VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

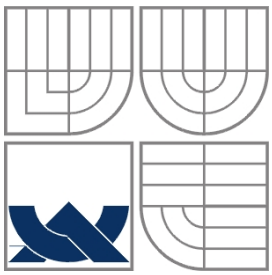
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SCRABBLE

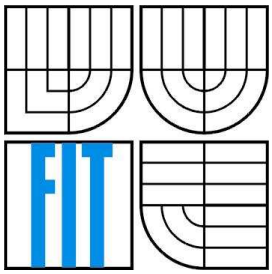
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. Radomír Picek
AUTHOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SCRABBLE

SCRABBLE

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

Bc. Radomír Picek

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. RNDr. Pavel Smrž, Ph.D.

BRNO 2008

Abstrakt

Tato diplomová práce se zabývá stolní společenskou hrou SCRABBLE a její realizací v podobě počítačové hry. Postupně rozebírá všechny důležité aspekty, které mají vliv na výkonnost dané implementace. Především potom zvolení vhodných datových struktur pro uchování použité slovní zásoby, ovlivňujících efektivitu generování tahů a výběr vhodných algoritmů s ohledem na maximální rychlost. Zvláštní důraz je přitom kladen na inteligenci umělého protihráče a jeho schopnost konkurovat nejenom amatérům, ale i profesionálním hráčům SCRABBLU.

Abstract

This thesis describes the social table game Scrabble, and its realization in the form of computer games. Gradually examines all important aspects that affect the performance of the implementation. Especially after the election of the appropriate data structures retained for the vocabulary, affecting the efficiency of generating moves, and the selection of appropriate algorithms with regard to the maximum speed. There is particular emphasis on artificial intelligence opponent and its ability to compete not only amateurs, but professional SCRABBLE players.

Klíčová slova

Scrabble, backtracking, teorie her, minimax, Trie, DAWG, GADDAG.

Keywords

Scrabble, backtracking, game theory, minimax, Trie, DAWG, GADDAG.

Citace

Picek Radomír: SCRABBLE. Brno, 2008, diplomová práce, FIT VUT v Brně.

Scrabble

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Doc. RNDr. Pavla Smrže, Ph.D.

……
Jméno Příjmení
Datum

Poděkování

Tímto bych chtěl poděkovat Doc. RnDr. Pavlu Smržovi, Ph.D. za odborné vedení a cenné rady ohledně implementace systému. Dále pak za poskytnuté materiály a mnoho zajímavých zdrojů, které mi pomohly při tvorbě této práce.

Zvláštní poděkování patří spoluautorovi oficinálního slovníku SCRABBLE pro českou verzi hry a profesionálnímu hráči SCRABBLE Ing. Petru Vetešníkovi. Poděkovat bych chtěl především za poskytnuté informace ohledně českých slovníků, materiály týkající se přípustnosti slov a vysvětlení některých taktik, používaných profesionálními hráči.

Obsah

Obsah 1
1 Úvod 3
2 Představení hry SCRABBLE 5
2.1 Historie 5
2.2 Pravidla 6
2.2.1 Začátek hry 6
2.2.2 Průběh hry 6
2.2.3 Pokládání kamenů 6
2.2.4 Bodování 7
2.2.5 Konec hry 8
2.2.6 Přípustná slova 9
3 Datové struktury pro uložení slovníku 10
3.1.1 Naivní přístup 10
3.1.2 Strom 10
3.1.3 Trie 10
3.1.4 Dawg 11
3.1.5 Dvojcestný Dawg (GADDAG) 11
4 Generování tahů 14
4.1 Analýza možných tahů 14
4.1.1 Cross-množina 14
4.1.2 Kotvy slova 14
4.2 Backtracking 15
4.2.1 Dawg 15
4.2.2 Backtracking pro GADDAG 17
5 Teorie Her 19
5.1 Maticové hry 19
5.2 Výběry strategií 20
5.2.1 John von Neumanova teorie 20
5.2.2 Strategie ve stromech 21
5.2.3 Použití minimaxu pro průchod stromem 22
5.2.4 α-β prořezávání stromu 22
6 Herní strategie 24
6.1 Počáteční fáze 25
6.1.1 Hladová strategie 25

6.1.2 Zásobníková heuristika 25
6.1.3 Uzavřenost prvního tahu 26
6.1.4 Monte Carlo 27
6.2 Konečná fáze 27
6.3 Přizpůsobení hry slabším hráčům 28
7 Implementace 30
7.1 Použité technologie 30
7.2 Lexikon 31
7.2.1 Použitý slovník 31
7.2.2 Reprezentace slovníku 32
7.3 Generátor tahů 32
7.3.1 Vyhledání kotev 32
7.3.2 Generování možných tahů 33
7.3.3 Výběr vhodného tahu 34
7.3.4 Provedení tahu 37
7.4 Stupně obtížnosti 37
8 Praktické výsledky 40
8.1 Vzájemné porovnání jednotlivých obtížností 40
8.2 Porovnání s reálnými hráči 43
9 Závěr 45
Literatura 46
Seznam příloh 48
Hrací deska 49
Ukázka grafického výstupu programu 50
Pravidla přípustnosti slov ČAS 51

1 Úvod

I v dnešní době digitálního věku se lidé stále vrací k tradičním stolním hrám. Mezi ty nejznámější, jako jsou Dáma, Šachy nebo Monopoly, patří i SCRABBLE. SCRABBLE je pravděpodobně nejrozšířenější a nejoblíbenější slovní deskovou hrou na světě. Hraní podle přesných pravidel testuje hráčovu vynalézavost a tvořivost, kdy je hráč nucen vytvářet stále nová slova z předem zadaných písmen.

Cílem práce bude vyvinout SCRABBLE v podobě funkční webové aplikace. Součástí bude i umělý protihráč který by měl být konkurenceschopným protivníkem nejen obyčejným amatérům, ale zároveň by měl dosti potrápit i hráče profesionální. Protože je SCRABBLE hrou se statisíci slovy a jejich různými tvary, budou tato slova provázána s výkladovým slovníkem objasňující jejich význam.

První kapitola uvádí čtenáře do dané problematiky, čtenář by se zde měl dozvědět co je hlavním cílem práce a jaké části obsahuje.

Druhá kapitola se zabývá představením deskové hry SCRABBLE, čtenář se dozví něco o historii a místě vzniku této hry. Součástí budou i detailní pravidla platná podle ČAS (česká asociace SCRABBLU [CAS])

Ve třetí kapitole se budeme zabývat klíčovým faktorem a to je uložení slovníku. Postupně probereme jednotlivé použitelné datové struktury, které jsou pro daný problém vhodné. Vysvětlíme si jejich klady a zápory.

V další části se budeme věnovat algoritmům generování tahů. Seznámíme se s pojmy jako jsou *kotva* nebo *crossmnožina* a jak se efektivně vypořádat s omezeními danými aktuálním stavem hrací plochy. Následně rozebereme způsoby, jak rychle a efektivně nalézt slova sestavitelná z našeho zásobníku v kontextu hracího pole.

Modelováním her mezi dvěma protihráči se věnuje obor teorie her, v páté části si tento obor představíme a zjistíme pomocí jakých postupů zhodnotit užitečnost námi provedených akcí. Ukážeme si, jak souvisí vývoj hry se stavovým prostorem a jak tyto prostory procházet a získávat z nich užitečné informace důležité pro naše rozhodování. Závěrem také zjistíme, že existují různé způsoby optimalizace těchto postupů.

Šestá kapitola se věnuje herním strategiím SCRABBLE. Zjistíme, že nejvýše hodnocený tah není někdy nejlepší, ale že je nutné uvažovat o tahu v širších souvislostech jako jsou ofenzivní a defenzivní taktika, využívání prémiových polí nebo také schovávání významných hracích kamenů pro pozdější tahy. Dozvíme se zde, jaké nejčastější postupy jsou používány k modelování inteligentního protihráče a proč je nutné, aby základní algoritmus generování tahů pracoval co nejrychleji.

Sedmá část se zabývá realizací celého systému v jazyce Java. Jsou zde shrnuty informace o použitých technologiích a jejich výhodách, pro které jsme se je rozhodli použít. Větší prostor je věnován implementaci klíčových problémům, je zde osvětleno, jakým způsobem byly dané problémy

řešeny a z jakého důvodu jsme se pro daná řešení rozhodli. Probereme tady jaký slovník použít a jakou konkrétní datovou strukturu pro jeho uložení zvolit. Dále se dozvíme, jak přesně funguje generátor tahů a jaké optimalizace byly použity pro dosažení lepších výsledků. Na závěr jsou popsány všechny dostupné úrovně hry a jejich vlastnosti.

Osmá kapitola se zabývá testováním implementovaného systému. Pomocí jednoduchého simulačního nástroje jsou provedeny simulace her jednotlivých úrovní obtížnosti a dosažené výsledky analyzovány. V druhé části kapitoly je pak provedeno porovnání naší aplikace se statistikami profesionálních hráčů.

V závěrečné části jsou zhodnoceny dosažené výsledky a konzultovány směry budoucího vývoje.

2 Představení hry SCRABBLE

SCRABBLE je slovní hra pro dva, tři nebo čtyři hráče. Hra spočívá v tvoření vzájemně do sebe zapadajících slov, podobně jako v křížovkách, na herním plánu pomocí hracích kamenů s jednotlivými písmeny o různé bodové hodnotě. Cílem každého hráče je získat co nejvyšší počet bodů tím, že hracích kamenů užívá v takových kombinacích a umístěních, která jsou nejvýhodnější jak z hlediska hodnoty jednotlivých kamenů, tak co se týká využití prémiových polí herního plánu.

2.1 Historie

V roce 1931 zuřila v Americe hospodářská krize. Jedním z mnoha kdo přišel o práci byl i mladý architekt z města Poughkeepsie jménem Alfred Mosher Butts, načeš se rozhodl rozvinout svou zálibu v hrách a v jazyku. Butts neměl přiliž v lásce hry s kostkou, protože v nich šlo čistě o štěstí. Na druhé straně zastával názoru, že čistě strategické hry jako jsou šachy jsou pro obyčejné lidi příliš složité. Proto se rozhodl vyvinout hru, která by měla být příjemným kompromisem. V roce 1931 tak spatřila světlo světa hra Lexico. Lexico se hrálo bez desky, smyslem bylo skládat slova, jež byla hodnocena podle jejich délky a určitých bonusů za některá písmena. Všechny snahy o získání patentu na tuto slovní hru a prosazení u výrobců her končily nezdarem.

V roce 1938 přivedla Buttse jeho obliba v křížovkách na nápad propojit tvorbu slov s hracím plánem. Všechny prototypy vytvořil Alfred Butts doma ze skládací šachovnice a dřevěných políček s názvy písmen. Hra prošla mnoha změny jména, až se ustálil název „Criss-Crosswords“. Už v té době obsahovala hra některé vlastnosti, jež se dochovaly až do dnešní doby. Hrálo se na herním plánu o velikosti 15x15 palců a každý hráč vlastnil 7 hracích kamenů, která si uchovával v zásobníku, stejný byl i způsob skládání slov a ohodnocení písmen. Přesto byla hra patentním úřadem odmítnuta.

Ke hře se Butts vrátil až v roce 1948, tentokrát se svým přítelem Jamesem Brunotem. Oba došli k názoru, že hra ještě potřebuje pár změn. Na hrací plán byla přidána prémiová pole, zjednodušili pravidla a hře dali konečný název SCRABBLE. V této podobě sepsali žádost o patent, která byla 1.prosince 1948 přijata.

I přes počáteční neúspěchy, kdy firma prvních pár let vykazovala ztrátu, se podařilo situaci stabilizovat, najmout si starou halu a rozjet výrobu. Licenční práva byla nakonec podána firmě Spear's. V roce 1991 se konalo první mistrovství světa ve SCRABBLE v Londýně. Více detailů [CAS].

2.2 Pravidla

Na tomto místě si shrneme pravidla SCRABBLE, jak jsou v současnosti schválena českou asociací SCRABBLE [CAS]. K původní verzi z roku 1998 byl 15.5.1999 schválen doplněk, který vešel v platnost 1.1.2000: nejdůležitější změnou bylo přijetí Akademického slovníku cizích slov mezi uznávané prameny.

2.2.1 Začátek hry

Před každou hrou se provádí losování. Všechny hrací kameny vložíme do sáčku a zamícháme. Každý hráč si vytáhne jeden kámen. Hráč, který si vytáhne písmeno nejblíže k začátku abecedy, zahajuje hru. Po losování vložíme tažené kameny zpět do sáčku a znovu zamícháme. Každý z hráčů si pak vybere sedm kamenů a uloží je do svého zásobníku. Jeden z hráčů zapisuje skóre, zároveň se sám může účastnit hry.

2.2.2 Průběh hry

První hráč sestaví ze dvou, či více svých kamenů slovo a umístí je na herní plán tak, aby se bylo možno číst shora dolů nebo zleva doprava. Alespoň jedno z písmen slova musí ležet na centrálním poli hracího plánu (hvězdě).

Tah končí, sečte-li hráč dosažené body a oznámí je zapisovateli. Poté si hráč ze zbývajících kamenů na stole (resp. sáčku) vezme tolik, kolik jich použil při tvoření slova – takže má opět sedm kamenů.

Hráči se střídají zprava doleva. Další hráč, který je na řadě, přidá jeden nebo více kamenů na hrací plochu tak, aby vytvořil nová slova.

Všechny kameny přidané v jednom tahu musí ležet v jednom sloupci či řádku herního plánu. Diagonální slova se nepočítají. Hrací kameny musí vytvořit kompletní slovo. Pokud se současně dotýkají i jiných kamenů v sousedních řádkách a sloupcích, musí i s nimi se všemi vytvořit smysluplná slova. Hráči přísluší plná odměna za všechna slova, která svým tahem vytvořil nebo která obměnil.

2.2.3 Pokládání kamenů

Nová slova mohou vzniknout pouze:

a) přidáním jednoho či více kamenů ke slovu, které již na hracím plánu je
b) umístěním slova kolmo k již existujícímu slovu na hracím plánu – takto vzniklé slovo buď využívá jednoho z písmen slova již existujícího nebo je přidáním písmene na konec či začátek modifikuje. Nové slovo může rovněž „přemostit“ dvě či více již existujících slov.

c) Položením celého nového slova rovnoběžně vedle již existujícího slova tak, že těsně sousedící kameny vytvoří rovněž smysluplná slova. Pravidla povolují v jednom tahu položení kamenů pouze v jednom řádku či sloupci. I tak lze vytvořit více slov.

Žádný kámen nelze přesunout, bylo-li s ním už jednou hráno.

Dva prázdné kameny mohou být užity na místě libovolného písmene (obdobně jako „žolík“ ve stejnojmenné karetní hře). Hráč oznámí, jakou úlohu prázdný kámen ve slově má. Tento jeho význam je po dobu trvání hry neměnný.

Každý z hráčů může využít svého tahu k výměně některých či všech svých kamenů. Výměnu provede tak, že položí své kameny lícem na stůl, vybere z ostatních na stole či v sáčku odpovídající počet nových kamenů a pak své staré kameny zamíchá mezi zbylé kameny na stole resp. v sáčku. V tomto kole již netvoří slova a hraje, když na něj opět přijde řada.

Namísto tvoření slov na herním plánu nebo výměny kamenů může hráč ohlásit PASS – tj. vynechat tah. To může udělat vždy ať je či není schopen z písmen ve svém zásobníku utvořit nová slova.

2.2.4 Bodování

Zapisovatel vede záznamy o stavu skóre každého hráče. Doplňuje je po každém tahu. Bodová hodnota každého písmene je vyznačena číselně v dolním rohu hracího kamene. Bodová hodnota prázdného kamene je nulová.

Body za každý tah jsou součtem bodových ohodnocení všech kamenů ve všech slovech vytvořených, či pozměněných tímto tahem. K tomuto prostému součtu se ještě přičítají prémie, pramenící z případného umístění některého z kamenů na prémiové pole herního plánu.

Písmenné prémie: Světle modrá pole zdvojují bodové ohodnocení písmene, které je na ně umístěno, tmavomodrá pole je ztrojnásobují.

Slovní prémie: Body za celé slovo se násobí dvěma (resp. třemi), je-li jedno z písmen slova umístěno na světle červeném (resp. tmavě červeném) poli.

Vyskytnou-li se současně ve slově i písmenné prémie, je třeba je započítat před slovními. Pokud jedno slovo obsahuje dvě slovní prémie, počítají se obě – lze tedy (pokud takové slovo vytvoříme) získat i čtyř až devítinásobek součtu bodových ohodnocení jednotlivých písmen.

Povšimněte si, že centrální pole (hvězda) je světle červené, proto je první slovo partie hodnoceno dvojnásobně.

Všechny prémie (písmenné i slovní) se započítávají jen v tahu, ve kterém je na ně vložen kámen, při dalších tazích se započítávají už jen hodnoty písmene uvedené na hracím kameni.

Pokud je umístěn na červeném poli prázdný kámen, zdvoj- či ztrojnásobuje se slovní skóre přesto, že prázdný kámen sám nemá žádné bodové ohodnocení.

Pokud se jedním tahem vytvoří dvě či více slov, všechna se bodují. Společná písmena se započítávají vždy (v každém slově) včetně svých případných prémií.

Každý hráč který ve svém tahu užije všech sedmi svých kamenů, obdrží prémii 50 bodů (tzv. *bingo*). Tato prémie se připočte k bodům získaným v tomto tahu až po započtení písmenných i slovních prémií.

Na konci hry je skóre každého z hráčů zmenšeno o hodnotu kamenů, které nepoužil. Pokud některému z hráčů nezbyl v zásobníku žádný kámen, k jeho skóre se přičtou hodnoty všech kamenů, které zbyly ostatním hráčům.

Přehled všech písmen platných pro českou verzi SCRABBLE, jejich počet a bodové ohodnocení se nachází v následující tabulce, na rozdíl od české abecedy neobsahuje Q a CH (písmeno „ch“ se v českém SCRABBLE hraje pomocí písmen C a H).

	ks			ks			ks			ks			ks
A$_1$	5		É$_3$	2		K$_1$	3		R$_1$	3		Ů$_4$	1
Á$_2$	2		Ě$_3$	2		L$_1$	3		Ř$_4$	2		V$_1$	4
B$_3$	2		F$_5$	1		M$_2$	3		S$_1$	4		X$_{10}$	1
C$_2$	3		G$_5$	1		N$_1$	5		Š$_4$	2		Y$_2$	2
Č$_4$	1		H$_2$	3		Ň$_6$	1		T$_1$	4		Ý$_4$	2
D$_1$	3		I$_1$	4		O$_1$	6		Ť$_7$	1		Z$_2$	2
Ď$_8$	1		Í$_2$	3		Ó$_7$	1		U$_2$	3		Ž$_4$	1
E$_1$	5		J$_2$	2		P$_1$	3		Ú$_5$	1			

2.2.5 Konec hry

Hra končí, jestliže některý hráč využil všechny své kameny a nemůže si již vylosovat žádné další. Nikdo další už nesmí táhnout a hráčům se upraví skóre podle kamenů, které jim zůstaly v ruce - viz Bodování.

Hra ovšem nekončí, pokud již nelze losovat nové kameny, ale všem hráčům ještě nějaké zbývají v ruce a hráči s nimi umí táhnout. Teprve když se všichni hráči ve dvou po sobě jdoucích kolech vzdají tahu, hra skončí.

2.2.6 Přípustná slova

Všechna slova uvedená ve standardním českém slovníku lze použít ve hře. Výjimku tvoří slova tvořená jedním písmenem, zkratky, předpony, přípony, slova obsahující apostrof nebo spojovník, vlastní jména a názvy psané s velkým písmenem na začátku.

Jako přípustná platí slova, která jsou v přípustném slovníku uvedena jako heslová slova. U podstatných jmen užívejme jen 1. pádu jednotného i množného čísla (např. ŽENA, ŽENY), u přídavných jmen 1. pádu obou čísel všech rodů (např. ČERNÝ, ČERNÁ, ČERNÉ, ČERNÍ), u sloves jen 1., 2. a 3. osoby jednotného i množného čísla v přítomném čase oznamovacího způsobu (např. NESU, NESEŠ, NESE, NESEME nebo NESEM, NESETE, NESOU) a tvary rozkazovacího způsobu (např. NES, NESTE). Obdobná omezení platí i pro zájmena. O slovech ostatních slovních druhů je vhodné se předem dohodnout, zda je připouštět či nikoliv.

Dvou-písmenná slova: dohodneme-li se na přípustnosti i jiných slovních druhů, můžeme sestavovat i předložky (např. NA, SE), spojky (např. AČ), citoslovce (např. AU, JE), názvy písmen (např. řeckého PÍ, našeho CÉ, EF), dětská slova (např. PA).

Cizí slova ve standardním českém slovníku považujeme za slova do češtiny přejatá a tudíž přípustná pro naši hru. Každé slovo může být napadeno před zahájením tahu dalšího hráče. Pokud je jedno z právě vytvořených slov uznáno za neplatné, hráč si vezme nazpět své kameny a ztrácí tah. Kompletní pravidla přípustnosti slov lze nalézt v příloze 3.

3 Datové struktury pro uložení slovníku

Množství různých algoritmických technik a heuristik je použito za účelem rychlého vyhledávání a generování slova. Klíčovým faktorem algoritmu je uložení slovníku. V následující kapitole si tak probereme všechny použitelné reprezentace slovníku a řekneme si něco o jejich výhodách a nevýhodách.

3.1.1 Naivní přístup

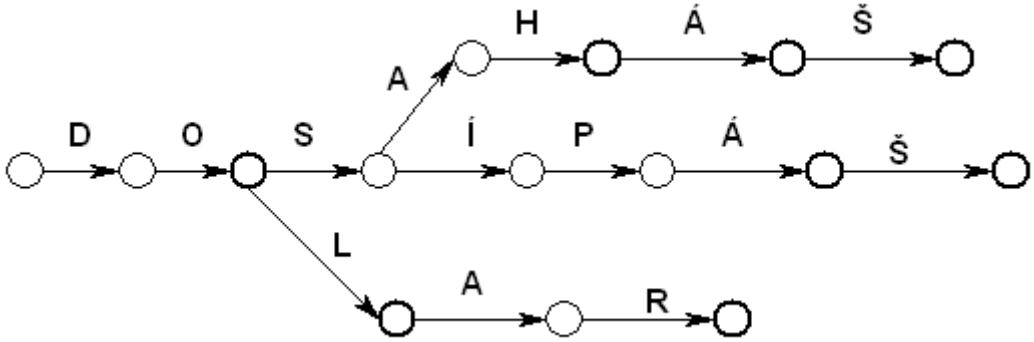
Naivní přístup zahrnuje textový soubor se seznamem slov lexikonu, asi nejčastější řešení je pole řetězců seřazených podle abecedy. Přestože tento způsob umožňuje poměrně rychlou kontrolu správnosti slova (např. s využitím binárního vyhledávání), neumožňuje generování slov pro danou herní situaci. Kromě toho nevyužívá taková datová struktura příliš efektivně paměť počítače.

3.1.2 Strom

Zajímavou formu představil Stuart Shapiro [Sha79] v jeho SCRABBLU z roku 1977, kdy pro uložení slovníku použil stromovou strukturu písmen, kde ke každé cestě dolů stromem existuje seznam slov, které jdou z těchto písmen složit. Písmena podél cesty jsou sestupně seřazena podle hodnoty (ceny) písmena na hracím kameni. Hodnotnější písmena jsou ve stromové struktuře řazena výše z toho důvodu, abychom vyhledali jako první důležitější (lépe hodnocená) slova. Uplatní se především v případech, že hledání je něčím přerušeno a nedoběhne až do konce.

3.1.3 Trie

Slovo *trie* pochází z anglického „retrieval“ (získávání vyhledání). Je to stromová datová struktura, jejíž hrany jsou ohodnoceny jednotlivými písmeny. Každé slovo ze slovníku je tak reprezentováno cestou z kořene stromu do koncového uzlu. Pokud tedy 2 slova začínají stejnými písmeny, sdílí první části cesty, proto je někdy *trie* označována jako prefixový strom. Kořen datové struktury je pak asociován s prázdným řetězcem. Koncové uzly nazýváme terminální a jsou v grafu *trie* speciálně označeny.

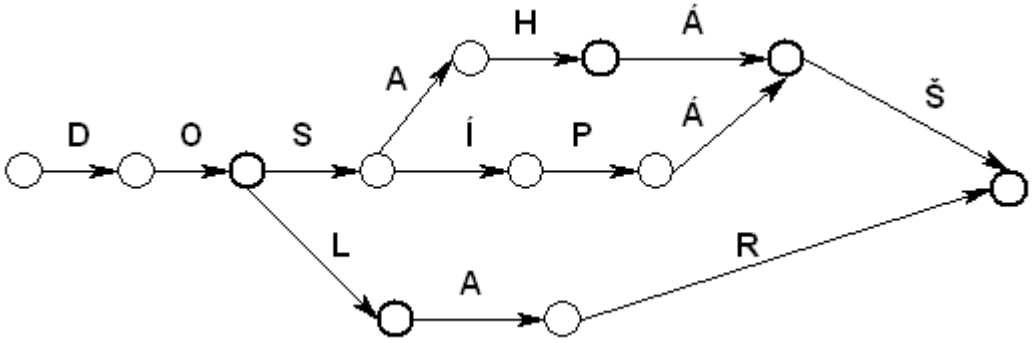


Obrázek 1: Trie pro slova do, dosah, dosahá, dosaháš, dosípá, dosípáš, dol, dolar

3.1.4 Dawg

Trie bývá poměrně mohutná. Pokud jí reprezentujeme jako graf, místo stromu, můžeme efektivně zmenšit velikost, přičemž zachováme všechny její ostatní přednosti. *Trii* můžeme považovat za konečný automat, kde uzly budeme chápeme jako stavy a hrany jako přechody a terminální uzly jako koncové stavy. Jazyk takového automatu bude prezentovat množinu slov, jenž chceme přijímat. Pro každý jazyk existuje množství automatů, my hledáme automat s minimálním množstvím stavů. Protože slovník pro SCRABBLE obsahuje konečné množství slov, je jednoduché najít daný minimální konečný automat v poměrně rychlém čase.

„Directed Acyclic Word-Graph“ (dále jen *Dawg*) je tedy *Trie* převedená na graf s tím že všechny shodné podčásti jsou sloučeny. Pro klasické slovníky (například slovník českého jazyka pro SCRABBLE) dosáhneme až 5x menšího počtu stavů.



Obrázek 2: Dawg pro slova do, dosah, dosahá, dosaháš, dosípá, dosípáš, dol, dolar

3.1.5 Dvojcestný Dawg (GADDAG)

Zatímco datová struktura *Dawg* je výhodná pro reprezentaci slovníků a kontrolu pravopisu, není to úplně nejefektivnější volba pro SCRABBLE. Algoritmy založené na *Dawgu* jsou extrémně rychlé, pokud by nám stačilo najít pouze nejlépe ohodnocený tah, plně by nám vyhovoval. Přestože by

takový program porazil většinu lidí, proti profesionálním hráčům by již nestačil. I když program zná všechna slova a vybere z nich nejlépe ohodnocený, projeví se proti tak zkušeným protihráčům jakákoliv absence taktiky. Stále by nebylo zapotřebí rychlejšího algoritmu, pokud bychom mohli strategii modelovat efektivně pomocí lehce spočitatelných heuristických funkcí. V našem případě tohle představuje poněkud složitější problém, jelikož je SCRABBLE (v literatuře [AJ88], [Gor94]) označována jako „hra nedostatečné informace“. Pokud chceme modelovat inteligentnější strategii, potřebujeme ještě rychlejší algoritmus.

Zdrojem menší efektivity *Dawgu* je nedeterministické generování předpon, kdy vytváříme prefixy, které nemohou být dokončeny se zbývajícími hracími kameny v zásobníku nebo již odehranými kameny na desce. Tento nedostatek můžeme odstranit použitím dvojcestného *Dawgu*, pro který se vžil název *GADDAG*. Prakticky se jedná o *Dawg* pro jazyk L= {REV(x)□y | xy je slovo a x není prázdné}, kde □ je oddělovač. Každé slovo má tedy právě tolik reprezentací, kolik obsahuje písmen. To znamená že, před minimalizací, by měla být celá struktura přibližně N-krát větší než neminimalizovaný související *Dawg*, kde N je průměrná délka slova ve slovníku.

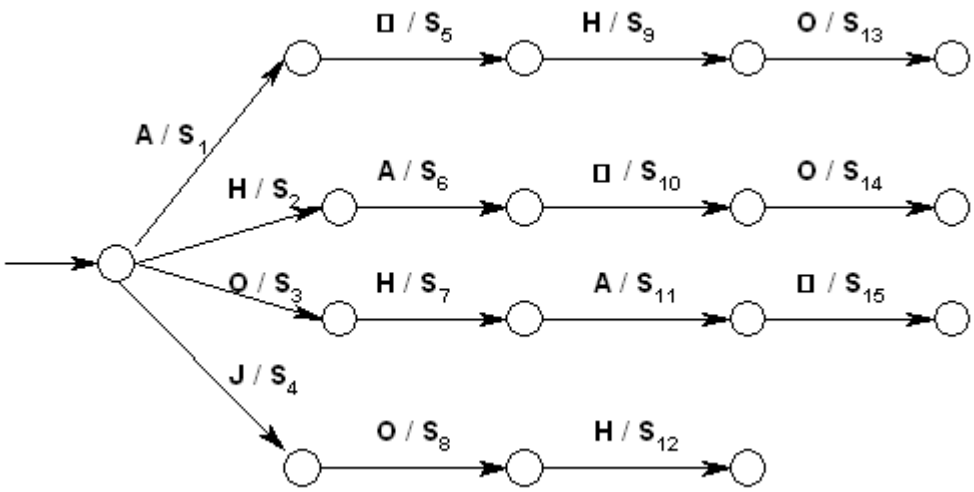
Každé slovo lexikonu může být generováno z každého jeho písmene tak, že umisťujeme kameny nejprve nalevo od *kotvy*. Kameny umisťujeme jeden za druhým, současně se v datové struktuře pohybujeme po hranách jejichž ohodnocení odpovídá hraným kamenům, až narazíme na znak □, poté začneme kameny pokládat napravo od *kotvy*, přičemž se v grafu pohybujeme stále stejným způsobem.

Stejně dobře může být *GADDAG* reprezentován i *Dawgem* pro jazyk L={ y□REV(x) | xy je slovo a y není prázdné }, pouze bychom odehrávali kameny nejprve vpravo od *kotvy* a po dosažení znaku □ bychom začali kameny pokládat vlevo od *kotvy*.

Minimalizace datové struktury dosáhneme následujícím algoritmem.

1. Pokud je y v REV(x) □y prázdné, znak □ je vynechán
2. Stavy jsou specifikovány množinou hran které z něj vedou a k nim příslušným písmenům
3. Hrany jsou specifikovány (a) cílovým stavem
   (b) množinou písmen,

Na obrázku č. 3 vidíme neminimalizovaný *GADDAG*, který obsahuje reprezentaci slova „ahoj“. Vidíme 4 odlišné cesty A□HOJ, HA□OJ, OHA□J, JOHA.



Obrázek 3: podgraf neminimalizovaného GADDAGu pro slovo AHOJ

4 Generování tahů

Generování tahů představuje základní „hnací motor“ všech programů SCRABBLE. Jde o nalezení všech možných platných tahů pro dané rozestavení kamenů na hrací ploše a momentální obsah zásobníku. Umělý protihráč pak z těchto tahů vybírá podle své strategie tah nejvýhodnější.

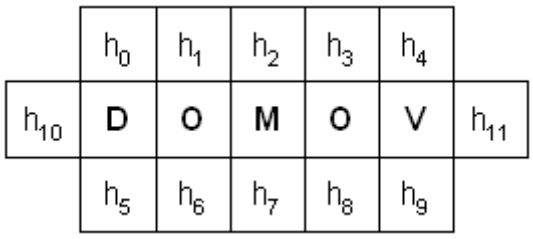
4.1 Analýza možných tahů

Ve SCRABBLE existují 2 druhy tahů, horizontální (odehráváme hrací kameny do jedné řady) a vertikální (kameny odehráváme do sloupce). Pro jednoduchost bude stačit, pokud budeme uvažovat pouze tahy horizontální, tahy vertikální jsou vlastně tahy horizontální, jenom převrácené o 90 stupňů.

4.1.1 Cross-množina

Pokud hrajeme horizontální tah a některý z hracích kamenů se přímo dotýká (nahoře nebo dole) kamene již odehraného, musí toto spojení dávat smysluplné slovo. Protože většinou takto budeme do sloupce přidávat pouze jeden hrací kámen, je jednoduché pro každé prázdné hrací pole předpočítat množinu písmen, jenž budou tvořit platná svislá slova při horizontálním tahu skrz tohle pole. Tyto množiny je nutné generovat před začátkem každého tahu a nazýváme je *cross-množiny*. Protože je ale množství hracích polí ovlivněné jedním tahem poměrně malé, dojde po každém tahu k pouze několika málo přepočtům.

Na obrázku č. 4 můžeme vidět jednotlivé *cross-množiny* pro slovo „domov“, například pro pole h_{11} by tato množina obsahovala hodnoty { a, e, y, u, ů}.



Obrázek 4: Cross-množiny kolem slova "domov"

4.1.2 Kotvy slova

Každé nové slovo musí zahrnovat nově položený hrací kámen, připojený vedle již odehraného hracího kamene (jedinou výjimku tvoří první tah, kdy je celá hrací plocha prázdná,v takovém případě

musí nově položené slovo ležet na středovém poli hrací plochy). Nejlevější takové místo pro připojení nazýváme *kotvou slova*.

Spočítáním *cross-množiny* a *kotev* pro daný řádek, můžeme generovat slovo bez ohledu na ostatní řádky nebo sloupce. Tím jsme jednoduše redukovali celý problém na jedinou dimenzi, kde využíváme pouze zásobník, obsah daného řádku na hrací ploše, body *cross-množiny* a *kotvy*.

Obrázek 5: Kotvy řádku k_1-k_4

4.2 Backtracking

Při generování nových tahů se prakticky vždy využívá některého z algoritmů odvozených od *backtrackingu*. Jde o způsob řešení kombinatorických problémů pomocí algoritmu, kterému je dovoleno prohledávat stavový prostor do doby, než se dostane do „slepé uličky“, v takovém případě se algoritmus vrací o krok zpět a snaží se jít jinou cestou. V případě generování slov ve SCRABBLE jde potom o inkrementální tvoření posloupností písmen pomocí kamenů v zásobníku a kamenů na hrací ploše a následné ověřování, zda daná posloupnost tvoří platné slovo. Pokud daná posloupnost není platná, vrací se algoritmus o krok zpět a zkouší tvořit slovo pomocí jiného kamene zásobníku. V dnešní době se používají pro generování tahů prakticky pouze 2 algoritmy, oba vychází z *backtrackingu* a jejich odlišnosti vychází z rozdílného způsobu uložení slovníku.

4.2.1 Dawg

Generování tahů jednoduše rozdělíme do dvou částí. Pro každou *kotvu* generujeme všechna možná slova takto:

1) Najdeme všechny možné levé části zakotveny na daném místě, pro řádek jsou to všechny hrací kameny nalevo od dané *kotvy*.
2) Pro každou nalezenou levou část hledáme odpovídající pravé části, skládající se z kamenů na políčku *kotvy* a napravo od ní.

4.2.1.1 Generování levé části

Levá část se skládá buď z kamenů již odehraných nebo z kamenů nových (ze zásobníku). V prvním případě se jednoduše podíváme z kterých písmen se levá část skládá a podle nich se následně v datové

struktuře pohybujeme. Druhý případ je složitější, musíme najít všechny možné levé části, které lze v daném kontextu sestavit.

Výše jsme si definovali *kotvu* jako nejlevější místo připojení k již existujícímu slovu. To znamená, že *cross-množina* je netriviální. Na rozdíl od všech polí na kterých se rozkládá levá část generovaného slova, všechna tato pole mají triviální *cross-množinu* a proto jsme při tvorbě levé části omezeni pouze zásobníkem. Současně s tím procházíme související datovou strukturu. Jedinou věcí kterou potřebujeme znát, je maximální délka levé části. Tu zjistíme spočítáním všech volných polí nalevo od *kotvy*, které mají netriviální *cross-množiny*.

```
function LevaCast( CastecneSlovo, uzel N v Dawgu, limit)
   PravaCast( CastecneSlovo, N, PoleKotvy)
   if limit > 0
      for kazdou hranu E vedouci z N
         if pismeno L nad hranou E je v zasobniku

            odstran pismeno L ze zasobniku
            N' = uzel na konci hrany E
            LevaCast( CastecneSlovo + L, N', limit - 1)
            vrat pismeno L do zasobniku
         endif
      endfor
   endif
end of function
```

Obrázek 6: Algoritmus generování levé části slova

Pro generování všech možných tahů z `PoleKotvy` s `k` volnými poli nalevo od ní vypadá potom volání takto `LevaCast( "", kořenový uzel Dawgu, k)`. Pokud nastane situace taková, že pole napravo od `PoleKotvy` je již obsazené, tzn. že `k = 0`, voláme rovnou algoritmus pro generování pravé části (obr. 7).

4.2.1.2 Generování pravé části

Po vytvoření levé části se můžeme pokusit slovo dokončit postupným přidáváním písmen doprava a současným průchodem související podčásti *Dawgu*. Tento průchod je na rozdíl od generování levé části omezen nejen písmeny v zásobníku, ale současně i kameny již položenými na hracím plánu a jednotlivými *cross-množinami*. Při pokládání nových kamenů napravo může nastat situace, že další uvažované pole nebude prázdné, ale bude se na něm nacházet kámen odehraný během předešlých tahů. Tahle možnost neukončí generování nového slova, pouze pokud je to možné, vezmeme odehraný kámen a použijeme ho místo kamene ze zásobníku.

```
function PravaCast( CastecneSlovo, uzel N v Dawgu, pole)
   if pole je prazdne
      if N je koncovy uzel
          PridejDoSeznamuSlov( CastecneSlovo )
      for kazdou hranu E vedouci z N
         if (pismeno L nad hranou E je v zasobniku) a
           (L je v mnozine bodu krizeni pole)

            odstran pismeno L ze zasobniku
            N' = uzel na konci hrany E
            dalsiPole = pole napravo od aktulniho pole
            PravaCast( CastecneSlovo + L, N', dalsiPole )
         endif
      endfor
   endif
   else
      L = pismeno polozene na poli
      if N ma hranu E oznacenou L vedouci do uzlu N'
         dalsiPole = pole napravo od aktulniho pole
         PravaCast( CastecneSlovo + L, N', dalsiPole )
      endif
   endelse
end of fuction
```

Obrázek 7: Algoritmus generování pravé části slova

4.2.2 Backtracking pro GADDAG

První zmínka o tomto algoritmu se nachází v [AJ88], detailní popis můžeme nalézt v [Gor94]. Algoritmus *GADDAG* pro generování všech možných tahů pro daný zásobník je ukázán na obrázku 7 ve formě *backtrackingu*. `Gen( 0, NULL, RACK, INIT)` je volán, kde `INIT` je hrana do počátečního uzlu struktury *GADDAG*. Procedura `Gen` je nezávislá na směru generování slova, pokládá písmeno na dané pole bez ohledu na to, zda se právě generuje levým nebo pravým směrem. V proceduře `Pokračuj` (obr. 9) daný směr rozhoduje na kterou stranu slova se má nové písmeno připojit. Směr je měněn vždy jednou, vždy zleva doprava, poté co narazíme na □. Viz struktura *GADDAG* (obr. 3).

```
function Gen( pozice, slovo, zasobnik, hrana)
   if pismeno L se jiz nachazi na aktualnim poli
      Pokracuj( pozice, L, slovo, zasobnik, DalsiHrana( hrana, L), hrana)
   endif
   elseif nejaka pismena zustavaji v zasobniku
      for kazde pismeno L v zasobniku, které je povoleno na aktualnim poli
         Pokracuj( pozice, L, slovo, zasobnik – L, DalsiHrana(hrana, L), hrana)
      endfor
      if zasobnik obsahuje BLANK
         for kazde pismeno L které je povoleno na aktualnim poli
            Pokracuj( pozice, L, slovo, zasobnik – BLANK, DalsiHrana( hrana, L), hrana)
         endfor
      endif
   endelse
end
```

Obrázek 8: Algoritmus generování slova pro GADDAG

```
function Pokracuj( pozice, L, slovo, zasobnik, NovaHrana, StaraHrana)
   if pozice ≤ 0
      slovo = L + slovo
      if L nad StaraHrana and zadne další pismeno vlevo
         PridejDoSeznamuSlov()
      endif
      if NovaHrana ≠ 0
         if vlevo je misto
            Gen( pozice – 1, slovo, zasobnik, NovaHrana)
         endif
         NovaHrana = DalsiHrana( NovaHrana, □)
         if NovaHrana ≠ 0 and zadne pismeno vlevo and místo napravo
            Gen( 1, slovo, zasobnik, NovaHrana)
         endif
      endif
   elseif pos > 0
      slovo = slovo + L
      if L nad StaraHrana and zadne další pismeno vpravo
         PridejDoSeznamuSlov()
      endif
      if NovaHrana ≠ 0 and napravo je místo
         Gen( pozice + 1, slovo, zasobnik, NovaHrana)
      endif
   endelse
end
```

Obrázek 9: Druhá část algoritmu generování slova pro GADDAG

5 Teorie Her

Teorie her a maticové hry jsou součástí množiny úloh, které slouží k řešení konfliktních situací. Jedná se o situace, kdy okolí může na naše rozhodnutí nějakým způsobem zareagovat, udělat nějaké protiopatření a tím zcela změnit naše postavení. Samotné maticové hry se zabývají hrou mezi dvěma protihráči a snaží se pro ně najít co nejlepší herní strategii, druhý hráč je automaticky považován za oponenta. Tento oponent může mít různé vlastnosti, od těchto vlastností je pak odvozena celá škála maticových her.

5.1 Maticové hry

Maticové hry patří do skupiny her, kde proti sobě hrají dva protihráči. Obvykle považujeme sami sebe za jednoho z hráčů, zatímco hráče druhého považujeme za našeho protivníka. Cílem je pak prosadit naši strategii na úkor protivníka tak, abychom ze hry vytěžili maximum. Obvykle jde o hry, kdy se hráči postupně střídají ve svých tazích (tzv. půltazích). Zatímco první hráč volí útočnou strategii, jeho oponent mu na ni odpovídá defenzivní strategií. Výsledkem je pro oba hráče určitý počet bodů. K přehlednému zapisování takovéhoto zisku slouží tzv. výplatní matice. Pro prvního hráče budeme takovou matici značit *M*, pro oponenta *N*.

$$M = \begin{pmatrix} m_{11} & m_{12} & \dots \\ m_{21} & m_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}, \qquad N = \begin{pmatrix} n_{11} & n_{12} & \dots \\ n_{21} & n_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

Obrázek 10: Výplatní matice pro hráče a jeho oponenta

Výplatní matice *M*, *N* jsou řádu *m* x *n*. Řádky matice znamenají strategie daného hráče, zatímco sloupce charakterizují strategie oponenta. Řádky *m* a *n* jsou obecně dvě rozdílná čísla, protože počty strategií jednotlivých hráčů se mohou velmi lišit. Prvky matice potom znamenají výdělek daného hráče, tedy hodnota prvku $m_{i,j}$ znamená výdělek hráče pokud zvolil strategii *i* a protivník mu odpověděl strategií *j*. Obdobně u matice *N* prvek $n_{i,j}$ značí výdělek druhého hráče pokud použil strategii *j*, proti strategii *i* prvního hráče.

Definice: Nechť *M*, *N* jsou výplatní matice $M, N \in \mathrm{Mat}_{mn}(\mathrm{T})$, pro které platí

$$\sum_{i=1}^{m}\sum_{j=1}^{n} m_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} n_{ij},$$

potom říkáme, že maticová hra je spravedlivá, v opačném případě říkáme, že je nespravedlivá.

Z praktického hlediska většinou transponujeme obě matice do jedné. Výsledná matice se vztahuje k některému z hráčů. Podle níže uvedené definice potom matice A představuje pro prvního hráče matici výher, zatímco pro druhého hráče představuje matici proher.

Definice: Nechť M, N jsou výplatní matice $M, N \in \mathrm{Mat}_{mn}(\mathrm{T})$, potom matice A, B řádu m x n definované vztahy

$$A \stackrel{\text{def}}{=} M - N,$$
$$B \stackrel{\text{def}}{=} N - M,$$

představují výplatní matice v maticové hře vztažené k jednotlivým hráčům.

Z výše uvedených údajů vyplývá, že hráči vstupují do hry s určitými strategiemi. Strategie prvního hráče (řádky matice) budeme značit S_i, zatímco strategii oponenta (sloupce matice) budeme značit S_j. Jednotlivé strategie můžeme mezi sebou porovnávat a úlohu poté optimalizovat.

Definice: Nechť $A \in \mathrm{Mat}_{mn}(\mathrm{T})$ je výplatní matice a nechť S_{i1} a S_{i2} jsou dvě různé strategie prvního hráče, potom pokud platí

- $\forall$ j = 1, 2, . . ., n: $a_{i1j} > a_{i2j}$, říkáme, že strategie S_{i1j} dominuje strategii S_{i2j}, naopak strategie S_{i2j} je dominována strategií S_{i1j}.
- $\forall$ j = 1, 2, . . ., n: $a_{i1j} \geq a_{i2j}$, za předpokladu, že existuje alespoň jedno j, pro nějž je nerovnost ostrá, říkáme, že strategie S_{i1j} je lepší strategií než S_{i2j}, naopak strategie S_{i2j} je horší strategií než S_{i1j}.

5.2 Výběry strategií

Vhodný výběr strategie je velice důležitý, protože se od něj odvíjí další osud naší hry. Chceme nalézt strategii, která je pro nás nejvýhodnější, zároveň předpokládáme, že hrajeme proti inteligentnímu protivníkovi, který se nám bude snažit hru co nejvíce znepříjemnit.

5.2.1 John von Neumanova teorie

Výběrem optimální strategie se zabývá John von Neumanova teorie. Jde o jednoduchou myšlenku, která vychází z představy že se ve výplatní matici (kapitola 5.1) nachází pole a_{ij}, které je z našeho pohledu nejvýhodnější. Předpokládejme, že proti nám stojí protihráč, který nám na každou naši strategii odpoví co nejúčinnější protistrategií. Pokud bychom ke každé naší strategii spočítali nejúčinnější soupeřovu protistrategii (z našeho pohledu minimum) a z takto vybraných hodnot potom maximum, dostali bychom strategii, která je z našeho pohledu klíčová.

Z výše uvedené John von Neumanovy teorie vychází teorie *minimaxu* a *maximinu*.

Definice: Nechť x je číslo definované vztahem

$$x \stackrel{\text{def}}{=} \operatorname{Max}_{i=1}^{m} \min_{j=1}^{n} a_{ij},$$

potom *x* označuje maximální hodnotu ze všech minimálních výplat pro každou strategii prvního hráče S_i. Nechť toto maximum z minim náleží strategii S_{i0}, potom S_{i0} je hledaná strategie pro prvního hráče. Takovýto výběr strategie označujeme jako *maximin*.

Definice: Nechť y je číslo definované vztahem

$$y \stackrel{\text{def}}{=} \min_{i=1}^{m} \operatorname{Max}_{j=1}^{n} a_{ij},$$

potom *x* označuje minimální hodnotu ze všech maximálních výplat pro každou strategii prvního hráče S_i. Nechť toto minimum z maxim náleží strategii S_{i0}, potom S_{i0} je hledaná strategie pro druhého hráče. Takovýto výběr strategie označujeme jako *minimax*.

Zajímavé teorie podobné *maximinu* a *minimaxu* jsou také *maximax* a *minimin*.

Pro *maximax* platí, že na každou z našich strategií, nám protivník odpoví z jeho pohledu nejhorším možným protitahem (už zde cítíme jistý náznak absurdnosti). Tímto způsobem hraní se dá zjistit náš nejvyšší možný výnos ze hry.

Opačnou teorii pak nazýváme *minimin*. Jde o způsob vedení hry tak, že pokaždé vybereme pro nás nejhorší možnou strategii. Soupeř pak reaguje z jeho pohledu nejlepším možným protitahem. Teorie *miniminu* vede k naší nejrychlejší (nebo nejvyšší) prohře.

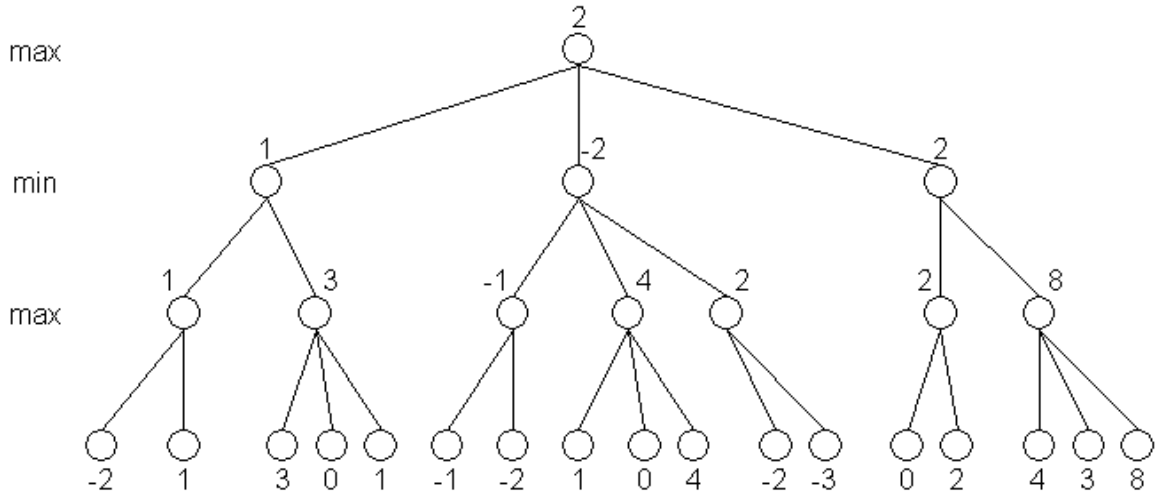
5.2.2 Strategie ve stromech

Dalším zajímavým problémem jsou hry, kde se hráči střídají v tazích (tzv. půltazích). Tahy se většinou opakují až do chvíle, kdy jeden z hráčů prohraje. Typickými představiteli tohoto druhu her jsou šachy, dáma nebo piškvorky. Celá problematika je snadno modelovatelná pomocí nelineárních datových struktur – stromů. Procházení těchto struktur vychází z teorie *maximinu* a *minimaxu*.

Strom nám vlastně představuje všechny možné případy, jak se hra může vyvíjet z aktuálního stavu. Pro nalezení optimálního tahu musíme všechny tyto případy projít až do zadané hloubky stromu, kde se nachází terminální uzly. Pomocí jistých kritérií a ohodnocovacích funkcí pak vyhodnotíme stav hry v terminálních uzlech a přeneseme zpátky do kořene stromu.

5.2.3 Použití minimaxu pro průchod stromem

Minimax algoritmus (název je trochu zavádějící, protože se algoritmus využívá většinou na základě teorie *maximinu* a ne *minimaxu*, viz. část 5.2.1) dělí stavový prostor do *max* a *min* úrovní. Na každé *max* úrovni vybere první hráč tah s maximálním užitkem, zatímco na každé *min* úrovni vybere protihráč tah s minimálním užitkem pro prvního hráče. Celý algoritmus přehledně zobrazuje následující obrázek.



Obrázek 11: Ukázka použití minimaxu pro průchod stromem

Heuristická funkce přidělí terminálním uzlům ohodnocení, to poté postupuje („probublává“) směrem ke kořenu následujícím způsobem:

- V uzlech na úrovni *max*, tj. uzlech prvního hráče bude ohodnocení rovno maximu ze všech ohodnocení následujících uzlů:

$$m_j \stackrel{\text{def}}{=} \operatorname*{Max}_{\forall i | i \succ j} m_i.$$

- V uzlech na úrovni *min*, tj. uzlech druhého hráče bude ohodnocení rovno minimu ze všech ohodnocení následujících uzlů:

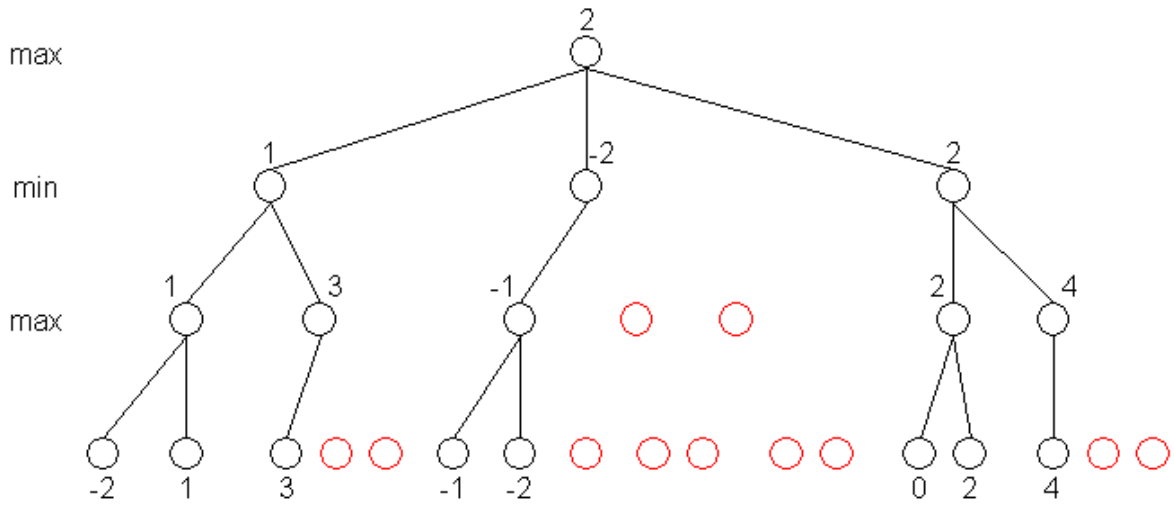
$$m_j \stackrel{\text{def}}{=} \min_{\forall i | i \succ j} m_i.$$

5.2.4 α-β prořezávání stromu

Při procházení stromové struktury je často kladen důraz na časovou složitost, ta je ovlivněna především průměrným větvením jednotlivých uzlů a maximální hloubkou prohledávání. Jedním

z možných vylepšení je neprocházení míst, která procházet nemusíme. Tuto optimalizaci provádíme tzv. *α-β* prořezáváním stromu (někdy označované jako *α-β* prohledávání).

Metoda je založena na myšlence neprohledávání stavů, které výběr vhodného tahu nijak nemůžou ovlivnit. Touto optimalizací lze docílit toho, že nám k nalezení optimálního tahu stačí projít pouze $\sqrt{m}$ stavů z celkových *m* uzlů (v nejlepším případě). Pro každý uzel si budeme pamatovat interval <α, β>, který značí nejlepší a nejhorší ohodnocení jakých může uzel nabývat (α-nejhorší, β-nejlepší).



Obrázek 12: α-β prořezávání

Na obrázku zobrazujícím α-β prořezávání jsou červeně označený uzly, které se v rámci optimalizace neprohledávají. Ať už nabývají jakýkoliv hodnot, nemají vliv na výběr optimální strategie.

6 Herní strategie

Pokud se chceme věnovat modelování inteligentního protihráče, je nutné si uvědomit, jak vůbec funguje lidská strategie [She02].

Taktika hráčů zahrnuje většinou tyto 4 body
- hledání 50 bodové prémie
- hledání bonusových míst
- vylepšování nalezených tahů
- uvažování o zůstatku na zásobníku

Jako první bychom se vždy měli snažit nalézt uplatnění 50 bodové prémie. Tzn. že hledáme buď 7-písmenné slovo s využitím celého zásobníku nebo 8-písmenné slovo, kde bereme v úvahu i některý kámen z hrací plochy, který byl odehrán již dříve. Profesionální hráči si z písmen na zásobníku vytvářejí předpony a přípony a zbytek písmen se snaží ve své mysli co nejvíce přeskupovat. Například v anglickém jazyce, 200 nejpoužívanějších předpon a přípon pokrývá 80% všech sedmi a osmi-písmenných slov.

Za druhé bychom se měli soustředit na prémiová místa, taková místa obsahují některé z prémiových polí (existují 4 druhy takových polí, dvojnásobek a trojnásobek hodnoty písmena položeného na takovém poli, resp. dvojnásobek a trojnásobek hodnoty celého slova procházejícího přes takové pole).

Jestliže jsme již našli nějaký výhodný tah, začneme se sami sebe ptát jak zahrát tah ještě lepší. Pokud jsme tedy například našli tah ohodnocený 30 body, zkoumáme jak zahrát ještě více a pomyslně hrací desku „ořezáváme“ o místa kde vyšší skóre zahrát nelze.

Nakonec musíme brát v úvahu i dopad písmen, které nám zůstanou na zásobníku, na budoucí tahy. Dobrá kombinace souhlásek a samohlásek, spolu se zbavováním se opakovaných písmen tvoří vyvážený zásobník, se kterým se nám bude v budoucnu lépe tvořit 50 bodová prémie.

Profesionální hráči dále zavádí do hry defenzivní uvažování. Žádný hráč nechce zahrát slovo tak, že tím vytvoří příležitost pro protihráče zahrát vysoké skóre. Dále pokud je aktuální stav hrací desky spolu s protihráčovým zásobníkem takový, že by mohl soupeř zahrát příští tah vysoce ohodnocen, můžeme se pokusit zablokovat takovou příležitost. Přestože nám takový blokující tah nepřinese tolik bodů jako jiné alternativy, může to být tah nejvýhodnější. Podobně se hráč chová i v případě, že si vytvoří dostatečný bodový náskok. V jeho zájmu je dále držet hru „uzavřenou“ tím způsobem, že se snaží blokovat místa na desce, kde by se daly zahrát vysoce ohodnocené tahy.

Logicky lze průběh hry rozdělit do 3 částí (viz. [She02] nebo [Sha01]), počáteční fázi (v cizí literatuře označovanou buď *early game* nebo *middle game*), fázi před koncí hry (*pre-end game*) a konečnou

fázi (*end-game*). Každá z fází vybírá nejvýhodnější tah podle jiných kritérií. Podrobněji jsou jednotlivé fáze probrány níže.

6.1 Počáteční fáze

Počáteční fáze začíná v prvním tahu a pokračuje až do doby, než v hracím sáčku zůstává okolo 7 hracích kamenů. Během celé počáteční fáze je SCRABBLE hrou neúplné informace, nevíme jaké kameny má náš soupeř na zásobníku a nemůže tedy ani předvídat jeho reakce. Strategie použitelné v počáteční fázi hry si představíme v následujících částech.

6.1.1 Hladová strategie

Program založen na této strategii jednoduše považuje nejlépe ohodnocený tah za tah nejlepší. Jelikož má na své straně výhodu znalosti celého slovníku a tudíž i všech možných tahů dosažitelných s aktuálním zásobníkem, nebude mu dělat problém porazit většinu hráčů SCRABBLE. Proti zkušeným profesionálním hráčům už si ale tak dobře nepovede, to je dáno především absencí jakékoliv inteligentnější taktiky. Hladová strategie často odehrává kameny jako „žolík" nebo „A", i když mnohem efektivnější by bylo zahrát méně ohodnocený tah a tyto cenné kameny si nechat. Opačným problémem je potom ponechání si písmen jako „X" nebo „Ť", když mnohem výhodnější by bylo zahrát o něco méně ohodnocený tah a těchto špatně použitelných písmen se zbavit.

6.1.2 Zásobníková heuristika

Zavádí do hry ohodnocení jednotlivých hracích kamenů v zásobníku podle jejich užitečnosti v budoucnosti. Pro ponechání si špatných písmen v zásobníku musíme zavést nějaký postih, který by donutil program těchto písmen se zbavit, a to i za cenu ztráty několika bodů. Rovnováha mezi současným a budoucím skórováním je ideálním řešením pro dosažení maxima bodů v průběhu celé hry. Tento způsob zavádí (na rozdíl od hladové strategie) i možnost výměny zásobníku, pokud by výměna zásobníku měla vyšší užitečnost než součet hodnoty odehraného slova a následného zásobníku. Přítomnost stejných kamenů na zásobníku snižuje počet výsledných kombinací a navíc snižuje pravděpodobnost odehrání všech sedmi kamenů a získání následného bonusu. Proto přítomnost stejných kamenů snižuje výslednou hodnotu zásobníku (zásobníková heuristika verze 2). Místo penále za trojnásobný a vícenásobný výskyt stejného písmene, opakovaně přičítáme hodnotu dvojnásobnou.

Hlavním problémem se zde stává nalezení vhodného způsobu ohodnocení zbytku zásobníku po odehrání tahu. Nechceme se zbavit špatných písmen za cenu příliš velké ztráty bodů.

Letter	Heuristic1	Heuristic2		Letter	Heuristic1	Heuristic2	
A	+0.5	+1.0	-3.0	B	-3.5	-3.5	-3.0
C	-0.5	-0.5	-3.5	D	-1.0	0.0	-2.5
E	+4.0	+4.0	-2.5	F	-3.0	-2.0	-2.0
G	-3.5	-2.0	-2.5	H	+0.5	+0.5	-3.5
I	-1.5	-0.5	-4.0	J	-2.5	-3.0	
K	-1.5	-2.5		L	-1.5	-1.0	-2.0
M	-0.5	-1.0	-2.0	N	0.0	+0.5	-2.5
O	-2.5	-1.5	-3.5	P	-1.5	-1.5	-2.5
Q	-11.5	-11.5		R	+1.0	+1.5	-3.5
S	+7.5	+7.5	-4.0	T	-1.0	0.0	-2.5
U	-4.5	-3.0	-3.0	V	-6.5	-5.5	-3.5
W	-4.0	-4.0	-4.5	X	+3.5	+3.5	
Y	-2.5	-2.0	-4.5	Z	+3.0	+2.0	
BLANK	+24.5	+24.5	-15.0				

Obrázek 13: Heuristika v.1 a Heuristika v.2 pro anglickou abecedu

Příklad výpočtu užitečnosti pro zásobník obsahující hrací kameny s písmeny I,I,I,S,S.

Heuristika v.1 = -1.5 + -1.5 + -1.5 + 7.5 + 7.5 = +10.5

Heuristika v.2 = -0.5 + (-0.5 - 4.0) + (-0.5 - 4.0 - 4.0) + 7.5 + (7.5 - 4.0) = -2.5

6.1.2.1 Kombinace souhlásek a samohlásek

Některé publikace [She01] uvádí, že je vhodné do ohodnocení zásobníku zahrnovat poměr mezi souhláskami a samohláskami. Např. u zůstatku „SDKTL“ vidíme, že není moc perspektivní. Pokud navíc vezmeme v úvahu, že si z hracího sáčku můžeme vylosovat další souhlásky, mohli bychom mít problémy s vytvořením jakéhokoliv slova. Ten samý problém nastává i v opačném případě, ponecháme-li si samé samohlásky.

6.1.3 Uzavřenost prvního tahu

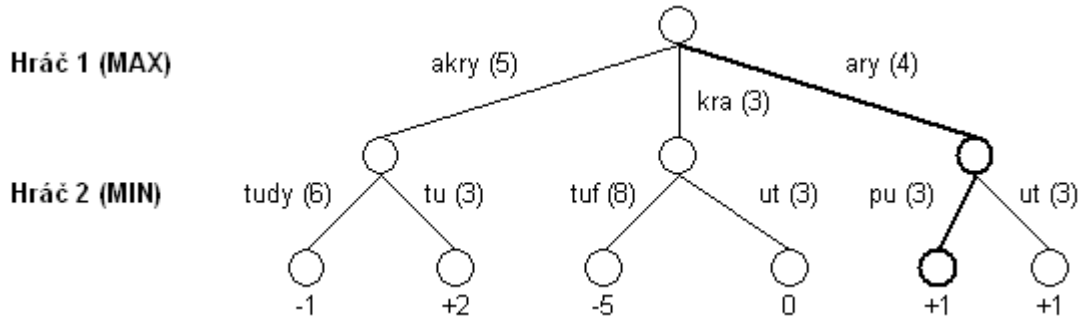
Tato drobná taktika je založená na tom, že je výhodné zahrát první tah hodně uzavřený, pouze s využitím několika málo písmen. Důvodem je znepříjemnění oponentova následujícího tahu, kdy oponent nemá mnoho možností kam svůj tah umístit. Navíc pokud zahrajeme tah s využitím pouze dvou nebo tří písmen, nemá oponent přístup k žádnému ze slovních bonusů.

6.1.4 Monte Carlo

K přizpůsobení metody Monte Carlo SCRABBLU [Wic06] musíme provádět nějaké odhady. Jestliže známe obsah našeho zásobníku a již odehrané kameny na hrací ploše, můžeme se pokusit odhadnout, co má nejspíše náš oponent na zásobníku. Čím více je odehráno kamenů, tím přesnější náš odhad bude. Teď tedy vezmeme 10 našich nejlepších tahů (nejlépe ty co mají nejvyšší ohodnocení) a provedeme 1000 odhadů oponentova zásobníku. Tím získáme 10,000 kombinací našeho tahu a soupeřova zásobníku a můžeme se pokusit generovat soupeřův nejlepší tah. Tímto způsobem můžeme vypočítat průměrnou hodnotu soupeřovy reakce na náš tah. A můžeme vybrat tah jež bude jak dostatečně ofenzivní, tak i defenzivní.

6.2 Konečná fáze

Na konci hry, kdy je sáček s hracími kameny zcela vyprázdněn, mění se SCRABBLE v hru kompletní informace. Známe-li přesné množství jednotlivých písmen v hracím sáčku, můžeme si v této fázi hry lehce spočítat jaké hrací kameny má soupeř na zásobníku. S těmito znalostmi můžeme snadno předvídat, jak soupeř pravděpodobně zareaguje na náš tah. Proto může program rozhodnout, zda je v dané situaci možno vyhrát, remizovat nebo zda nejspíš prohraje. To znamená, že známe kompletní stavový prostor hry, a můžeme si zvolit optimální strategii pomocí algoritmu *minimax* představeného v kapitole 5. Jako heuristické ohodnocení nám poslouží rozdíl skóre na konci hry. Postupně tedy vygenerujeme všechny možné tahy, které v dané situaci můžeme zahrát, ke každému našemu tahu vygenerujeme všechny možné reakce našeho oponenta a takto postup opakujeme dokud nenastane konec hry. V takovém případě vyhodnotíme rozdíl skóre, pomocí algoritmu *minimax* potom tyto hodnoty „probublávají“ zpátky ke kořenu, kde se rozhodneme který z našich tahů bude mít nejpříznivější vliv na budoucí vývoj hry. Na následujícím obrázku je uveden příklad jak by daný konec hry mohl vypadat, hrany uzlu obsahují slovo, které je daný tah zahrané a jeho bodovou hodnotu.



Obrázek 14: Příklad použití minimaxu pro konec hry

Z uvedeného tahu vychází že je pro nás nejvýhodnější zahrát slovo „ary“, i když není v ten moment nejlépe ohodnoceným tahem. Ať na něj náš soupeř zareaguje jakkoliv, nemůže již partii vyhrát.

6.3 Přizpůsobení hry slabším hráčům

Zatímco v předchozích částech jsme se věnovali především optimalizaci herní strategie a jednotlivým vylepšením herního programu, v této části si probereme přizpůsobení hry amatérům a slabším hráčů. Pravděpodobně by nás dlouho nebavilo hrát proti programu, který by nás pravidelně porážel o velké množství bodů. Proto se zamyslíme nad možným přizpůsobením generování tahů a výběru vhodného tahu pro takovéto jednodušší úrovně hry.

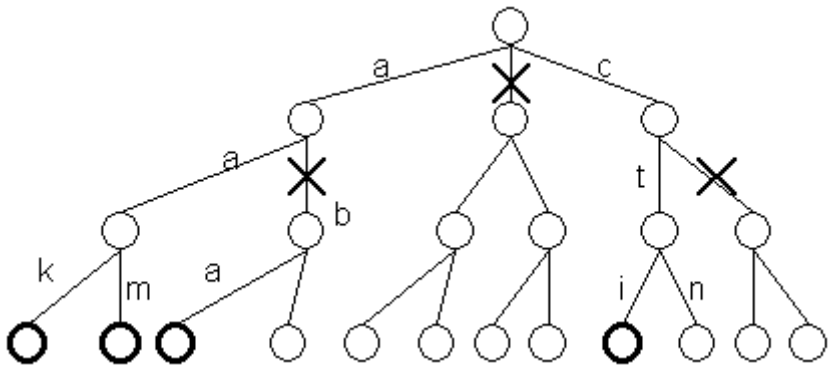
Pokud se zamyslíme nad jednotlivými fázemi tahu ve SCRABBLU, najdeme 2 možná místa pro omezení tahu. Jedním z nich je fáze generování tahů, druhým je výběr vhodného tahu ze všech vygenerovaných slov.

Omezení ve fázi generování tahu jsou tato:

- **Délka slova** – i když omezíme algoritmu maximální délku slova, program může s využitím bonusových polí a lépe hodnocených písmen získat poměrně velké množství bodů. Takové omezení navíc nemá velký vliv na konečnou fázi hry, kde pracujeme s menším počtem hracích kamenů a skládáme převážně kratší slova. Pokud omezíme délku slova na hodnotu menší než 7, zamezíme tím programu dosáhnutí *binga*.

- **Počet použitých kamenů ze zásobníku** – poměrně zrádné omezení, i s malým počtem písmen ze zásobníku může program, s pomocí kamenů již odehraných na hrací ploše, sestavovat dlouhá a dobře ohodnocená slova. Dosáhneme však toho, že program nebude schopen dosáhnout 50 bodové prémie za odehrání všech písmen ze zásobníku (*binga*).

- **Omezení průchodu strukturou slovníku** – nejpřirozenější omezení tvorby slov. Program při generování slov neprochází celou stromovou strukturu slovníku, ale pouze některé její podčásti. Simulujeme tak vlastně malou slovní zásobu. Stav přehledně zobrazuje následující obrázek. Velikost omezení závisí na vhodně zvolených parametrech. Jde o prořezávání stromové struktury.



Obrázek 15: Omezení průchodu grafem.

Omezení ve fázi výběru vhodného tahu jsou tato:

- **Počet bodů za tah –** jde o maximální počet bodů, které může program zahrát. Poměrně snadno a průhledně tak omezíme protihráče. Pokud známe náš průměrný bodový zisk, můžeme program nastavit na hru přesně pro naši úroveň.

- **Výběr náhodného tahu –** po vygenerování všech možných tahů a jejich následném ohodnocení, nevybereme nejlepší možný tah, ale výběr provedeme náhodně. Z dlouhodobého hlediska bude takový program odehrávat slova v hodnotě aritmetického průměru všech nalezených slov.

7 Implementace

Cílem praktické části bylo vytvořit funkční implementaci hry SCRABBLE ve formě webové aplikace. Aplikace slouží k hraní lidského hráče proti umělému protivníkovi, přičemž umělý protivník by měl zdatně konkurovat nejen amatérům, ale i hráčům profesionálním. Pro dosažení dané inteligence budou implementovány strategie zmíněné v předešlých částech práce. Hra je rozdělena do několika výkonnostních úrovní, pomocí kterých si člověk bude moci regulovat sílu protivníka. Hra je provázána s výkladovým slovníkem, objasňujícím význam některých slov.

Kromě samotného programu sloužícího k hraní SCRABBLE proti umělému oponentovi, vznikl i menší systém pro simulaci her mezi jednotlivými úrovněmi umělého protihráče. Systém byl navržen speciálně pro simulaci až několika tisíc her a následné vygenerování základních statistik, s jejichž pomocí by se dá posoudit inteligence protihráčů a jejich výkonnost. Oba programy se nachází na přiloženém CD.

7.1 Použité technologie

S ohledem na fakt, že aplikace má být webová a jedním z hlavních kritérií je rychlost, bylo rozhodnuto implementovat daný program pomocí jazyku Java. Jedná se o objektově orientovaný jazyk vyvinutý firmou Sun Microsystems. Java je jedním z nejpoužívanějších programovacích jazyků na světě a je vyvíjena jako „opensource“. Mezi její základní výhody patří:

- Jednoduchost – syntaxe je zjednodušenou verzí syntaxe jazyka C a C++.
- Objektově orientovaný – s výjimkou 8 základních datových typů jsou všechny datové typy objektové.
- Interpretovaný – místo skutečného zdrojového kódu se vytváří pouze tzv. mezikód, tento formát je nezávislý na architektuře počítače.
- Výkonnost – přestože se jedná o jazyk interpretovaný, není ztráta výkonu významná, neboť překladače fungují v režimu „just in time“ a do strojového kódu se překládá ten kód, který je opravdu potřebný.

Výše uvedené vlastnosti jazyku Java byly pouze stručným seznámením a představením, podrobné informace o historii a vlastnostech jazyku Java lze nalézt např. zde [Wik1].

Uživatelské rozhranní bylo implementováno ve formě appletu s využitím knihoven AWT (Abstrakt Window Toolkit). AWT umožňuje tvorbu grafického uživatelského prostředí (GUI), tj. práci se vstupy a výstupy v graficky orientovaných systémech. Takto vytvořené programy jsou plně

přenositelné, ale zároveň je ovládání jednotlivých komponent přizpůsobeno zvyklostem konkrétního systému, více [KT01].

Celý vývoj programu SCRABBLE probíhal ve vývojovém prostředí NetBeans IDE. Jedná se o nástroj umožňující psaní, překlad, ladění a šíření programu. Vývojové prostředí NetBeans je bezplatně šiřitelný produkt, který je možné používat bez jakýchkoliv omezení. Je použitelný na operačních systémech Windows, Linux, Mac OS a Solaris. Podrobnější informace o NetBeans [Wik2].

7.2 Lexikon

Základem každé implementace hry SCRABBLE je slovník. Ten slouží nejen ke kontrole slov vložených hráčem, ale v případě umělého protivníka i k samotnému generování tahů. Slovník musí být dostatečně obsáhlý, aby pokryl slovní zásobu hrajících lidí, na druhé straně aby podle něj mohl umělý hráč vygenerovat dostatečné množství slov, z kterých potom vybere to nejvýhodnější.

7.2.1 Použitý slovník

V české republice existují 4 oficiálně uznávané slovníky. Všechny mohou sloužit pro posuzování slov na soutěžích, pořádaných pod hlavičkou České asociace SCRABBLE. Slovníky obsahují slova do maximální velikosti 9 písmen. Jelikož je ve hře SCRABBLE možno skládat až 15 písmenná slova, jsou všechna slova delší než 9 písmen posuzována podle pramenů z nichž dané slovníky vznikají (příloha č. 3).

Dané slovníky jsou tyto:

- **Blex-Klasik** - obsahuje slova o délce 2-5 písmen, celkem obsahuje 34263 slov.
- **Blex-Plus** - nejnovější tj. 3. rozšířené vydání (2007) obsahuje slova o délce 2-5 písmen, celkový počet slov je 55378.
- **Novex-Klasik**- obsahuje slova v rozmezí 2-9 písmen, celkem obsahuje přibližně 768 tisíc slov.
- **Novex-Plus** - obsahuje slova v rozmezí 2-9 písmen, na rozdíl od verze Klasik obsahuje slovesa včetně jejich přechodníků. Celkový počet slov je cca 1 milión 217 tisíc.

Námi používaný slovník odpovídá starší verzi Novex-Klasik, přičemž obsahuje i část slov delších než je 9 písmen. Slovník je uložen v textovém souboru, jednotlivá slova jsou oddělena novým řádkem, celková velikost souboru je cca 7,6M.

Všechna slova uvedená ve slovníku vyhovují pravidlům přípustnosti slov. Podrobnější informace viz. příloha č. 3. Ve výše uvedené příloze jsou také uvedeny všechny prameny, z nichž dané slovníky vznikají.

7.2.2 Reprezentace slovníku

K uložení lexikonu jsme použili *trii*, představenou v části 3.1.3. Důvodem byla především jednoduchost implementace kombinovaná s dostatečnou rychlostí vyhledávání daných slov i vhodnost celé struktury k pozdějšímu generování tahů pro umělého protihráče. Jelikož je slovník uložen ve formě textového souboru, je pro nás velice snadné vytvořit stromovou strukturu, u které jednotlivá slova sdílejí společné předpony.

Základem stromu je reprezentace uzlu, pro něj byla použita následující třída:

```
public class cNode
{
   boolean terminal;
   char    ch;
   cNode   child;
   cNode   next;
}
```

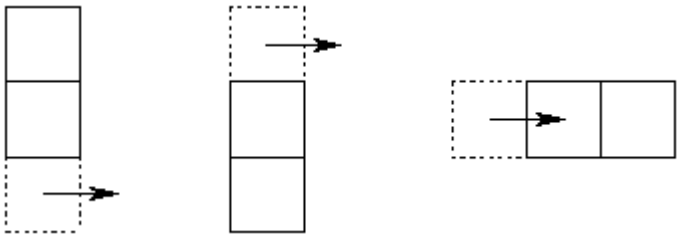
Vlastnost `terminal` nám značí konec slova, pokud je hodnota této vlastnosti nastavena na `true`, získáme při průchodu stromem od kořene k aktuálnímu uzlu platné slovo. Vlastnost `ch` obsahuje znak slova reprezentovaný uzlem, `next` potom sousední uzel na stejné úrovní stromu a `child` uzel synovský.

7.3 Generátor tahů

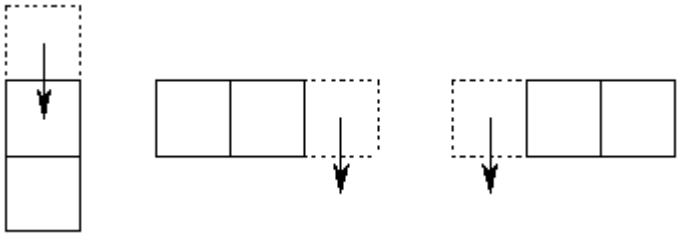
Základním kamenem hry SCRABBLE obsahující umělého oponenta je generátor tahů. Cíl generátoru je v daném kontextu hrací plochy, soupeřova zásobníku a aktuální fáze hry vygenerovat všechny možné tahy, z daných tahů vybrat ten nejvýhodnější a poté jej provést. Vše musí proběhnout dostatečně rychle, aby lidský hráč nemusel dlouho čekat.

7.3.1 Vyhledání kotev

Před samotným generováním slov potřebujeme najít místa, na kterých se tahy mohou nacházet. Ve hře SCRABBLE jsou takovými místy *kotvy* (představeny v 4.1.2). Jelikož lze slova generovat dvěma směry, byly i *kotvy* rozděleny na dva typy, vertikální (z kterých lze generovat slova pouze vertikálním směrem) a horizontální (pro horizontální tahy). Každý z těchto typů navíc umožňuje generování nového tahu jiným způsobem. Tyto způsoby jsou znázorněné na následujícím obrázku .



Obrázek 16: Horizontální kotvy



Obrázek 17: Vertikální kotvy

Kotvy jsou implementovány pomocí třídy `cAnchor`, jejíž strukturu vidíme níže.

```
public class cAnchor
{
   cCoordinate position;
   boolean     isVertical;
   int         limit;
}
```

Vlastnost `position` nám udává pozici na hrací ploše, `isVertical` slouží k rozlišení vertikálních a horizontálních kotev. Limit nám ukazuje, kolik hracích polí je volných před danou *kotvou* (podrobněji jsme se s touto vlastností seznámili v části 4.2), tato vlastnost je potřebná pro generování předpon slova, kdy slouží jako omezení délky předpony. Po každém provedeném tahu tedy procházíme hrací plochu, hledáme všechny *kotvy* a počítáme pro ně počet volných míst nacházejících se před *kotvou*.

7.3.2 Generování možných tahů

Po získání všech *kotev* pro aktuální stav hrací plochy, následuje generování všech možných tahů pro takovou *kotvu*.

V části 4.2 jsme si představili hlavní dva algoritmy použitelné pro generování, jedná se o DAWG a GADDAG. Ačkoliv z dostupné literatury [Gor94] vyplývá, že algoritmus GADDAG provádí generování přibližně 2x rychleji, přesto byl použit algoritmus DAWG, především z důvodu až 5x úspornějšího uložení slovníku. Stejný algoritmus používá převážná většina dnešních programů. S využitím tohoto algoritmu trvá samotné generování tahů průměrně 15ms. V nejhorším případě (v situaci, kdy máme na zásobníku oba „žolíky“) je to potom 400ms.

Po vygenerování všech tahů jsou tyto tahy bodově ohodnoceny, základ tvoří body za písmena, z kterých je slovo složeno. Poté jsou započteny prémiová pole, na kterých se dané slovo nachází (princip bodování kapitola 2.2.4) a následně je přičtena 50 bodová prémie (v případě, že jsme použili všechny kameny ze zásobníku).

7.3.3 Výběr vhodného tahu

Po vygenerování všech dostupných slov a jejich následném ohodnocení se nacházíme ve fázi, kdy je potřeba rozhodnout, který z tahů vyhodnotíme jako nejvýhodnější. Pro výběr tahu byly implementovány 3 základní mechanismy. Podle stádia hry a zvolené úrovně programu je jeden z mechanismů zvolen.

7.3.3.1 Hladová strategie

V počáteční fázi hry, to je od prvního tahu až po odebrání všech písmen z hracího sáčku, byla implementována jako výběr nejvýhodnějšího tahu hladová strategie (podrobněji 6.1.1). V kombinaci s vhodným slovníkem a generováním všech existujících tahů stačí tento způsob k poražení většiny amatérských hráčů, přesto je ve vyšších úrovních hry používána následující strategie.

7.3.3.2 Zásobníková heuristika

V části 6.1.2 byla vysvětlena strategie, která jako nejvýhodnější nevybírá nejlépe ohodnocený tah, ale zvažuje vybírání tahu v širších souvislostech. Konkrétně šlo o zásobníkovou heuristiku a dopad zbytku písmen v zásobníku na budoucí vývoj hry. V té samé kapitole byl ukázán příklad takové heuristiky pro anglický oficiálně používaný slovník a anglickou verzi SCRABBLE.

Podobná heuristika není pro českou verzi SCRABBLE dostupná, proto se pokusíme vytvořit vlastní heuristiku a v části testování porovnat dosažené výsledky s hladovou strategií.

Pokud se podíváme na tabulku v části 6.1.2, zjistíme, že až na některé výjimky se bodová ohodnocení písmen pohybují v rozmezí +4 až -4 bodů. K výjimkám patří „žolík“, ohodnocen +24,5 body. Hodnota +24,5 nám naznačuje cenu tohoto kamene, z ní vychází, že by se měl používat pro pokládání *binga* a nebo výjimečně vysoko ohodnocených tahů). Jelikož v české verzi SCRABBLE plní „žolík“

stejnou funkci, převezmeme také tuto hodnotu. Další výjimkou je písmeno „Q“ (-11,5), to v české verzi neexistuje, proto jej nemusíme uvažovat. Písmeno „S“ (+7,5) vytváří v anglickém jazyce množné tvary podstatných jmen, proto je velmi dobře ceněno. V českém jazyce toto neplatí, proto jej nebudeme nijak zvýhodňovat. Posledními dvěma písmeny vymykajícími se z rozsahu <-4, +4> jsou „U“ (hodnota -4,5) a „V“ (-6,5), která jsou v anglické verzi těžce použitelná, my je nebudeme považovat za ničím výjimečné a přiřadíme je ke zbytku.

Budeme vycházet z četnosti jednotlivých písmen ve slovech slovníku, následující tabulka zobrazuje počet slov v kterých se dané písmeno vyskytuje.

písmeno	počet	počet(%)	písmeno	počet	počet(%)
n	524560	74,1	h	110813	15,7
e	509064	71,9	ř	107681	15,2
o	423835	59,8	š	87100	12,3
a	341040	48,2	ě	85492	12,1
l	289692	40,9	b	82528	11,7
t	281439	39,8	c	68480	9,7
v	273675	38,7	č	58200	8,2
p	259861	36,7	ý	48054	6,8
i	230624	32,6	é	43348	6,1
r	220499	31,2	ž	39100	5,6
u	214801	30,3	f	18562	2,6
z	194522	27,5	g	9493	1,3
á	187183	26,4	ň	6391	0,9
d	175305	24,8	ť	3988	0,6
k	167963	23,7	ů	3633	0,5
s	167155	23,6	ú	3008	0,4
í	159536	22,5	ď	2650	0,4
m	150519	21,3	x	2094	0,3
j	135175	19,1	ó	1327	0,2
y	132650	18,7			

Obrázek 18: Četnost výskytu písmen ve slovech slovníku

Podle počtu výskytů pak těmto písmenům přidělíme hodnotu z intervalu <-4, 4>, tím dostaneme následující heuristické ohodnocení kamenů v zásobníku.

písmeno	ohodnocení	písmeno	ohodnocení	písmeno	ohodnocení
a	+1.5	i	-0.5	s	-1.5
á	-1	í	-1.5	š	-2.5
b	-2.5	j	-2.0	t	+0.5
c	-3.0	k	-1.5	ť	-4.0
č	-3.0	l	+0.5	u	-0.5
d	-1.5	m	-1.5	ú	-4.0
ď	-4.0	n	+4.0	ů	-4.0
e	+3.5	ň	-4.0	v	-4.0
é	-3.5	o	+2.5	x	-4.0
ě	-2.5	ó	-4.0	y	-2.0
f	-3.5	p	+0.0	ý	-3.5
g	-4.0	r	-0.5	z	-1.0
h	-2.0	ř	-2.5	ž	-3.5
žolík	+24.5				

Obrázek 19: námi vytvořená heuristická tabulka pro českou abecedu

Příklad použití:

Předpokládejme že máme v úvodním tahu hry na zásobníku písmena S, A, N, E, Ó, Č, J. Nejlépe ohodnocený tah, který můžeme složit je v tomto případě „NASEČ“ za 22 bodů, na zásobníku nám v tomto případě zůstane Ó, J, ohodnoceno pomocí naší heuristiky na -6 bodů (-2,0 – 4,0). Celková užitečnost tohoto tahu tedy bude 16 bodů (22 - 6). Můžeme ale také zahrát slovo „JÓ“ s bodovým ziskem 18 bodů, přičemž nám na zásobníku zůstane S, A, N, Č, E, tento zůstatek bude ohodnocen pomocí heuristiky na +4,5 bodů (-1,5 + 1,5 + 4,0 + 3,5 – 3,0). Celkový užitek bude vyhodnocen na 22,5 bodů. Vidíme, že z hlediska dlouhodobějšího užitku pro nás bude výhodnější druhý možný tah.

7.3.3.3 Minimax

Pro řešení konce, kdy hrací sáček neobsahuje žádné kamenný a hra se mění v hru kompletní informace, byl implementován výběr tahu pomocí algoritmu *minimax* (detailně v částech 5.2.3 a 6.3). Pro úsporu prostředků byl algoritmus implementován s jistým omezením prohledávání stavového prostoru.

Minimax bez omezení by měl fungovat způsobem uvedeným v části 6.3. Považujme se za prvního hráče, pak bychom měli vygenerovat všechny pro nás dostupné tahy. Pro každý z našich tahů vygenerovat všechny protitahy soupeře, pro každý protitah opět generovat naše možné tahy. Generování by proběhlo rekurzivně až do konce hry, došlo by k vyhodnocení skóre a hodnota by putovala opět ke kořenu. Hloubka rekurze v nejhorším případě je 14 (každý tah by jednotlivý hráči odehráli jeden kámen, každý jich může mít až 7). Průměrně je každé kolo generováno okolo 100 možných tahů, tzn. že průměrné větvení je okolo 100 (v případě heuristické strategie až 250), i když s koncem hry takovéto větvení klesá, mohou se vyskytovat případy, kdy dojde k obrovskému nárůstu stavů a prohledávání bude trvat neúměrně dlouho (speciálně pokud se na zásobníku vyskytuje „žolík“ a počet generovaných tahů může přesahovat 1000 možností). Jako optimalizaci tedy omezíme větvení *minimaxového* stromu. Po vygenerování všech tahů (na *max* i na *min* úrovni), tahy seřadíme podle

hodnocení, vezmeme 10 nejlepších a s těmi pracujeme podle výše zmíněných pravidel. Tím omezíme větvení stromu na přijatelnou hodnotu.
Protože neřadíme velké množství dat, byl pro jednoduchost zvolen algoritmus *Bubblesort*.

Zatímco u hladové strategie trvá průměrný tah okolo 20ms, s využitím heuristického ohodnocení 60 ms (bráno pro hardwarové parametry uvedené v části 7.1), zavedením řešení konce hry (s naším omezením) stoupne průměrná hodnota generování na stále přijatelných 7700ms.

7.3.4 Provedení tahu

Jakmile vybereme vhodný tah, je nutné dané slovo umístit na hrací plochu a upravit místa hrací plochy, kterých se dané slovo dotýká.

Hrací deska byla implementována jako dvourozměrné pole `cSquare` (třída modelující herní pole).

```
public class cSquare
{
   ArrayList<Character> verticalCrosschecks;
   ArrayList<Character> horizontalCrosschecks;

   int         bonus;
   char        tile;
   boolean     isJoker;
   cCoordinate position;
}
```

Vlastnosti `verticalCrosschecks` a `horizontalCrosschecks` nám představují *crossmnožiny* daného herního políčka (viz. 4.1.1), pro jednodušší práci jsme je rozdělili na horizontální a vertikální. Po každém tahu, zasahujícím do sousední oblasti pole, je nutné tuto množinu aktualizovat. Tile značí hodnotu kamene, jenž je na hracím poli položen, pakliže je pole prázdné, obsahuje hodnotu '-'. Vlastnost `isJoker` nám ukazuje, zda se na aktuálním místě nachází klasický kámen nebo žolík. `Position` potom obsahuje souřadnice pole na hrací ploše. Vlastností bonus značíme, zda je aktuální pole prémiové. Po provedení tahu zasahujícího do prémiového pole se daný bonus ruší (tzn. že může být započítán pouze jednou).

7.4 Stupně obtížnosti

V rámci vytvoření několika rozdílných herní úrovní bylo implementováno 6 rozdílných stupňů obtížnosti oponenta. Základní úrovní, od které se ostatní odvíjí, je úroveň „profesional" (hladová

strategie). Vytvořeny byly 3 nižší obtížnosti, každá využívá jiné omezení z části 6.4, a 2 vyšší obtížnosti, které využívají vylepšení uvedené detailně v 7.3.3.

- **Amateur** – Nejjednodušší z obtížností hry, daná úroveň protihráče má implementováno omezení v podobě maximálního počtu bodů za tah. Po sérii testů byl daný počet bodů ustálen na 10. Maximální možný tah zahraný tímto protihráčem je tak desetibodový (včetně započítaných prémiových polí).

- **Adept** – Úroveň hry odvozená od úrovně *profesional* používající omezení průchodu datovou strukturou slovníku (podrobně popsáno v 6.4). Úroveň simuluje menší slovní zásobu programu, protože se při průchodu datovou strukturou stromu nedostává do všech podčástí (na každé úrovni stromu „prořezává“ třetinu dané datové struktury). Dosahuje průměrného počtu 12 bodů za tah.

- **Experienced** – Tento algoritmus má implementováno omezení v podobě délky slova, které může za tah položit. Poměrně zrádná strategie, používáním měně kamenů sice nikdy nedosáhne 50 bodové prémie ani těch nejvýhodnějších tahů, na druhou stranu hraním kratších slov hraje defensivním stylem a redukuje počet míst, kam může protihráč umístit své kameny. Úroveň *experienced* používá maximálně 4 kameny ze zásobníku, čímž dosahuje průměrně 16 bodů za tah.

- **Profesional** – Základní algoritmus generování tahu a výběru vhodného tahu pomocí hladové strategie. Tato úroveň hry nepoužívá žádné z dostupných omezení, naopak ani žádné z dostupných vylepšení. Dosahuje průměrného počtu bodů za tah okolo 19 bodů.

- **Inhuman** – Oponent hrající na úrovni *inhuman* generuje všechny možné tahy dostupné pro danou plochu, na rozdíl od úrovně *profesional* však nevybírá tah nejlépe ohodnocený, ale bere v úvahu i užitečnost zbylých kamenů na zásobníku pro budoucí tvorbu slov. Dosahuje až 20 bodů za tah.

- **Godlike** – Nejvyšší úroveň oponenta dostupná v dané hře, je odvozená z obtížnosti *inhuman*, přidáno je pak řešení konce hry algoritmem *minimax* (detailní informace v části 7.3.3). V praxi to znamená, že pokud hrajete s touto obtížností vyrovnanou hru až do konečné fáze hry, kdy je hrací sáček prázdný, dokáže si tato úroveň spočítat zda má šanci vyhrát a pokud taková šance existuje, tak pravděpodobně hru opravdu vyhraje. Průměrně dosahuje přes 20 bodů za tah a jeho úroveň je srovnatelná s profesionálními hráči.

Názvy jednotlivý obtížností jsou náhodně vybrány z nejrůznějších počítačových her tak, aby co nejpřesněji charakterizovaly danou úroveň. Všechny parametry uvedené u jednotlivých úrovní byly měřeny ve hrách proti úrovni „profesional“. Podrobnější údaje jsou uvedeny v kapitole 8, zabývající se měřením výkonu v reálných hrách.

8 Praktické výsledky

Následující kapitola prezentuje výsledky v reálných hrách. Budou provedeny simulace herních partií mezi jednotlivými úrovněmi umělého protihráče s cílem zjistit, jaké důsledky budou mít jednotlivá vylepšení (nebo omezení) na konečné výsledky. Bude provedeno také porovnání statistik reálných hráčů České Asociace Scrabble [CAS] a našeho programu.

Simulace probíhaly na stroji Intel Celeron 1400MHz, 512 MB RAM, operační systém Windows XP Professional SP2, pomocí jednoduchého simulačního nástroje, který se nachází na přiloženém CD.

8.1 Vzájemné porovnání jednotlivých obtížností

Mezi porovnávané metriky v této části patří následující: počet vítězství, počet bodů za hru, průměrný počet bodů za tah, průměrná délka tahu, průměrné množství generovaných tahů a počet zahraných *bing* za hru. Všechny obtížnosti jsou porovnávány s úrovni *profesional*, jakožto základní úrovní hrající pomocí hladové strategie. Aby byly výsledky co nejméně zkreslené, bude simulováno dostatečné množství partií.

- Simulování her obtížnosti *amateur* dopadlo následovně.

úroveň	amateur	profesional
vítězství	6	993
vítězství(%)	1	99
body/hra	156	336
body/tah	9	19
délka tahu(ms)	10	12
nalezená slova/tah	42	92
binga/hra	0	0,17

Obrázek 20: Statistiky her úrovně amateur

Úroveň amatér má implementováno omezení na 10 bodů na tah, z tabulky vidíme že dosahuje průměrně menšího počtu bodů, především proto, že s pozdější fází hry už není na výběr takové množství tahů.

- Následující tabulka ukazuje odehrané hry úrovně *adept*.

úroveň	adept	profesional
vítězství	19	980
vítězství(%)	2	98
body/hra	183	323
body/tah	12	19
délka tahu(ms)	4	13
nalezená slova/tah	14	87
binga/hra	0,15	0,15

Obrázek 21: Statistiky her úrovně adept

Oproti nejnižší úrovni dosahuje adept o něco vyšší procento vítězství, generátor neprochází celou datovou strukturu slovníku (z důvodu omezení průchodu), tudíž generování trvá menší dobu, z tabulky vidíme že až 3x rychleji. Poměrně překvapivé je potom četnost generování *binga*, která je stejná je stejná jako u úrovně *profesional*.

- Testování úrovně *experienced* je uvedeno zde.

úroveň	experienced	profesional
vítězství	282	711
vítězství(%)	28	71
body/hra	268	300
body/tah	16	18
délka tahu(ms)	14	13
nalezená slova/tah	81	84
binga/hra	0	0,13

Obrázek 22: Statistiky her úrovně experienced

Úroveň *experienced* má implementováno omezení v podobě délky slova, které může být zahrané (konkrétně 4). Vidíme, že i s použitím krátkých slov lze dosáhnout poměrně vysokého ohodnocení bodů za tah i za celou hru, především díky využívání bonusových polí. Hraním kratších slov se hra stává více uzavřená a defensivní, což se projevuje i na bodovém zisku protihráče. Jelikož úroveň *experienced* nikdy neodehraje ze svého zásobníku více než 4 hrací kameny, nemá šanci dosáhnout 50 bodové prémie.

- Simulace her úrovně *profesional.*

úroveň	profesional	profesional
vítězství	511	483
vítězství(%)	51	48
body/hra	293	293
body/tah	19	19
délka tahu(ms)	16	16
nalezená slova/tah	101	101
binga/hra	0,17	0,17

Obrázek 23: Statistiky her úrovně profesional

Touto simulací bylo pouze testováno, jak moc vyrovnaná bude partie mezi dvěma stejnými obtížnostmi. Vidíme, že až na počet vítězství jsou všechny uvedené parametry shodné. To za předpokladu, že se hráči spravedlivě střídají v zahajování hry, pokud by tak tomu nebylo, vypadaly by výsledky následovně.

úroveň	profesional	profesional
vítězství	550	444
vítězství(%)	55	44
body/hra	295	288
body/tah	18	18
délka tahu(ms)	15	15
nalezená slova/tah	98	103
binga/hra	0,14	0,15

Obrázek 24: Simulace her úrovně profesional

Z tabulky vidíme, jaký vliv má počátek partie na konečný výsledek hry. Hráč který ve SCRABBLE začíná hru svým tahem, má na své straně výhodu, která rozhodně není zanedbatelná.

- Následující tabulka ukazuje simulace her úrovně *inhuman*.

úroveň	inhuman	profesional
vítězství	571	424
vítězství(%)	57	42
body/hra	314	295
body/tah	20	19
délka tahu(ms)	33	13
nalezená slova/tah	262	94
binga/hra	0,48	0,15

Obrázek 25: Simulace her úrovně inhuman

Úroveň *inhuman* (oproti úrovni *profesional*) nevybírá tah podle nejvyššího skóre, ale bere v úvahu i vliv zbylých kamenů na zásobníku na budoucí vývoj hry. Lépe použitelné kameny se tak snaží šetřit a zbavuje se především špatně hratelných kamenů. Už s použitím jednoduché zásobníkové heuristiky, popsané podrobněji v 7.3.3, je vidět pozitivní vliv této strategie na vývoj her. Ponecháváním si lepších kamenů stoupla šance na 50 bodovou prémii 3x, z původního 0,15 *binga* za hru, na 0.48. Také počet nalezených slov na tah se významně zvětšil. Z daných výsledků je vidět, jak důležité je ve SCRABBLE správné uspořádání zásobníku.

- Výsledky úrovně godlike.

úroveň	godlike	profesional
vítězství	318	179
vítězství(%)	64	35
body/hra	327	292
body/tah	20	19
délka tahu(ms)	7700	13
nalezená slova/tah	241	99
binga/hra	0,46	0,17

Obrázek 26: Simulace her úrovně godlike

Godlike je nejlepší z implementovaných úrovní, dosahuje nejvyššího průměrného skóre za hru i za tah. Implementace řešení konce hry má i po optimalizaci za následek poměrně výrazný nárůst času, potřebného pro generování všech tahů. Z časových důvodů byl omezen počet simulací her na 500.

8.2 Porovnání s reálnými hráči

Důležitým měřítkem pro hodnocení úspěšnosti umělého protihráče je porovnání s lidskými protivníky. Aby bylo takové srovnání důvěryhodné, bylo by zapotřebí testovat daný program profesionálními hráči SCRABBLE na dostatečném množství her. Takovou možnost bohužel nemáme, úspěšně ovšem můžeme porovnat statistiky dosahované lidskými hráči se statistikami programu. K tomu nám poslouží statistiky uvedené na stránkách české asociace SCRABBLE, k co nejmenšímu skreslení informací byly použity statistiky pětileté. Porovnávat se potom budou hlavní měřitelné parametry, a to počet bodů za hru, počet bodů za tah a počet *bing* za hru.

Následující tabulka ukazuje v případě člověka statistiky první stovky hráčů žebříčku české asociace SCRABBLE. Úplné údaje lze najít na [CAS]. U umělého protivníka jde o hodnoty úrovně *godlike* získané v průběhu simulací.

	člověk	godlike
body/hra	0,61	0,47
body/tah	20,7	20,4
binga/hra	342	327

Obrázek 27: Statistiky profesionálních hráčů SCRABBLE a našeho programu.

Vidíme, že i když náš program nedosahuje úrovně nejlepších profesionálních hráčů, alespoň se jim přibližuje. Jasným důvodem je slabší práce se zbytkem kamenů na zásobníku. Profesionální hráči se snaží nechávat si různé výhodné skupiny písmen tvořících předpony (např „ne“, „nej“) nebo naopak přípony („ají“, „ejší“). Tím si vytváří lepší výchozí pozici pro budoucí tahy a případné zahrání *binga*.

Zatímco nejlepší český hráč dosahuje v počtu zahraných 50 bodových prémií za kolo hodnoty 1,35, nejvyšší implementovaná úroveň v programu dosahuje přibližně hodnoty 0,5 *binga*. Pro zajímavost uvedeme ještě průměry druhé stovky hráčů v oficiálním žebříčku.

	člověk	godlike
body/hra	0,35	0,47
body/tah	18,3	20,4
binga/hra	317	327

Obrázek 28: Statistiky profesionálních hráčů SCRABBLE a našeho programu

Podle těchto hodnot můžeme usoudit, že mezi těmito slabšími hráči by již náš program vynikal. Ve všech změřených parametrech dosahuje nadprůměrných výsledků.

9 Závěr

Cílem práce bylo vytvoření webové aplikace společenské stolní hry SCRABBLE, která by posloužila k hraní amatérským hráčům SCRABBLE a zároveň by byla schopná potrápit hráče profesionální. K dosažení vytyčených cílů tak vznikl systém rozdělený do 6 výkonnostních úrovní, ve kterém by si měl každý najít optimálního oponenta. Vytvořené jádro hry je schopno generovat tahy s velice dobrou odezvou. I nejvyšší obtížnost, používající stavové prohledávání prostoru v konečném stádiu hry, je schopna nalézt optimální tah v řádech sekund. I z tohoto důvodu nebylo uvažováno časové omezení hry, které by hráče spíše obtěžovalo. Zmíněné omezení by mělo smysl používat v případě vzájemné hry dvou lidských protihráčů. I když tuto možnost naše aplikace neumožňuje, je to jedna z cest, kterou by se budoucí vývoj mohl ubírat, jádro hry je na tuto skutečnost připravené, stačilo by tedy přizpůsobit uživatelské prostředí.

Přestože vytvořená umělá inteligence neobstojí v porovnání s českou špičkou, je schopná konkurovat většině profesionálních hráčů. Rezervy lze stále hledat v organizaci zásobníku a odehrávání tahů s ohledem na budoucí možný vývoj partie. Už námi vymyšlený způsob ohodnocení zbytku písmen na zásobníku prokázal své výsledky. S použitím této funkce jsme dokázali ztrojnásobit pravděpodobnost položení padesátibodové prémie a umělý protivník dokázal generovat mnohem více slov. Budoucí vývoj by se tak měl věnovat podrobné analýze používaných slovníků a zjištění, která písmena a množiny písmen je výhodné si ponechávat, což by vedlo k ještě lepší organizaci zásobníku a většímu bodovému zisku. Úroveň umělého protihráče dále roste s používáním robustnějšího slovníku, lepších výsledků bychom tak dosáhli používáním slovníku se všemi možnými tvary platnými podle pravidel přípustnosti slov české asociace SCRABBLE.

Hráč hrající SCRABBLE proti umělému protivníkovi je neustále konfrontován s novými slovními výrazy. Jelikož oponentovy tahy vychází z rozsáhlého slovníku, používá i zastaralá, málo používaná a často až obskurní slova. Častým hraním se hráči tato slova vryjí do paměti a postupem času už je má hráč zautomatizována. I já jsem tak po chvilce začal používat slova jako „xu“, „oó“, „aak“ nebo „aam“ a rozšiřovat si tak slovní zásobu. Stejně tak se každý rychle naučí využívat výhod bonusových hracích polí a naopak nenahrávat protivníkovi tak, aby tato pole mohl snadno využívat. Kromě potěšení ze hry tak častým hraním prohlubujeme své taktické znalosti.

Literatura

[AJ88] Appel, A. W., Jacobson, G. J. *The world's fastest Scrabble program*, 1988, Communication ACM, vol. 31, s. 572–585.

[Gor94] Gordon, S. A. *A faster Scrabble move generation algorithm*, 1994, Software-Practice and Expirience, vol. 24, s. 219-232.

[LK92] Lucchesi, C. L., Kowaltovski, T. *Applications of Finite Automata Representing Large Vocabularies*, 1992

[She02] Sheppard, B. *World-championship-caliber Scrabble*, 2002, Artificial Inteligence 134, s. 241-275.

[RA07] Richards, M., Amir E. *Oponent modeling in Scrabble*, 2007, Proceedings of the Twentieth. International Joint Conference on Artificial Inteligence, s. 1482-1487

[Sha79] Shapiro, S. C. *A scrabble crossword game playing program*, 1979, Proceedings of the Sixth IJCAI, s. 797-799

[Sha01] Shaeffer Jonathan. *A Gamut of Games*, 2001, AI Magazine Volume 22 Number 3, s. 35-37

[Vy98] Vychodil Villém. *Prohledávání stavového prostoru*, 1998

[CAS] Česká asociace Scrabble, 2007
URL: http://scrabble.hrejsi.cz/, [online], 12. 5. 2008

[Wic06] Wickman, J. B. *Scrabble on the TeraScale*, Univerzity of Nebraska-Lincoln, 2006
URL: http://www.gtoal.com/wordgames/bwickman/, [online], 3. 5. 2008

[Wik1] Java, Wikipedie
URL: http://cs.wikipedia.org/wiki/Java, [online], 8. 5. 2008

[Wik2] NetBeans, Wikipedie
URL: http://cs.wikipedia.org/wiki/NetBeans, [online], 14. 4. 2008

[Sun1] Applets, Sun Developer Network (SDN)
URL: http://java.sun.com/applets/, [online], 12. 2. 2008

[KT01] Kotala Z., Toman P. Java – Abstract Window Toolkit (AWT), Dioné, 2001
URL: http://dione.zcu.cz/java/sbornik/18.html, [online], 4. 4. 2008

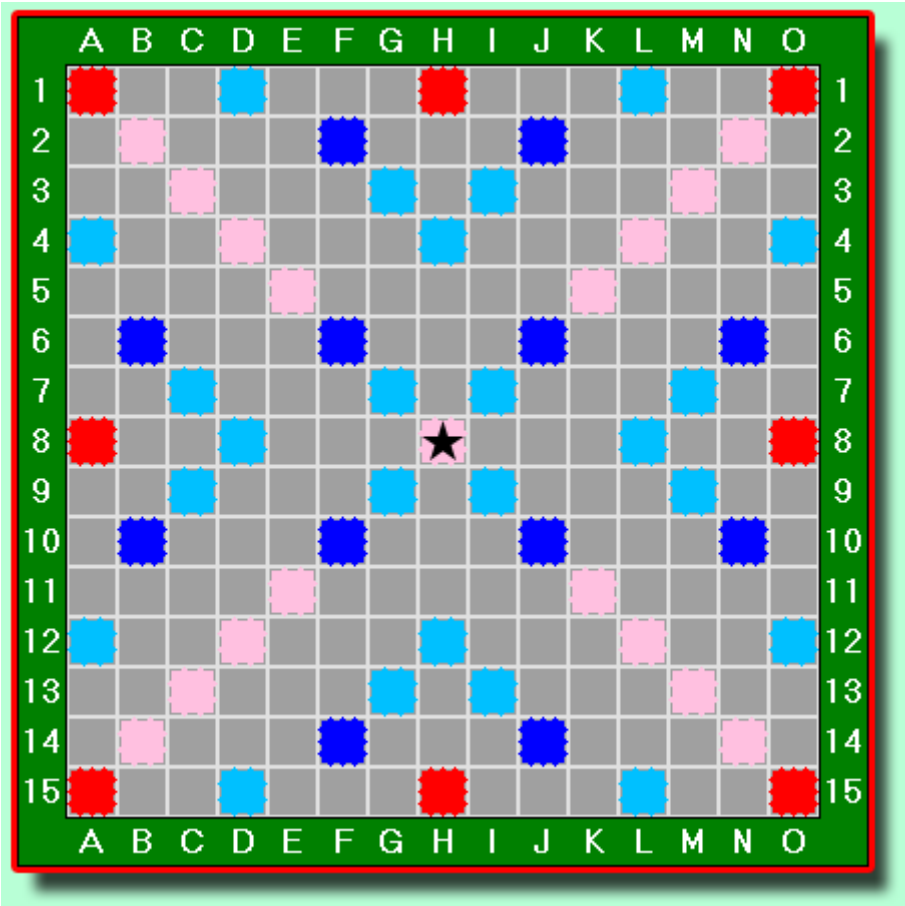
Seznam příloh

Příloha 1. Hrací deska

Příloha 2. Ukázka grafického výstupu programu

Příloha 3. Pravidla přípustnosti slov ČAS

Hrací deska



Dvojnásobná hodnota slova

Trojnásobná hodnota slova

Dvojnásobná hodnota písmene

Trojnásobná hodnota písmene

Ukázka grafického výstupu programu

Pravidla přípustnosti slov ČAS

Základní ustanovení

1. Tato Pravidla přípustnosti slov (dále jen PPS) jsou závazná pro veškeré soutěže, pořádané pod hlavičkou České asociace SCRABBLE® (dále jen ČAS). PPS jsou zpracována pro dvě konkrétně definované varianty hry, a to „VARIANTU PLUS“ (dále jen VP) a „VARIANTU KLASIK“ (dále jen VK). Není-li uvedeno jinak, platí níže uvedená ustanovení pro obě varianty.

Slovníky ČAS

2. Slovníky ČAS jsou jednotlivé konkrétní množiny slov, vytvořené (speciálně pro rozhodování o přípustnosti slov ve scrabblu) v souladu s PPS (body 3 a 5 - 8) a definované konkrétním tištěným dílem nebo elektronickou verzí (včetně případných oprav k těmto dílům či verzím). Část 2.0 je k dispozici v elektronické podobě ve formě vyhledávacího programu, části 2.1 - 2.2 jsou k dispozici kompletně v tištěné podobě. Podle jednotlivých Slovníků ČAS se rozhoduje v případech, stanovených konkrétně v bodě 4. Slovníky ČAS jsou tato díla:

2.0 Jiří Novotný - Petr Vetešník: NOVEX (Elektronický slovník) (dále jen NOVEX) - zahrnuje (ve formě souhrnného vyhledávacího programu) v elektronické podobě kompletně všechna (pro hru přípustná) 2-9písmenná slova[1]), obsažená v částech 2.1 - 2.3 (tj. BLEX1, BLEX+3 a BLEX+7). NOVEX je zpracován ve dvou verzích: NOVEX PLUS (pro VP) a NOVEX KLASIK (pro VK).

2.1 Jiří Novotný: BLEX 2001 - Seznam dvou až pětipísmenných slov přípustných pro hru scrabble (modrá obálka): s výjimkou slov označených „ : “ (dále jen BLEX1)

2.2 Jiří Novotný: BLEX PLUS 2003 - Slovník dvou až pětipísmenných tvarů slov přípustných pro hru scrabble (2. rozšířené vydání - červená obálka) (dále jen BLEX+3):
pro VP: kompletně
pro VK: s výjimkou slov označených „ +, », « “

2.3 Jiří Novotný: BLEX PLUS 2007 - Slovník dvou až pětipísmenných tvarů slov přípustných pro hru scrabble (3. rozšířené vydání - zelená obálka) (dále jen BLEX+7)

- - - - -

[1]) výjimkou jsou slova, v nichž se vyskytují písmena s diakritickými znaménky, která nelze do elektronického slovníku vůbec zapsat. Seznam těchto slov, doplněný o slova, která lze zapsat výhradně pomocí klávesy Alt, je uveden v Příloze 1 těchto PPS (viz též 3.2.4).

Prameny

3. Podle Pramenů se rozhoduje v případech, stanovených konkrétně v bodě 4. Prameny jsou tato díla:

3.1 Prameny normativní:

3.1.1 Slovník spisovného jazyka českého I-VIII (včetně Doplňků a oprav uvedených v VIII. díle). 2. vydání, Academia, Praha 1989 (dále jen SSJČ).

3.1.2 Slovník spisovné češtiny pro školu a veřejnost. 3. vydání, Academia, Praha 2004 nebo

4. vydání (s podtitulem Studentské vydání), Academia, Praha 2005 (dále jen SSČ).

3.1.3 Pravidla českého pravopisu - akademická. Academia, Praha 1993 (dále jen APČP).

3.1.4 Pravidla českého pravopisu. Školní vydání včetně Dodatku. 2. rozšířené vydání, Fortuna, Praha 1999 (dále jen ŠPČP).

3.1.5 Nový akademický slovník cizích slov. Academia, Praha 2005 (dále jen NASCS).

3.1.6 Sochová, Poštolková: Co v slovnících nenajdete. Portál, Praha 1994 (dále jen CVSN).

3.1.7 Martincová: Nová slova v češtině. Slovník neologizmů 1-2. Academia, Praha 1998 - 2004 (dále jen SN 1, SN 2).

3.1.8 Polívková: Naše místní jména a jak jich užívat (část VII: Abecední seznam místních jmen, str. 52-201), Euromedia Group, Praha 2007, 2. upravené a rozšířené vydání

3.2 Prameny ČAS:

3.2.1 Seznam všech přípustných názvů písmen české abecedy. ČAS, 1999.

3.2.2 Seznam všech přípustných názvů písmen řecké abecedy. ČAS, 2000.

3.2.3 Seznam všech přípustných názvů zvýšených a snížených tónů. ČAS, 2000.

3.2.4 Seznam všech 2-9písmenných slov, v nichž se vyskytují písmena s diakritickými znaménky, která nelze na počítači (Cz) ve slovníku Novex zapsat buď vůbec, anebo pouze s pomocí klávesy Alt. ČAS, 2005.

Posuzování přípustnosti slov

4. Přípustnost slov je posuzována pouze v případě vznesení námitky proti přípustnosti konkrétního zahraného slova. Tuto přípustnost posuzují slovníkoví rozhodčí (přičemž funkci slovníkového rozhodčího může zastávat též hlavní rozhodčí) podle Slovníků ČAS (viz bod 2): Rozhodnutí rozhodčím jednou učiněné je v rámci hry konečné a nemůže být měněno. Elektronické i tištěné verze jsou pro posuzování přípustnosti slov rovnocenné. Slova, jejichž přípustnost nelze (např. z důvodu jejich délky nebo z důvodu, uvedeného v poznámce u bodu 2.0) posoudit podle Slovníků ČAS, je vždy nutno posuzovat podle Pramenů (viz bod 3). Přípustnost slov se posuzuje na základě délky slova (tj. počtu písmen daného slova, přičemž je písmeno „ch“ považováno za dvě písmena) následujícím způsobem (viz též Příloha 2):

4.1 <u>platí pouze pro VP:</u>

a) při rozhodování se posuzují

4.1.1 2–9písmenná slova podle slovníku NOVEX PLUS (s výjimkou, uvedenou u bodu 2.0) (2–5písmenná slova lze posuzovat též podle slovníku BLEX+3 nebo BLEX+7).

4.1.2 10–15písmenná slova podle PRAMENŮ.
V některých případech (viz bod 4.3) je však možno i tato slova posuzovat podle slovníku NOVEX PLUS.

4.2 <u>platí pouze pro VK:</u>

a) při rozhodování se posuzují

4.2.1 2–9písmenná slova podle slovníku NOVEX KLASIK (s výjimkou, uvedenou u bodu 2.0) (2–5písmenná slova lze posuzovat též podle slovníku BLEX1 nebo BLEX+3).

4.2.2 10–15písmenná slova podle PRAMENŮ. V některých případech (viz bod 4.3) je však možno i tato slova posuzovat podle slovníku NOVEX KLASIK.

4.3 <u>Poznámka ke slovníkům NOVEX PLUS a NOVEX KLASIK</u>

Pro posuzování přípustnosti slov jsou slovníky NOVEX PLUS a NOVEX KLASIK plně použitelné pouze v rozsahu, stanoveném v bodech 4.1.1 a 4.2.1. Při systémech rozhodování,

stanovených Soutěžním řádem v bodech 9.4.1 – 9.4.3, lze však v mnoha případech tyto slovníky použít i při posuzování tvarů v nich přímo neuvedených, tj. tvarů delších než 9písmenných. Tvary zde přímo neuvedené je možno v některých případech posuzovat na základě přípustnosti tvarů zde uvedených (např. 1. pádu či infinitivu). Nedokáže–li však slovníkový rozhodčí takovýto tvar tímto způsobem bezpečně posoudit, je povinen ho posoudit podle PRAMENŮ.

[10písmenné slovo „maturantův“ je možno ověřit na základě přípustnosti slova „maturant“]

Nepřípustné kategorie slov

5. Následující kategorie slov jsou pro hru nepřípustné (viz též Příloha 3):

5.1 Slova specificky označená - hledisko stylistické

Je-li u výrazu uvedeno více stylistických kategorií, je třeba rozlišovat, zda jsou tyto kategorie uváděny výčtem bez použití mluvnických spojek („a“, „i“, „nebo“, „ , “ („čárka“)), nebo s jejich použitím. Pokud je spojka použita, jde o různé významy a je-li aspoň jeden z nich přípustný, je i slovo přípustné [zast. a knižní.]. Naproti tomu uvedení více kategorií v řadě bez použití spojky označuje jediný význam a je-li aspoň jedna z těchto kategorií nepřípustná, je i slovo nepřípustné [zast. knižní.].

5.1.1 Slova zastaralá. [mluno]
5.1.2 Slova řídká (zřídka používaná). [kárce]
5.1.3 Slova nářeční. [mrskut]
5.1.4 Slova vulgární. [hajzl]
5.1.5 Slova argotická. [čík]
5.1.6 Slova nesprávná (nesprávně utvořená). [jeřábista, lyžiny]
5.1.7 Slova nevhodná (nevhodně utvořená). [peleton (NASCS)]

5.2 Slova specificky označená - hledisko nestylistické (např. pravopis, slovní druhy, značky)

5.2.1 Citoslovce [prr, tú, hó]
5.2.2 Zkratky [aj., apod., atd., např.]
5.2.3 Značky [km, hl]
5.2.4 Slova psaná starým pravopisem (označená „dříve psáno“, „původně psáno“)
5.2.5 Slova „ve spojení“ viz bod 6.1.
5.2.6 Slova, jichž se v heslové podobě nepoužívá (jen SSČ) viz bod 6.2.

5.3 Slova specificky neoznačená

5.3.1 Jednopísmenné výrazy [k]

5.3.2 Slova obsahující velké písmeno [Afrika, Josef, eBanka]

5.3.3 Výrazy, obsahující jiné znaky než písmena [chceš-li]

5.3.4 Odkazy, neuvedené v cílovém hesle [ix viz x]

5.3.5 Víceslovná hesla viz bod 6.3.

5.3.6 Nesamostatná slova a slova, vytvořená spojením těchto nesamostatných slov s jinými slovy (pokud nejsou takováto slova přípustná na základě jiných hesel), není-li v konkrétní případech (viz body 8.3 a 8.4) stanoveno jinak.(Nesamostatná slova jsou slova doplněná spojovníkem. Jedná se jednak o slova heslová, uvedená v Pramenech [pro-, prů-, vz-, vze-, iso-, izo-, mikro-, ezo-; -fág, -fagie, -fil, -filie, -fob -fobie, -ositi, -uhliti, -ť, -tě, -ž, -že], jednak o zkrácené (nesamostatné) tvary odvozených slov [např. „-s“, tj. zkrácený tvar slova „jsi“, odvozeného od hesla „být“: byls, nesens, mnohos, anebos, dceras, tys, žes]).

Kategorie slov, přípustné jen za určitých podmínek

6. Následující kategorie slov jsou pro hru přípustné jen za níže uvedených podmínek:

6.1 Slova „ve spojení“

6.1.1 Jednoslovná heslová slova, u nichž je uvedena poznámka typu „jen v ustáleném spojení“, „v ustáleném spojení“, „jen ve spojení“, „ve spojení“, „jen v pořekadle“, „v pořekadle“, „jen v přísloví“, „v citátovém spojení“, „jen v knižním spojení“, „jen v odborném spojení“ apod., mohou být pro hru přípustná pouze ve tvaru, uvedeném v tomto „spojení“.
[heslo „zoun“ - (jen v ob. expr. spoj. jako zoun) - tvar „zoun“ je přípustný;
heslo „čud“ (jen v ob. expr. ust. spoj. být v čudu - tvar „čudu“ je přípustný (jen ve VP)]

6.1.2 Jiné tvary (než uvedené ve „spojení“), vytvořené ohýbáním, lze od těchto slov tvořit pouze v případě, že lze v tomto „spojení“ současně ohýbat všechna ohebná slova.
[alfa a omega = počátek a konec, (bez) alfy a omegy = (bez) počátku a konce, …;
hladká ančka = mléčná polévka, …, (s) hladkou ančkou = (s) mléčnou polévkou, …]

6.2 Slova nepoužívaná v heslové podobě (platí jen pro SSČ)

Slova, jichž se v heslové podobě nepoužívá (v SSČ označena „ * “), jsou v této heslové podobě nepřípustná [* bach]. Přípustné jsou pouze tvary, uvedené u jednotlivých hesel v příkladech [dát si bacha - tvar „bacha“ je přípustný (jen ve VP)].

6.3 Víceslovná hesla

6.3.1 Jednotlivé části víceslovného hesla (s výjimkami, uvedenými v bodech 6.3.2 - 6.3.4) jsou pro hru nepřípustné .
[na základě hesla „big beat“ není přípustné ani „big“, ani „beat“];
(Takováto slova však mohou být přípustná na základě jiných hesel: [např. „beat“ na základě samostatného přípustného hesla „beat“, „big“ jako tvar přípustného hesla „biga“ (ve VP)].)

6.3.2 Přídavná jména, vyskytující se v NASCS v části Stručný přehled jazyků světa (NASCS, str. 827 - 834) pouze ve víceslovném spojení (jako slova heslová), jsou pro hru přípustná. Použít je lze samostatně ve všech rodech.
[akanské jazyky - tvary „akanský, akanská, akanské, akanští“ jsou přípustné]

6.3.3 Přídavná jména, vyskytující se v APČP v části Seznam antických jmen (APČP, str. 382 - 389) pouze ve víceslovném spojení (jako slova odvozená od základních hesel), jsou pro hru přípustná. Použít je lze samostatně ve všech rodech.
[elejská škola (heslo Elea) - tvary „elejský, elejská, elejské, elejští“ jsou přípustné]

6.3.4 Jednotlivé části zvratných sloves [smát se] jsou pro hru přípustné. Použít je lze pouze samostatně, a to jak sloveso [smát], tak zvratnou částici [se].

Přípustné tvary slov

7. Pro rozhodování podle PRAMENŮ platí tato pravidla:

7.0 Tvary slov přípustných pro hru

7.0.1 Přípustné pro hru je automaticky každé slovo, které je alespoň v jednom Pramenu uvedeno jako slovo heslové (není-li ovšem v tomtéž Pramenu zařazeno do nepřípustné kategorie - viz body 5 a 6). Pro odvozená slova, uvedená v Pramenech u základního hesla jiným typem písma (např. slova zdrobnělá), platí stejné podmínky, jako pro slova heslová. Jiná než heslová

slova jsou přípustná pouze v případech, stanovených těmito PPS. (Cizí slova, uvedená v Pramenech jako slova heslová, považujeme za slova do češtiny přejatá a tudíž přípustná pro hru.)

7.0.2. Od přípustných heslových slov lze tvořit další tvary podle platných mluvnických zásad jazyka českého (skloňováním, stupňováním, časováním). Patřičné tvary se ověřují podle mluvnice jazyka českého. (Jako základní se doporučuje *Mluvnice češtiny 1-3*, Academia, Praha 1986-1987; pro hru je hlavní 2. díl - Tvarosloví.)

7.0.3 U tvarů, správně vytvořených podle výše uvedených zásad, se neověřuje jejich smysluplnost a správnost z hlediska významového [např. „ženatá, vdaný, zrňme, voň“ jsou tvary přípustné].

7.0.4 Obligatorní tvary, správně vytvořené od přípustného heslového slova, jsou přípustné i v případě, že jsou v Pramenech označeny jako řídké (zřídka se vyskytující).
Existují-li ovšem jiné synonymní tvary, jsou tvary označené jako řídké (zřídka se vyskytující) přípustné pouze v tom případě, není-li přípustný žádný z uvedených synonymních tvarů.

7.0.5 Tvary, uvedené v seznamu „Opravy nedostatků, vyskytujících se v Pramenech“ (viz Příloha 4) ve sloupci „A“, jsou pro hru nepřípustné. Přípustné pro hru jsou tvary, uvedené ve sloupci „B“.

7.0.6 Při splnění přípustnosti daného heslového slova jsou u jednotlivých slovních druhů přípustné tvary (heslové nebo vytvořené v souladu s mluvnicí), uvedené v bodech 7.1 - 7.9.

7.1 Podstatná jména

7.1.1 VP: 1.-7. pád čísla jednotného i množného;
VK: 1. pád čísla jednotného i množného.

7.2 Přídavná jména

7.2.1 VP: 1.-7. pád čísla jednotného i množného všech rodů;
VK: 1. pád čísla jednotného i množného všech rodů.

7.2.2 1., 2. a 3. stupeň [zelený, zelenější, nejzelenější]

7.2.3 Jmenné tvary, jsou-li uvedeny alespoň v jednom tvaru v Pramenech [(i jm. živ); (… též tvary jmenné šťasten, -tna, -tno,…)], a to včetně tvarů, uvedených pouze v příkladech [buď tak laskav, jsem zvědav]. - (Predikativa viz bod 7.6.2.)

7.2.4 Záporné tvary jsou přípustné pouze v případě, jsou-li přímo uvedeny alespoň v jednom rodě v Pramenech.

7.3 Zájmena

7.3.1 VP: 1.-7. pád čísla jednotného i množného všech rodů a druhů;

VK: 1. pád čísla jednotného i množného všech rodů a druhů.

7.4 Číslovky

7.4.1 VP: 1.-7. pád čísla jednotného i množného všech rodů a druhů;
VK: 1. pád čísla jednotného i množného všech rodů a druhů.

7.4.2 Číslovky určité všech rodů a druhů jsou přípustné i v případě, že nejsou v Pramenech konkrétně uvedeny (pouze však v případě, jsou-li v Pramenech uvedeny srovnatelné přípustné tvary jiných, většinou menších, číselných hodnot).

7.5 Slovesa

Pro hru jsou přípustná slovesa nezvratná i slovesa zvratná (přičemž tato se používají bez zvratné částice „se“ - viz též bod 6.3.3).
Pro hru jsou přípustné tyto tvary sloves:

7.5.1 Infinitiv v obou tvarech („-t/-ti“ nebo „-ct/-ci“) [nést i nésti, říct i říci].

7.5.2 1.-3. osoba jednotného i množného čísla přítomného i budoucího času, včetně všech zakončení hovorových a knižních (pokud dané sloveso patří ke vzoru, u něhož jsou tato zakončení uvedena v Pramenech nebo Mluvnici češtiny 2)
[nesu, neseš, nese, nesem(e), nesete, nesou; ponesu, poneseš, ponese, ponesem(e), ponesete, ponesou].

7.5.3 Příčestí činné (minulý čas) v jednotném i množném čísle všech rodů [nesl, nesla, neslo, nesli, nesly]. (Nepřípustné je vynechávání koncového „-l“ v mužském rodě jednotného čísla [„fouk“ namísto přípustného „foukl“].)

7.5.4 Rozkazovací způsob [nes, nesme, neste, pones, ponesme, poneste]

7.5.5 Příčestí trpné (výjimky z možností zde uvedených viz mluvnice):
a) v jednotném i množném čísle všech rodů přechodných sloves (tj. nezvratných sloves s předmětem ve 4. pádě bez předložky) [nesen, nesena, neseno, neseni, neseny];
b) v jednotném čísle středního rodu nezvratných sloves s předmětem v jiném než 4. pádě bez předložky [pomoženo].

7.5.6 VP: Přechodník přítomný nedokonavých sloves [dělaje, dělajíc, dělajíce].
(Přechodník přítomný dokonavých sloves [řka, řkouc, řkouce] je pro hru přípustný pouze v případě, je-li alespoň v jednom tvaru uveden v Pramenech.)

7.5.7 VP: Přechodník minulý dokonavých sloves [udělav, udělavši, udělavše].
(Přechodník minulý nedokonavých sloves [byv, byvši, byvše] je pro hru přípustný pouze v případě, že je alespoň v jednom tvaru uveden v Pramenech.)

7.5.8 Všechny tvary, přípustné u slovesa bez předpony (s výjimkou tvarů, vytvořených podle bodů 7.5.5 - 7.5.7), platí automaticky u slovesa s předponou (pokud daný tvar není tvořen nepravidelně). Toto pravidlo lze použít i obráceně, tedy přenést tvary uvedené u slovesa s předponou k příslušnému slovesu bez předpony (resp. k nesamostatnému slovesnému základu) a z něj dále k odvozeným slovesům s jinými předponami [počešť / -češť / odčešť].

7.6 Příslovce

7.6.1 1., 2. a 3. stupeň [zeleně, zeleněji, nejzeleněji]

7.6.2 Predikativa (kategorie stavu - v některých mluvnicích považováno za samostatný slovní druh), uvedená v Pramenech [označování predikativ není jednotné: možno přísl. v přís. (SSČ); možno přísl. kniž. (SSJČ); … ve spoj. je patrno; … kniž. (je) slušno], včetně všech predikativ zakončených na „-o“, uvedených pouze v příkladech [libo, nutno] (jedná se v podstatě o jmenný tvar přídavného jména rodu středního v jednotném čísle).

7.7 Předložky

Tvary uvedené v Pramenech.

7.8 Spojky

Tvary uvedené v Pramenech.

7.9 Částice

Tvary uvedené v Pramenech.

Odvozování slov

8. Slova, neuvedená v Pramenech, mohou být utvořena pouze odvozením od slov, přípustných podle bodu 7. Od všech slov, vzniklých odvozením, lze dále v plné míře tvořit všechny tvary, přípustné podle bodu 7. Přípustné jsou následující skupiny slov:

8.1 Podstatná jména

8.1.1 Složená podstatná jména [jednočlen] vzniklá spojením nesamostatného podstatného jména [-člen] s číslovkou určitou celou (ve tvaru pro složená slova) [jedno-].

8.1.2 Podstatná jména významu číselného [pětka, stovka, tisícovka, pětina, miliontina]

8.1.3 Podstatná jména slovesná z (existujícího či fiktivního) příčestí trpného [dělání, bytí]

8.2 Přídavná jména

8.2.1 Přídavná jména individuálně přivlastňovací, odvozená od podstatných jmen, označujících živého či neživého jedince (osobu nebo zvíře) rodu mužského nebo ženského, tzn. od podstatných jmen obecných [otcův, matčin, psův, liščin, nebožtíkův] nebo zpersonifikovaných [čertův, robotův, ufonův].

8.2.2 Přídavná jména odvozená z příčestí trpného [nesený, nesená, nesené, nesení]

8.2.3 Složená přídavná jména [jednočlenný] vzniklá spojením nesamostatného přídavného jména [-členný] s číslovkou určitou celou (ve tvaru pro složená slova) [jedno-].

8.2.4 Přídavná jména slovesná z přechodníku přítomného nedokonavých sloves [dělající]. (Přídavná jména slovesná z přechodníku přítomného dokonavých sloves lze tvořit pouze v případě, že je tvar tohoto přechodníku uveden alespoň v jednom tvaru v Pramenech [řkoucí].

8.2.5 VP: Přídavná jména slovesná z přechodníku minulého dokonavých sloves [udělavší]. (Přídavná jména slovesná z přechodníku minulého nedokonavých sloves lze tvořit pouze v případě, že je tvar tohoto přechodníku uveden alespoň v jednom tvaru v Pramenech [byvší]).

8.3 Zájmena

VP: Stažené tvary, vytvořené spojením zkráceného tvaru 4. pádu zájmena s předložkou [o co = oč, na něho = naň].

8.4 Slovesa

Slovesa vzniklá přidáním jedné z předpon uvedených u daného hesla (přičemž se za přípustné heslo v tomto případě považuje i nesamosatný slovesný základ), není-li ovšem takto vytvořené sloveso označeno v samostatném hesle téhož Pramenu jako slovo nepřípustné - viz body 5 a 6). (Od takto vzniklých sloves lze tvořit všechny tvary jako od sloves, majících samostatné heslo - viz bod 7.5.)

8.5 Příslovce

Příslovce významu číselného [poprvé, posté, podvaadvacáté, zaprvé, zasté]

8.6 Předložky

VP: Stažené tvary, vytvořené spojením předložky a zkráceného tvaru zájmena - viz bod 8.3.

8.7 Předpona „ne-„

Ke všem přípustným tvarům sloves (viz bod 7.5) a tvarům od nich odvozených (viz body 8.1.3, 8.2.3, 8.2.4 a 8.4) je možno vytvořit záporný tvar přidáním předpony „ne-“ (pokud není tento záporný tvar tvořen nepravidelně [je - není]).