blades. Nebudu zde popisovat jednoduché skripty použité pro naivní implementaci jednotlivých agentů (pro jejich specifikaci viz. kapitola 5), ale zaměřím se na algoritmy s přidanou hodnotou.

4.1 Bayesovská síť

V rámci snahy adaptivně reagovat na protihráčovu strategii a její změny vyvstává problém automatického určení protihráčovy strategie. Tento problém lze chápat jako problém klasifikace s tím, že v prostředí RTS je kvůli herní mechanice fog of war⁹ velké množství neúplných informací. Jako ukázkový algoritmus pro demonstraci modularity a snadné nahraditelnosti jednotlivých agentů v Princess Of Blades, jsem proto implementoval Bayesovskou síť pro určení build orderu protivníka (inspirovanou prací [9]) a reaktivní změnu strategie na základě klasifikovaného build orderu.

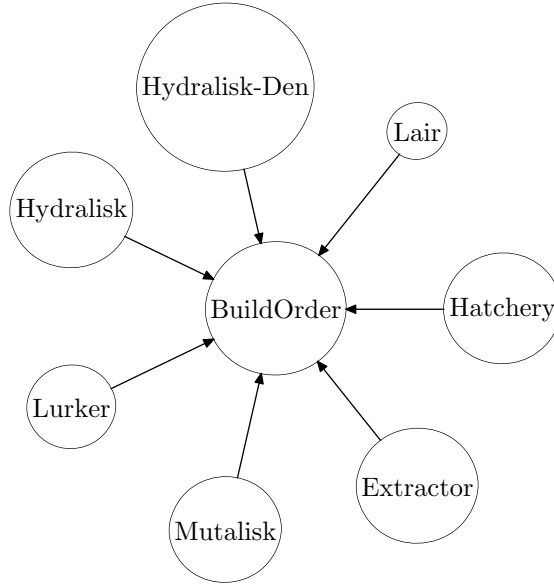
Bayesovská síť je grafický model pro vyjádření vzájemných vztahů náhodných proměnných. Bayesovská síť k tomu používá orientované grafy, kde uzly grafu reprezentují náhodné proměnné a orientované hrany potom jejich vztahy (závislost náhodných proměnných). Každá náhodná proměnná (uzel grafu) má své rozdělení pravděpodobnosti a závislé proměnné se navzájem ovlivňují podle Bayesova teorému. Jako klasifikátor se Bayesovská síť používá tak, že proměnné (uzly) v síti jsou proměnné¹⁰ klasifikovaného vektoru a další proměnnou (uzlem) je třída. Poté je nutné (obvykle pomocí expertů, ale existují i algoritmická řešení) určit strukturu sítě, tedy orientované hrany. Kvalifikace pak probíhá tak, že známé proměnné (pokud kvalifikujeme úplný vektor tak triviálně všechny proměnné kromě třídy) označíme jako důkazní a přiřadíme jim hodnotu. Následuje výpočet algoritmu, který přepočítá podmíněné pravděpodobnosti v ne-důkazních proměnných a za přiřazenou třídu považujeme tu, která má největší pravděpodobnost. Algoritmů pro výpočet Bayesovské sítě existuje několik, ať už přesných, které mají často NP-těžké instance, obvykle libovolné sítě obsahující neorientovaný kruh, nebo aproximačních.

Bylo nutné zvolit dva algoritmy pro implementaci v Princess Of Blades a to algoritmus pro učení sítě (učí se pouze pravděpodobnosti, struktura sítě je pevná viz. obrázky 7, 8 a 6) a algoritmus pro výpočet. Pro učení jsem zvolil offline učící algoritmus MLE, což je ve skutečnosti obecnější statistická metoda pro určení a odhadování parametrů ve statistických modelech. Data použitá pro učení jsou potom získána z replayů ze stránek klanu teamliquid¹¹ (pro podrobnější

⁹Fog of war zakrývá ty části mapy, kam přímo nevidí hráčovy jednotky. Hráč tak sice má informace o terénu a naposledy spatřených statických objektech (budovy, zdroje), ale nevidí pohyb a složení nepřátelských jednotek, dokud se nedotnou na dohled hráčových jednotek.

¹⁰Předpokládáme, že známe rozdělení pravděpodobnosti jednotlivých proměnných. Pokud ne, lze použít některý z učících algoritmů pro získání těchto rozdělení.

¹¹<http://wiki.teamliquid.net/starcraft/>



Obrázek 6: Bayesovská síť pro ZwZ matchup

informace viz. 4.1.3). Jako algoritmus výpočtu jsem použil Gibbsův Sampler což je [Markov Chain Monte Carlo \(MCMC\)](#) algoritmus, který lze použít na výpočet Bayesovské sítě. Použil jsem ho zejména proto, že má dobrou implementaci a podporu v knihovně DLib, kterou jsem k reprezentaci Bayesovských sítí používal.

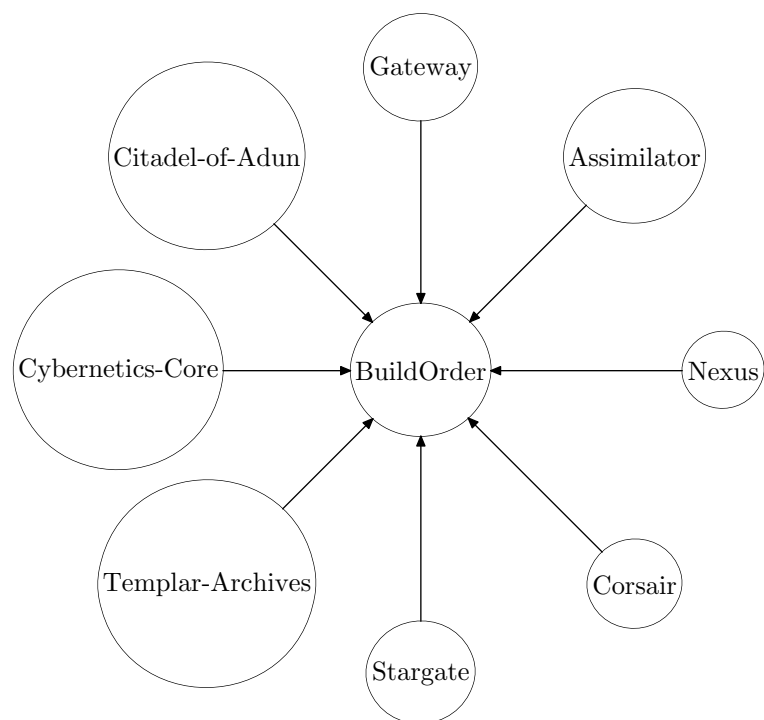
4.1.1 MLE

Problém určení volných parametrů Bayesovské sítě (se známou strukturou) je stanoven následovně. Necht bn je Bayesovská síť s n proměnnými $X_1, X_2, \dots, X_n$. Necht $pa(X)$ je množina rodičů proměnné (uzlu) X . Necht $D = \{D_1, D_2, \dots, D_m\}$ kde $D_i = \{d_{i1}, d_{i2}, \dots, d_{in}\}$ je dataset s m nezávislými řádky s identickou distribucí - [Independent and Identically Distributed \(i.i.d.\)](#). Problém nalezení volných parametrů Bayesovské sítě je problém nalezení pravděpodobností $P(X_i|pa(X_i))$ pro všechna $i \in \{1, \dots, n\}$.

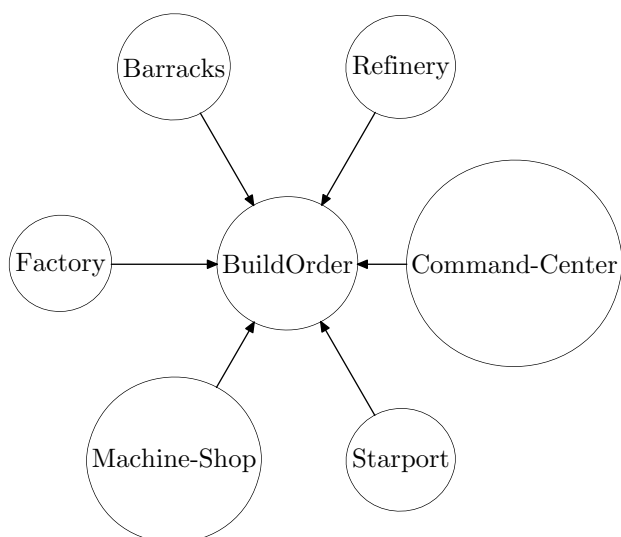
Předpokládáme, že v datasetu D nejsou žádná chybějící data. Pak podmíněné pravděpodobnosti podle [MLE](#) lze spočítat tak, že pro danou proměnnou X_i , pro danou hodnotu proměnné v a pro dané hodnoty rodičů w se spočítá počet řádků v datech D kde $X_i = v$ a zároveň $pa(X_i) = w$ a podělí se počtem řádků v datech D kde $pa(X_i) = w$. Viz pseudokód 1.

4.1.2 Gibbsův sampler

Gibbsův sampler je algoritmus pro generování posloupnosti vzorků z rozdělení pravděpodobnosti, za účelem aproximace tohoto rozdělení. Dá se použít, když samotnou distribuci neznáme, ale známe podmíněné pravděpodobnosti každé proměnné. Gibbsovo vzorkování se používá ke generování jednoho vzorku z rozdělení



Obrázek 7: Bayesovská síť pro ZvP matchup



Obrázek 8: Bayesovská síť pro ZvT matchup

Zdrojový kód 1: [MLE](#) v Princess Of Blades

```

input: Bayes Network  $bn$ ,
      Dataset  $D = \{D_1 = \{d_{11}, d_{12}, \dots, d_{1n}\}, \dots, D_m\}$ 
output:  $P(X|pa(X))$  for each  $X \in bn$ 
begin
  for all node  $X_i$  of  $bn$ 
    for all parent state  $pa(X_i)$  of node
      for all value  $v$  of  $X_i$ 
         $P(X_{node}|pa(X_{node})) = \frac{|\{D_j|D_{ji}=v \wedge pa(D_{ji})=pa(X_i)\}|}{|\{D_j|pa(D_{ji})=pa(X_i)\}|}$ 
      end
    end
  end
end

```

každé proměnné v jednom kole, na základě hodnot podmiňujících proměnných. Lze ukázat, že posloupnost vzorků obsahuje Markovův řetěz a že stacionární rozdělení tohoto řetězu je právě hledané rozdělení pravděpodobnosti.

Nechť $p(X_1, \dots, X_n | e_1, \dots, e_m)$ je rozdělení pravděpodobnosti náhodných proměnných $(X_1, \dots, X_n)$ podmíněnou množinou důkazních proměnných $(e_1, \dots, e_m)$. Sampler vybírá posloupnost vzorků $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}, \dots$ z Markovova řetězce nad všemi možnými stavy proměnných. Stacionární rozdělení markovova řetězce je rozdělení pravděpodobnosti $P(X_1, \dots, X_n | \mathbf{e})$. Tedy pokud bude řetězec dostatečně dlouhý, aby dosáhl stacionárního rozdělení, pak bude vracet náhodné vzorky z rozdělení $P(X_1, \dots, X_n | \mathbf{e})$. Příklad fungování můžete vidět v algoritmu [2](#).

Zdrojový kód 2: Gibbsův sampler

Vstup: Množina důkazních proměnných $\mathbf{e} = (e_1, \dots, e_m)$, podmíněné rozdělení pravděpodobnosti $P(X_i | (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n), \mathbf{e})$

Výstup: Markovův řetězec $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots$

1. Inicializace
 - (a) Přiřaď $X_i \leftarrow$ jednu z hodnot x_i , $1 \leq i \leq n$.
 - (b) Nechť $\mathbf{x}^{(0)} \leftarrow (x_1, \dots, x_n)$
2. Pro $t = 1, 2, \dots$
 - (a) Vyberte náhodně s uniformním rozdělením index i , $1 \leq i \leq n$.
 - (b) Vygeneruj náhodný vzorek $x_i^{(t)}$ z $P(X_i | (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)^{(t-1)}, \mathbf{e})$.
 - (c) Nechť $\mathbf{x}^{(t)} = (x_1^{(t-1)}, \dots, x_{i-1}^{(t-1)}, x_i^{(t)}, x_{i+1}^{(t-1)}, \dots, x_n^{(t-1)})$

4.1.3 Technická řešení

Sběr dat pro učení sítě probíhá pomocí automatického přehrávání replayů¹² v [BWAPI](#). Bohužel současná verze [BWAPI](#) (4.0) obsahuje chybu, která toto přehrávání znemožňuje. Bot, který data schraňuje je tedy založen na starší verzi [BWAPI](#) 3.7, je to jeden z vedlejších projektů v rámci Princess Of Blades a nese název DataGathering. Z pohledu [BWAPI](#) se jedná o běžného bota, který dělá pouze to, že pomocí objektu reprezentace znalostí o protivníkovi (viz. 5) zaznamenává stav jednotlivých hráčů v čase. Čas je měřen v rozlišení 5 sekund. Po přehrávání každého replaye se uloží získaná data do souboru a volitelně i build orderly jednotlivých hráčů. Umístění souborů s daty, volba sledované rasy a přepínání ukládání build orderů se nalézá v konfiguračním souboru `DataGathering.ini`.

Soubory týkající se bayesovského prediktoru v Princess Of Blades jsou uspořádány následovně. Ve složce se soubory bota¹³ se nachází složka `BayesData` a v ní jsou podsložky `PvZ`, `TvZ` a `ZvZ`, odpovídající jednotlivým matchupům pro zergy. V každé složce jsou následující soubory:

- `net.bn` - soubor bayesovské sítě
- `BOS.txt` - soubor obsahující jména rozpoznávaných strategií a jména odpovídajících protistrategií

Soubor `BOS.txt` je potom ve formátu, kdy na každém řádku je jedna rozeznávaná strategie a za ní oddělené mezerou jsou známe protistrategie. Tedy `<strategie> <protistrategie_1> <protistrategie_2>...`

Princess Of Blades se zapnutou bayesovskou predikcí načítá jednotlivé sítě ze souborů. Soubory jsou v proprietárním binárním formátu založeném na serializované bayesovské síti z knihovny `DLib`. Tento formát je nicméně upraven, tak aby mohl pojmut pojmenované uzly sítě. Samotné sítě lze potom libovolně nahrazovat za předpokladu dodržení následujících konvencí:

- Síť musí obsahovat uzel nazvaný `BuildOrder`.
- Indexy jednotlivých build orderů odpovídají indexům řádků v souboru `BOS.txt` ve stejném adresáři jako síť.
- Každý build order má alespoň jednu protistrategii v souboru `BOS.txt` ve stejném adresáři jako síť.
- Uzly odpovídající budovám mají alespoň dvě hodnoty (v tom případě - nepozorován a pozorován) s tím, že větší počet hodnot určuje počet budov a maximální hodnota označuje stejný nebo vyšší počet budov (např. nepozorován, pozorován 1 exemplář, pozorovány 2 exempláře, pozorováno 3 a více exemplářů).

¹²Replay je zaznamenaný zápas ve Starcraft: Brood war v proprietárním binárním formátu. Struktura souborů replayů (*.rep) není známá a přehrávání v enginu hry je jediný známý způsob jak z nich získat data

¹³`<Starcraft dir>/bwapi-data/AI/PrincessOfBlades/`

- Uzly odpovídající jednotkám (které nejsou budovy) mají právě dva stavy (nepozorováno, pozorováno).
- Názvy uzlů odpovídající jednotkám (včetně budov) odpovídají názvům tak jak se používají v [BWAPI 4.0](#) (metoda `UnitType::getName()`).
- Název uzlu odpovídajícímu času v rozlišení pěti sekund je `time`.

Sítě je možné vytvářet pomocí doprovodného programu

`BayesNetworkCreate.exe`. Jedná se o dávkový soubor určený pro spuštění v příkazové řádce. Jako první argument přijímá cestu k souboru s definicí sítě, druhý argument je cesta k souboru s daty¹⁴ a třetí argument je jméno souboru s výslednou sítí.

Soubor s definicí sítě je textový soubor skládající se ze dvou částí. První část obsahuje na samostatném řádku vždy jméno uzlu sítě a počet jeho stavů oddělený mezerou: tedy `<jmeno_uzlu> <pocet_stavu>`. Druhá část souboru oddělená prázdným řádkem obsahuje orientované hrany sítě ve formátu `<pocatecni_uzel> <koncovy_uzel>`. Na jednom řádku je právě jedna hrana.

Soubor s daty pro učení sítě je textový soubor, kde na prvním řádku je hlavička souboru (jména proměnných) a na každém dalším řádku je vektor hodnot proměnných. Oddělovačem hodnot i názvů je vždy mezera.

Při výpočtu sítě se za důkazní proměnné považují nepřátelské budovy a jejich počet, zpozorované nepřátelské jednotky a čas. Hodnoty všech těchto proměnných poskytuje modul sdílených informací o protivníkovi (viz. kapitola 5.6). Hodnoty proměnných jsou nastaveny vždy na začátku výpočtu. Počet vzorků, potřebných pro to aby byl výsledek považován za reprezentativní, je udán konstantou (2000 vzorků), přičemž délka výpočtu se může lišit na základě nastavení (viz. kapitola 3.6). Výsledek výpočtu je potom nejpravděpodobnější hodnota proměnné `BuildOrder`.

Protože knihovna [BWAPI](#) není thread-safe, není možné běžným způsobem použít vlákna při výpočtu bayesovské sítě. Nicméně vzhledem k použitému aproximačnímu algoritmu (Gibbsovo vzorkování) je možné poměrně snadno výpočet rozdělit na několik samostatných celků, kde se vždy za nějaké pevně dané časové kvantum provede co největší počet vzorkování a ukládá si mezivýsledky. V *Princess Of Blades* je to realizováno tak, že při výpočtu bota (metoda `onFrame()`) je část času rezervována pro výpočet sítě a zbytek pak běžné režii bota. Jakmile síť provede dostatečně velký počet vzorkování, všechny mezivýsledky se agregují a určí se pravděpodobnosti proměnné `BuildOrder`. Hodnota s největší pravděpodobností je potom považována za protivníkovu strategii a v tabulce jsou nalezeny vhodné protistrategie. Jestliže bot už některou z těchto strategií provádí neděje se nic, ale pokud provádí jinou proběhne, dynamická náhrada nové strategie (náhodně vybrané z možných) za starou (viz. 5.1).

¹⁴V datech musí být všechny proměnné použité v definici sítě!

5 Implementační detaily

V této kapitole se budu zabývat různými komplexnějšími implementačními detaily Princess Of Blades. Nejdříve popíšu použitou reprezentaci build orderů a související témata, dále pak technický princip komunikace a rozkazů v hierarchii bota, dále specifikuji použitý naivní ekonomický a taktický skript, techniky použité pro naivní mikromanagement a způsob uchovávání znalostí o protivníkovi.

5.1 Reprezentace build orderů a související nástroje

Pro účely snadného přidávání nových strategií a jejich vzájemného zaměňování pro Bayesovský prediktor, bylo nutné nějakým obecným způsobem reprezentovat build ordery. V úplně nejjednodušším případě si lze build order představit jako prostou frontu úkonů určených pro Ekonomického agenta Princess Of Blades. Nicméně tento přístup se velice brzy ukazuje jako nedostačující a to z následujících důvodů: Build ordery, tak jak je obvykle chápou hráči, jsou pouze jakousi základní šablonou, která se v průběhu hry mění a ohýbá podle situace (a to často i dost zásadně) a neexistuje nic jako jediný build order, který by bylo možné dodržet v rámci celého zápasu¹⁵. Navíc pokud bychom se skutečně uchýlili k jediné frontě úkonů, hrozí nám, že fronta se někde v průběhu zápasu vyprázdní a bot už pak prostě nebude nic dělat.

Zvolil jsem o něco sofistikovanější reprezentaci build orderů a okolo ní potom vytvořil aparát pro zacházení s nimi (parseř pro ukládání a načítání do souborů, aplikaci pro snadné vytváření a editaci, funkce pro dynamickou výměnu build orderu za běhu bota v zápase): Build order (třída `BuildOrder`) obsahuje tři kolekce a mapu proměnných. Kolekce odpovídají následujícím fázím: Přípravná fáze kdy se buduje infrastruktura a připravují se podmínky pro budoucí výrobu (obvykle zahrnuje výrobu dělníků a postup v technologickém stromu). Cyklická fáze kdy se vyrábí jednotky určené pro boj a dlouhodobá fáze kde se nachází dlouhodobé podpůrné cíle (obvykle provedení upgradů, další expanze a posilování ekonomiky). Mapa proměnných obsahuje záznamy ve tvaru `JMÉNO_PROMĚNNÉ = HODNOTA` a využívá se hlavně pro uchovávání meta-informací o build orderu. Například jeho název, strana pro a proti které je určen. Nicméně bylo by možné takto například nastavovat i nějaké proměnné přímo v botovi (v současnosti nepodporováno).

Provedení build orderu je dvojího druhu. První případ je na začátku zápasu, kdy si Princess Of Blades náhodně vybere build order z množiny možných. Pak se provede celá přípravná fáze a jakmile je dokončena, nastane čas pro cyklickou fázi a dlouhodobé cíle. Ty se potom střídají tím způsobem, že se vždy provede celá cyklická fáze a následuje jeden úkon z dlouhodobé fáze. Tak to probíhá dokud se celá dlouhodobá fáze nevyčerpá, pak následují už jen cyklické fáze. Druhým

¹⁵Opomím tady samozřejmě build ordery a strategie obecně, které jsou zaměřeny na rychlé vítězství nějakou překvapivou taktikou tzv. Cheese. Ty jsou vždy specifické a lze je obecně považovat za zvláštní případ běžných build orderů

případem provedení build orderu, je dynamická změna v rozehraném zápasu. Nejdříve se vyprázdní fronta úkonů v Ekonomickém agentovi (úkony které už jsou delegovány na podřízené agenty a mají vyhrazené suroviny jsou ponechány). Potom je nahrána přípravná fáze s tím rozdílem, že je jsou vynechány následující položky:

- Dělníci a to až do počtu, kolik jich hráč aktuálně vlastní, ostatní dělníci jsou ponecháni.
- Jednotky poskytující supply a to až do počtu, kolik jich hráč aktuálně vlastní, ostatní jsou ponecháni.
- Budovy a to až do počtu, kolik jich hráč aktuálně vlastní, ostatní jsou ponechány.
- Technologie, které hráč již vyzkoumal.
- Upgrady, které hráč již provedl.

Všechny ostatní úkony jsou ponechány. Cyklická a dlouhodobá fáze probíhají normálně.

Pro snadné vytváření build orderu ve výše popsaném formátu jsem vytvořil doplňkovou aplikaci BOBuilder. Jedná se o jednoduchou GUI aplikaci pro snadné prohlížení, vytváření a editaci build orderů, všech jejich fází a nastavování proměnných. Obrázek z aplikace můžete vidět na 9.

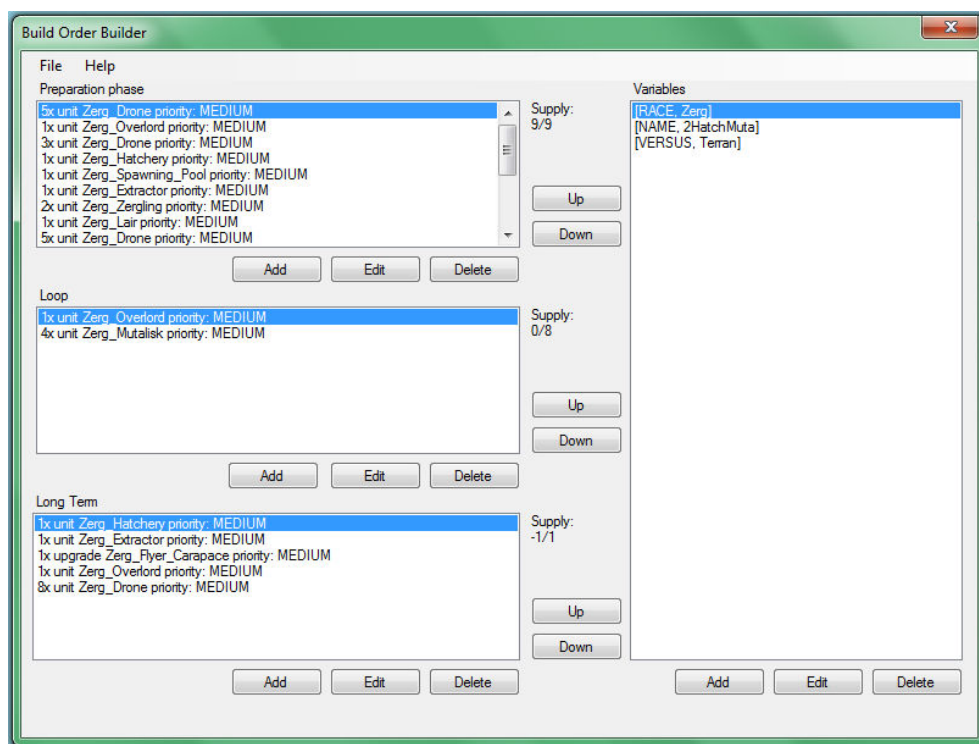
5.2 Hierarchy velení a zasílání zpráv

Jak jsem již popsal v kapitole 2 struktura Princess Of Blades je přísně hierarchická. Agenti výše v hierarchii vydávají rozkazy podřízeným agentům a naopak podřízení agenti zasílají zprávy se svými požadavky svým nadřízeným. Vydávání rozkazů směrem dolů je řešeno pomocí volání veřejných metod podřízených agentů. Naopak zasílání zpráv směrem nahoru se provádí jako v návrhovém vzoru řetěz odpovědnosti (chain of responsibility), kde další uzel na cestě je vždy nařízený agent.

Každá takto zasláná zpráva obsahuje odkaz na agenta, který zprávu původně vytvořil a taky odkaz na posledního agenta, který ji odeslal výše v hierarchii. Pro další účely může zpráva obsahovat i další informace.

Když agent přijme takovou zprávu, nejdříve zjistí, jestli má pro daný typ zprávy definovanou obsluhu. Pokud ano provede kód uložený v obsluze zprávy. Jestliže pro danou zprávu obsluhu nemá, pak mohou nastat dvě možnosti: Jestliže existuje výše postavený agent v hierarchii, pak mu zprávu předá, jinak (případ agenta Strategy Manager) zprávu zahazuje.

Celý aparát zasílání a předávání zpráv použitý v Princess Of Blades je implementován v malé knihovně chainOfResponsibility, kterou jsem pro tento účel vytvořil. Každá ze tříd jednotlivých agentů je potom potomkem třídy `chainOfResponsibility::Client`. Pro podrobnosti viz. dokumentace `chainOfResponsibility` a zdrojové kódy.



Obrázek 9: Aplikace BOBuilder

5.3 Ekonomický hard-skript

Naivní implementace ekonomické části hraní Starcraft: Brood war v Princess Of Blades je poměrně sofistikovaná a za předpokladu, že bude dostávat adekvátní příkazy od nadřízených agentů, by měla být schopna bez větších problémů podporovat libovolně sofistikované strategie bojové části bota a velení. Naivní implementace je schopna samostatně těžit suroviny (a balancovat jejich těžbu), na příkaz stavět budovy a vyrábět jednotky (i opožděně až ve chvíli kdy jsou k dispozici prostředky, pak pracuje jako FIFO s prioritami), zkoumat technologie a provádět upgrady. Rovněž má implementovány predikáty pro testování schopnosti vyrábět daný typ jednotky (ať už z důvodů nedostatku surovin, nedostatečného postupu v technologickém stromě nebo obojího). Následuje stručný popis.

Agent implementovaný v Princess Of Blades jako `NaiveEconomyManager` se skládá ze dvou hlavních částí. Těmi jsou výrobní fronta a alokace zdrojů.

Výrobní fronta je implementačně prioritní fronta obsahující úkony výroby nebo výzkumu zadané nařízeným agentem. Úkon (třída `CreateTask`) je kontainer obsahující typ jednotky (technologie, upgradu), timestamp, prioritu a volitelně i pozici. Možných hodnot priority je 10 a jsou to tyto `SMALLEST`, `SMALL_MEDIUM`, `SMALL_HIGH`, `MEDIUM_SMALL`, `MEDIUM`, `MEDIUM_HIGH`, `HIGH_LOW`, `HIGH_MEDIUM`, `HIGHEST`, `IMEDIATELY`, při-

čemž defaultní hodnotou je MEDIUM. Úkony se ve frontě řadí primárně podle priority a poté podle timestampu.

Alokace zdrojů souvisí s podřízenými agenty. Ve Starcraft: Brood war může nastat situace, kdy sice má hráč suroviny a technologie potřebné k výrobě jednotky nebo stavbě budovy, ale stavba nemůže začít okamžitě (typicky protože stavitel musí nejdříve dorazit na místo, kde má budova stát nebo hráč právě nemá k dispozici žádnou larvu k mutaci jednotky). Pak je nutné i po předání úkonu podřízenému agentovi rezervovat zdroje potřebné k danému úkonu, jinak by se mohlo stát, že mu je sebere další úkon ve frontě (protože předaný úkon ještě z výše vedených důvodů nezapočal a hra tedy zdroje hráči neodečetla). Alokace zdrojů je vnitřně implementována v podřízených agentech (abstraktní třída `AbstractManagers::SubEconomyManager`) a `NaiveEconomyManager` tyto informace spravuje a využívá je v predikátech možnosti výroby.

Agent implementovaný v Princess Of Blades jako `NaiveGatheringManager` ovládá těžbu minerálů a plynu. Má k dispozici kolekci dělníků a musí tuto těžbu vyvažovat. Poměrně snadno se může stát (pokud protivník zvolí agresivní strategii jako například Zealot Rush nebo Mutalisk Harras), že velká část hráčových dělníků bude zničena (některé strategie Starcraft: Brood war se přímo specializují na tzv. economy damage). V tom případě by mohlo u zcela naivní implementace dojít k tomu, že budou zničeni všichni dělníci těžící jednu surovinu a přestože hráč stále má dělníky, prohraje, protože mu surovina dojde. Agent v Princess Of Blades toto řeší tak, že vždy udržuje na těžbě plynu buďto polovinu dělníků nebo 3 na každém obsazeném zdroji plynu (počet, kdy je zdroj maximálně využit), podle toho co vyjde na méně dělníků. Zbytek dělníků pak těží minerály.

Výroba jednotek implementovaná v agentovi `NaiveUnitBuildManager` je specifická pro rasu zergů (pro kterou je Princess Of Blades primárně určena). Důvodem je specifický způsob jakým zergové vyrábí jednotky. Zatímco rasy terran a protoss vyrábí své jednotky v různých budovách (až na omezený počet výjimek např. protosský Archon a Dark Archon), zergové mají jediný typ budovy (Hatchery), která sama periodicky generuje jednotky typu larva, s tím, že každá hatchery může mít nejvýše tři larvy zároveň. Larvy jsou asociované s mateřskou hatchery, extrémně odolné a nelze je nijak ovládat s výjimkou mutace. Mutace potom promění larvu na žádanou jednotku. Výjimku opět tvoří tři jednotky (lurker, guardian a devourer), které se nemutují z larev, ale z jiných jednotek (hydralisk pro lurkera a mutalik pro guardianu a devourera). `NaiveUnitBuildManager` toto řeší tak, že si udržuje množinu larev a množinu hydralisků a mutalisků, které má k dispozici. Ty potom mutuje přímo, pokud nemá danou speciální jednotku pro druhotnou mutaci, zažádá o ni nadřízeného agenta.

Stavba základny (umísťování budov) je s jedinou výjimkou ponechána na vnitřních algoritmech Starcraft: Brood war, které umožňuje BWAPI využívat. Jedinou výjimku tvoří stavba tzv expandů. Expand je nová hlavní budova umístěná na mapě tak, aby umožnila rychlou těžbu surovin ze vzdálených nalezišť. V Princess Of Blades v `NaiveStructureManager` je umístění expandů řešeno

následovně. [BWAPI](#) poskytuje každému botovi statické informace o mapě¹⁶, z těch `NaiveStructureManager` zjistí umístění gejzírů plynu (vespene geyser), z těch vyfiltruje ty již hráčem obsazené a seřadí je podle vzdálenosti. Umístěním pro expand je potom nejbližší neobsazený gejzír (o finálním umístění budovy rozhoduje opět vnitřní algoritmus Starcraft: Brood war).

5.4 Taktický hard-skript

Taktický hard-skript pro řízení vyšší úrovně boje funguje v *Princess Of Blades* pouze na velmi jednoduchém principu. Nicméně pro úplnost jsem se rozhodl jeho defaultní chování rozebrat v této kapitole.

Taktický hard-skript sestává ze čtyř samostatných naivních implementací agentů. Tyto implementace jsou samostatné a za předpokladu dodržení rozhraní by neměl být problém je jednotlivě nahrazovat sofistikovanějšími implementacemi. Naivní implementaci tvoří tyto čtyři agenti `NaiveTacticalManager`, `NaiveAttackManager`, `NaiveDefenseManager` a `NaiveScoutingManager`.

Agent `NaiveTacticalManager` zastřešuje celý taktický hard-skript. Jeho chování je jednoduché, při výběru cíle útoku se vždy dotáže na sdílené informace o protivníkovi (viz. 5.6) a zvolí za cíl nejslabší známou základnu protivníka. Všechny přidělené jednotky zařazuje automaticky do obrany s výjimkou definovaných průzkumných jednotek (*Overlord*). Pokud je `NaiveTacticalManager` požádán některým z podřízených agentů o jednotky, postupuje následovně: Jestliže žádá podřízený agent obrany, pak žádané jednotky odebere ostatním pořízeným agentům a přidělí je okamžitě. Jinak zkontroluje jestli není obraný agent zaneprázdněn a pokud není, odebere mu jednotky a přidělí je. Jinak žádost ignoruje.

Oba bojovní agenti tedy `NaiveAttackManager` i `NaiveDefenseManager` pracují tak, že své přidělené jednotky pošlou na místo určení (obranu nebo útoku), s tím, že jestliže nemají dostatek jednotek, zažádají o další. `NaiveDefenseManager` se potom snaží na daném místě držet pozice, kdežto `NaiveAttackManager` agresivně útočí (viz. 5.5)

Posledním agentem je agent průzkumu `NaiveScoutingManager`. Ten na rozdíl od bojových agentů nepoužívá základní bojovou implementaci mikromanagementu, ale používá specializovanou implementaci pro průzkum (viz. 5.5). Dále vybírá lokace k průzkumu, ty se generují náhodně z následujících: startpointy protivníků, statické lokace gejzírů plynu (místa pro expand), náhodné lokace v okolí středu mapy. Rovněž je nutné zmínit, že `NaiveScoutingManager` automaticky přiděluje každou svou jednotku do nového svazu, aby prozkoumal co největší množství vygenerovaných bodů.

¹⁶Mezi statické informace se řadí počáteční umístění hráčských základen (tvz. startpointy) a počáteční umístění neutrálních jednotek

5.5 Naivní micromanagement

Mikromanagement je součástí hraní RTS, kde hráč kontroluje jednotlivé jednotky (obvykle v bojové situaci) s účelem způsobit co největší újmu protivníkovi při zachování co největšího počtu vlastních jednotek. Pro mikromanagement existuje několik zajímavých algoritmů jako například [10] nebo [11]. Pokud vyvíjíme algoritmus pro plánování nebo volbu build orderů, pak se může hodit jednoduchý skript založený na jednoduchém principu „zaměř nejzraněnější jednotku“. Princess Of Blades poskytuje dva základní naivní mikromanagery.

Hlavním agentem pro mikromanagement je agent `NaiveMicroManager`. Je to poměrně jednoduchý skript, založený na zaměřování nejzraněnější nepřátelské jednotky v dosahu zbraní. Agent si udržuje tabulku jednotek, které ovládá s asociovaným cílem. Cílem může být buďto jednotka nebo místo. Jestliže je cílem jednotka, pak se agent snaží svou jednotku dostat na dosah zbraní a zaútočit. Jestliže je cílem místo, pak se snaží dostat svou jednotku do určité vzdálenosti od toho místa. Agent operuje ve dvou módech, v agresivním módu (`ENGAGE`), kde automaticky napadá a pronásleduje nepřítele v dohledu a v módu držet pozice (`HOLD`), kdy při přesunu nepřítele ignoruje a na místě pronásleduje nepřítele jen do určité vzdálenosti od cílové oblasti.

Druhým mikromanagement agentem je specializovaný agent pro průzkum. Průzkum je z hlediska mikromanagementu specifický, protože je účelem vidět co největší část mapy (a nepřátelských jednotek) a přitom se vyvarovat konfliktu (škody v tomto případě nehrají žádnou roli). Agent se jmenuje `NaiveScoutingMicroManager` a funguje podobně jako `NaiveMicroManager` s tím rozdílem, že nevyhledává jednotky pro útok a při napadení ustupuje směrem od nepřátelských jednotek.

5.6 Reprezentace znalostí o protivníkovi

Uchovávání znalostí o protivníkovi je důležitou součástí plánování strategie a taktiky. Protože ve Starcraft: Brood war funguje princip tzv. fog of war, je nutné nějakým způsobem ukládat informace o tom, jaké nepřátelské jednotky byly spatřeny a umístění nepřátelských budov objevených průzkumem. Pro tyto účely implementuje Princess Of Blades sdílené informace o protivníkovi ve třídě `EnemyIntel`.

Třída `EnemyIntel` automaticky ukládá spatřené typy nepřátelských jednotek, pokud se jedná o budovy ukládá i jejich umístění a počet. Budovy potom vnitřně rozděluje do jednotlivých známých nepřátelských základen (za základnu je považován sluk budov v určité vzdálenosti od sebe). Třída umožňuje dotazování se na nejslabší nepřátelskou základnu a přiřazení budovy do základny a naopak.

Sdílené informace o protivníkovi jsou napojené v abstraktní třídě nejvyššího agenta bota (`AbstractStrategyManager`) a skrz konstruktory se automaticky distribuují do podřízených agentů, kde jsou dostupné jako slot `Intelligence`.

Matchup	počet her	počet výher	proc. výher	počet proher	proc. proher
ZvT	19	0	0%	19	100%
ZvP	16	1	6.25%	15	93.75%
ZvZ	17	2	3.4%	15	96.6%

Tabulka 2: Výkon Princess Of Blades proti vestavěné [AI](#), bez požití Bayesovské sítě

6 Experimenty a výkon bota

V této kapitole se věnuji několika experimentům, kterým jsem základní implementaci Princess Of Blades podrobil. Vzhledem k zaměření bota se jedná spíše o testy menších rozměrů a očekávaný výkon není nijak velký. Testuje se jak verze se zapnutou, tak s vypnutou bayesovskou sítí pro predikci nepřátelského build orderu. V pozorováních uvedu celkovou statistiku i příklady některých zajímavějších zápasů.

6.1 Výkon proti vestavěné [AI](#)

[AI](#) vestavěná ve Starcraft: Brood war není nijak zvláště sofistikovaná. Při hraní lze v zásadě odpozorovat dvě předem připravené strategie za každou hratelnou stranu. Zdá se, že [AI](#) preferuje ve své taktice zejména poměrně rychlý a ničivý počáteční útok (podpořený podváděním se surovinami a informacemi o mapě). Jestliže je tento útok odražen, vestavěné [AI](#) poměrně dlouho trvá, než se připraví a provede další. Obvykle dokonce čeká na pokročilé jednotky pozdní části hry.

Co se výkonu Princess Of Blades proti vestavěné [AI](#) týče, ukazuje se, že extrémně záleží na volbě build orderu. Původní vestavěná [AI](#) se od vydání Starcraft: Brood war příliš (pokud vůbec) nezměnila, zatímco moderní způsob hraní, podle kterého jsou napsané build orderly v Princess Of Blades, se od toho, který se používal v roce 1998, značně liší. To má za následek to, že některé build-orderly jsou vůči taktice vestavěné [AI](#) velice zranitelné (zejména ty zaměřené na ekonomiku) a tak se může snadno stát, že při náhodné volbě build orderu na začátku zápasu, bude Princess Of Blades během prvních deseti minut poražen.

V prvním experimentu proti sobě stojí Princess Of Blades a vestavěná [AI](#) Starcraft: Brood war. V prvním experimentu nemá zapnutou Bayesovskou predikci, bot hraje postupně proti každé hratelné straně. Výsledky můžete vidět v tabulce 2.

V experimentu je vidět, že náhodná volba build orderu není dobrý nápad. Ve vzácných chvílích, kdy Princess Of Blades trefili vhodný build order dokázali hrát poměrně vyrovnanou hru a někdy dokonce i vyhrát. Nicméně případů, kdy se zvolený build order ukázal jako nevhodný, je podstatně více.

Druhým experimentem je test Bayesovské predikce build orderu. Výsledky můžete vidět v tabulce 3.

Matchup	počet her	počet výher	proc. výher	počet proher	proc. proher
ZvT	18	0	0%	18	100%
ZvP	8	0	0%	8	100%
ZvZ	7	0	0%	7	100%

Tabulka 3: Výkon Princess Of Blades proti vestavěné AI, se zapnutou Bayesovskou sítí

6.1.1 Vítězná ZvP

V této části stručně popíšu jeden z vítězných zápasů proti vestavěné AI, konkrétně v matchupu proti straně protoss.

- 0.00 Začíná zápas. Princess Of Blades náhodně volí build order 3HatchMuta.
- 0.00-2.29 Počáteční fáze zápasu. Princess Of Blades buduje svoji ekonomiku, postupně umísťuje první expand a základní budovy.
- 2.29-3.26 3HatchMuta je build order zaměřený na rychlý technologický rozvoj, proto Princess Of Blades začíná brzy těžit plyn a přechází na Lair tech.
- 3.26-4.28 Princess Of Blades staví pro zergy zásadní budovu Spire. Ta jim umožňuje produkovat létající jednotky, důležitou součást zergské strategie.
- 4.28-5.08 Další expand.
- 5.51 Vyrobení prvních Mutalisků.
- 5.08-7.30 Vestavěná AI provádí svůj prvotní útok. Jedná se o silnou jednotku Zealotů, za cíl si vybrali poslední expand Princess Of Blades.
- 7.59 Princess Of Blades i za cenu ztráty expandu odráží útok vestavěné AI. Pokračuje ve výrobě Mutalisků.
- 7.59-8.27 Princess Of Blades nashromáždila dostatek jednotek a rozhodla se zaútočit. Hlavní útočnou silou jsou Mutaliskové.
- 8.27-9.28 Princess Of Blades proniká do nepřátelské základny. Vestavěná AI se pokouší bránit běžnými protivzdušnými jednotkami (Dragoon, Photon Cannon), které ale nejsou proti Mutaliskům ve velkém počtu účinné.
- 8.43 Zaměřuje se na nepřátelské dělníky, aby způsobila co největší škodu ekonomice.
- 9.28-11.41 Hlavní nepřátelská základna je v troskách, nicméně vestavěná AI stále vlastní expandy a bojeschopné jednotky.

- 12.26 Armáda Mutalisků Princess Of Blades se přeskupuje v hlavní základně
- 12.26-13.12 Útok na objevený nepřátelský expand. Zničení zbývajících budov. Výhra Princess Of Blades.

6.1.2 Prohra v TvZ

Stručně popíšu jeden z prohraných zápasů proti vestavěné [AI](#), jde o matchup bez zapnuté Bayesovské predikce proti straně terran.

- 0.00 Začíná zápas. Princess Of Blades provádí náhodnou volbu build orderu a volí 12Hatch
- 0.00-2.06 Probíhá počáteční fáze hry, budování ekonomiky (ani jeden z hráčů nezvolil cheese taktiku)
- 2.06 Princess Of Blades se vydává vytvořit první expand
- 2.06-3.24 Princess Of Blades se zabývá technologickým rozvojem, staví základní budovy (spawning pool) a začíná těžit plyn
- 3.24 Princess Of Blades dokončuje spawning pool a staví první jednotky určené pro obranu, čtyři Zerglingy
- 3.24-3.49 Přejít na Lair tech
- 4.49 Začíná stavba kritické zergské budovy - spire
- 5.44 Princess Of Blades začíná těžit simultánně ze dvou zdrojů plynu
- 6.09 Dokončena výstavba Spire. Začíná produkce prvních pěti Mutalisků
- 6.39 Vestavěná [AI](#) zahajuje svůj první útok, nejdříve jsou proti němu vysláni Zerglingové
- 6.44 Zerglingové jsou zničeni. Do obrany nastupují mutaliskové
- 6.48-7.00 Mutaliskové bojují s nepřátelským útokem sestávajícím se z jednotek Marine, Medik a Firebat. Zatímco Firebat nemůže na Mutalisky vůbec zaútočit, kombinace [Marine and Medik \(M&M\)](#) je proti Mutaliskům velice účinná.
- 7.05 Mutaliskům se podaří zničit nepřátelské Marine. Zbytek armády už na ně nemůže útočit a je postupně zlikvidován
- 7.45 Zemřel poslední Firebat z útoku, zbývají jen Medikové
- 7.59 Útok je odražen, ale Princess Of Blades utrpěl značné škody obzvláště v oblasti ekonomiky a sběru surovin
- 8.22 Obnovena produkce Mutalisků
- 9.00 Princess Of Blades se s Mutalisky pokouší zaútočit na nepřátelskou základnu
- 9.30 Útok je odražen pomocí [M&M](#). Nezpůsobil skoro žádné škody

Matchup	počet výher	proc. výher	počet proher	proc. proher
ZvP	0	0%	10	100%

Tabulka 6: Výkon Princess Of Blades proti botovi MaasCraft, se zapnutou Bayesovskou sítí

- 11.20 Vestavěná [AI](#) provádí první expand. Princess Of Blades se jej pokouší napadnout s Mutalisky
- 11.52 Útok je opět neúspěšný. Vestavěná [AI](#) posiluje statickou obranu
- 12.14 Vestavěná [AI](#) shromažďuje jednotky na útok. Již má k dispozici Siedge Tanky, mocnou jednotku terranů pro útok na dálku
- 13.04 Vestavěná [AI](#) napadla expand Princess Of Blades
- 13.27 Expand zničen, Princess Of Blades ztrácí Lair tech
- 13.44 Hlavní základna Princess Of Blades napadena
- 14.26 Hlavní základna zničena. Princess Of Blades prohrává zápas

6.2 Výkon proti jiným botům

Bohužel vzhledem k tomu, že Princess Of Blades je postavena na [BWAPI](#) verze 4.0 a většina v současnosti vyvíjených botů pracuje stále pod [BWAPI](#) 3.7, je hraní s nimi proti Princess Of Blades na jednom počítači značně komplikované (jedním z možných řešení by bylo použít virtuální stroj a virtuální lokální síť, viz. [3](#)). Nicméně se podařilo najít funkční bot založený na [BWAPI](#) 4.0, jmenuje se MaasCraft [\[12\]](#) a je vytvořený pro hraní za stranu protoss.

Experiment je hraní 1v1 na náhodné mapě se zapnutou Bayesovskou predikcí v Princess Of Blades. Hrál se 10 her. Výsledky můžete vidět v tabulce [6](#).

6.2.1 Prohra proti MaasCraft

Zde stručně popíšu jednu z prohraných her proti MaasCraft.

- 0.00-3.31 Počáteční fáze, budování ekonomiky
- 2.29 Princess Of Blades obsazuje první zdroj plynu, bude se pokoušet o technologicky zaměřený build order
- 3.31 Princess Of Blades staví Spawning pool, MaasCraft už postavil jednu gateway a staví další
- 4.48 MaasCraft dokončil výrobu prvního zealota
- 5.07 Princess Of Blades staví první expand
- 5.32 MaasCraft začíná hromadnou výrobu zealotů
- 6.19 Princess Of Blades staví další expand, MaasCraft staví třetí gateway

- 6.52 Princess Of Blades staví prvních několik zerglingů
- 7.36 Princess Of Blades přechází na lair tech, MaasCraft obsazuje první zdroj plynu
- 8.13 Princess Of Blades staví hydralisk den
- 8.58 Princess Of Blades začíná produkci hydralisků
- 9.09 MaasCraft vyráží do útoku
- 9.40 Střet zerglingů a zealotů uprostřed mapy, zealoti vyhrávají
- 10.12 Princess Of Blades posílá do obrany hydralisky
- 10.34 Hydraliskové zničení
- 10.54 MaasCraft mívá hlavní základnu a útočí na expand
- 11.43 Expand zničen
- 12.26 Útok na hlavní základnu Princess Of Blades
- 14.46 Zničení hlavní základny Princess Of Blades prohrává

6.3 Výkon proti živým hráčům

Pro testování výkonu Princess Of Blades proti živým hráčům jsem rozeslal jednu z verzí bota několika svým spolužákům z katedry informatiky. Všichni hráči mají alespoň nějaké zkušenosti se Starcraft: Brood war a někteří hrají na poměrně vysoké úrovni. Počet experimentů se vzhledem k čistě dobrovolné a volnočasové povaze experimentu liší u každého z hráčů. Výsledky můžete vidět v tabulce 8.

6.3.1 Prohra proti Martinu Hořavovi

Zde popíšu jednu z her Princess Of Blades proti živému hráči, jedná se o zápas Zerg vs. Protoss, za protossy hrál Bc. Martin Hořava.

- 0.00-3.06 Počáteční fáze zápasu, budování ekonomiky
- 3.06 Princess Of Blades staví první expand, protivník začíná těžit plyn
- 3.53 Princess Of Blades staví Spawning pool
- 4.30 Protivník staví Gateway
- 5.01 Princess Of Blades se vydává postavit druhý expand
- 5.20 Princess Of Blades staví první bojové jednotky, tradičně se jedná o zerglingy

Hráč (rasa)	počet her	počet výher	proc. výher	počet proher	proc. proher
Jan Vytřísal (Protoss)	4	1	25%	3	75%
Martin Hořava (Protoss)	3	0	0%	3	100%
Radek Janoščík (Terran)	5	0	0%	5	100%

Tabulka 8: Výkon Princess Of Blades proti živým hráčům

- 6.07 Protivník postupuje v technologickém stromu a staví Cybernetics Core a Forge
- 6.10 Princess Of Blades začíná těžit plyn
- 6.45 Princess Of Blades staví Hydralisk Den
- 6.58 Protivník upgraduje Protoss Ground Weapons (důležitá součást taktiky Protossů proti Zergům)
- 7.52 Princess Of Blades upgraduje rychlost hydralisků, protivník začíná stavět Zealoty a Dragoony
- 8.00 Princess Of Blades začíná produkci hydralisků
- 8.51 Princess Of Blades vyráží do prvního útoku
- 9.35 V prostřední části mapy se střetnou Zerglingové Princess Of Blades s protivníkovými Dragoony
- 10.15 Zerglingové odráží Dragoony a postupují k nepřátelské základně
- 10.25 Doráží protivníkovi posily, Zealoti a Dragooni, Zerglingové zlikvidují Dragoony, ale podlehnou Zealotům
- 10.45 Princess Of Blades posílá do boje hydralisky
- 11.23 Střet hydralisků se zealoty
- 11.45 Hydraliskové odráží zealoty
- 12.15 Princess Of Blades se pokouší napadnout nepřátelskou základnu, ale je odražena
- 12.47 Protivník vyráží do protiútoků
- 13.59 Protivník se přeskupuje u svého expandu
- 14.35 Protivník útočí na první expand
- 15.42 První expand zničen
- 16.30 Na bojišti se objevuje první protivníkův corsair
- 19.23 Útok na druhý expand
- 19.54 Druhý expand zničen
- 23.55 Útok na hlavní základnu, Princess Of Blades prohrává

7 Závěr

V závěru nejdříve shrnu obsah práce a poté navrhnou několik dalších směrů kterými by se mohl vývoj Princess Of Blades vydat.

7.1 Shrnutí

V práci jsem pojednával o specializovaných AI pro hraní RTS videoher, konkrétně tedy Starcraft: Brood war. Čtenáře jsem seznámil s motivací pro vývoj těchto botů, samotnou videohrou Starcraft: Brood war a dalšími AI v této oblasti. Hlavní část práce zahrnovala popis struktury mnou vyvinutého bota jménem Princess Of Blades, jeho algoritmů a implementačních podrobností. V závěru práce jsem prezentoval některé drobné experimenty provedené na Princess Of Blades, proti různým hráčům (počítačovým i lidským), ale vzhledem k zaměření Princess Of Blades jako frameworku není výkon její základní implementace nijak oslnivý.

7.2 Budoucí práce

Budoucí práce na Princess Of Blades budou zřejmě souviset s implementací sofistikovaných algoritmů a náhradě naivních implementací agentů za jejich plnohodnotné protějšky. Kromě toho jistě existuje i prostor pro rozšíření a vylepšení stávajících skriptů a naivních implementací zejména v oblasti mikromanagementu, kdy si lze například představit specializovaný skript pro ovládání jednotek pro boj na krátkou vzdálenost, který činí současné naivní implementaci problémy a bojové části bota (simultánní útoky ze dvou stran apod.). Co se týče pokročilých algoritmů, například vzhledem k poměrně úspěšnému použití data-miningových metod pro predikci protivníkovi strategie, bylo by jistě zajímavé použít pro tento účel formální konceptuální analýzu. Možná by se daly použít algoritmy pro získávání formálních konceptů z dat pro generování build orderů. Princess Of Blades je pouze rámcem, frameworkem, který by měl být použit pro další práci, možnosti jsou jistě široké.

Seznam zkratek

AI Artificial Intelligence

AIUR Artificial Intelligence Using Randomness

BWAPI A Brood War API

EIS Extensive Intelligence Studio

i.i.d. Independent and Identically Distributed

M&M Marine and Medik

MCMC Marcov Chain Monte Carlo

MLE Maximal-Likelihood Estimation

RTS Real-Time Strategy

TSC Trobar Clus Starcraft

Literatura

- [1] CHURCHILL, David. *UAlbertaBot: StarCraft AI Competition Bot*. Dostupný z: [⟨https://github.com/davechurchill/ualbertabot⟩](https://github.com/davechurchill/ualbertabot).
- [2] KLEIN, Dan. *The Berkeley Overmind Project*. Dostupný z: [⟨http://overmind.cs.berkeley.edu/⟩](http://overmind.cs.berkeley.edu/).
- [3] WEBER, Ben. *EISBot: A repository for Ben Weber dissertation project*. - Google Project Hosting. Dostupný z: [⟨http://code.google.com/p/eisbot/⟩](http://code.google.com/p/eisbot/).
- [4] BWAPI. : *An API for interacting with Starcraft: Broodwar (1.16.1)* - Google Project Hosting. Dostupný z: [⟨http://code.google.com/p/bwapi/⟩](http://code.google.com/p/bwapi/).
- [5] BWAPI. : *The Brood War API*. Dostupný z: [⟨https://github.com/bwapi/bwapi⟩](https://github.com/bwapi/bwapi).
- [6] KING, Davis E. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*. 2009, roč. 10, s. 1755–1758.
- [7] *Easylogging++*. : *Cross-platform logging made easier for C++ applications*. Dostupný z: [⟨http://easylogging.muffihun.com/⟩](http://easylogging.muffihun.com/).
- [8] KAPOULKINE, Arseny. *pugixml: Light-weight, simple and fast XML parser for C++ with XPath support*. Dostupný z: [⟨http://pugixml.org/⟩](http://pugixml.org/).
- [9] SYNNAEVE, Gabriel; BESSIERE, Pierre. A bayesian model for opening prediction in rts games with application to starcraft. In. *Computational Intelligence and Games (CIG), 2011 IEEE Conference on*. 2011, s. 281–288.
- [10] GABRIEL, Iuhasz; NEGRU, Viorel; ZAHARIE, Daniela. Neuroevolution Based Multi-agent System for Micromanagement in Real-time Strategy Games. In. *Proceedings of the Fifth Balkan Conference in Informatics*. Novi Sad, Serbia: ACM, 2012, s. 32–39. BCI '12. Dostupný také z: [⟨http://doi.acm.org/10.1145/2371316.2371324⟩](http://doi.acm.org/10.1145/2371316.2371324). ISBN 978-1-4503-1240-0.
- [11] WENDER, Stefan; WATSON, Ian. Applying reinforcement learning to small scale combat in the real-time strategy game starcraft: broodwar. In. *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*. 2012, s. 402–408.
- [12] SOEMERS, Dennis. Tactical Planning Using MCTS in the Game of StarCraft1. 2014.