Nekooperativní hry v normální formě
Genetické algoritmy pro vyhledání Nashova Ekvilibria

Petr Dvořáček – xdvora0n@stud.fit.vutbr.cz

21. ledna 2015

1 Úvod

Nalezení Nashova ekvilibria více hráčů je výpočetně náročný problém, neboť prohledávací prostor se zvětšuje s rostoucím počtem hráčů a jejich strategií. Jedním z možných přístupů se osvědčilo použití Lemke-Howsonova algoritmu, který vyhledá smíšené Nashovo ekvilibrium ve hře se dvěma hráči. Pro hry o více hráčích se vyplatí použít evoluční algoritmy, jejichž výsledkem je nalezení suboptimálního řešení. Na toto téma byly v minulosti představeny různé genetické algoritmy (GA), např. NSGA což jsou multikriterální GA (optimalizují více kriterií najednou, výsledkem je Paretoa množina), či koevoluční přístup v GA (souběžná evoluce dvou a více populací). V tomto projektu se zabývám právě evolučním přístupem k vyhledání ryzího Nashova ekvilibria her s více než dvěma hráči.

Tato práce je členěna následovně. V kapitole 2 se nachází definice ryzího a smíšeného Nashova ekvilibria. Kapitola 3 pojednává o základních principech genetických algoritmů (zkráceně GA). V kapitole 4 je vysvětleno použití GA k vyhledání ekvilibria. Experimentální vyhodnocení se nachází v kapitole 5. Práci uzavírá kapitola 6, kde shrnuji dosažené výsledky.

2 Nashovo ekvilibrium

Nechť Γ je hra, která je definovaná $(2N+1)$-ticí $\Gamma = (Q, S_1, S_2, \ldots, S_N, U_1, U_2, \ldots, U_N)$. Hra se skládá z množiny hráčů $Q = \{1, 2, \ldots N\}$, kde každý hráč $i \in Q$ hraje strategii s_i z konečné množiny ryzích strategií S_i. Funkce užitků hráče $i \in Q$ je definována jako $U_i : S_1 \times S_2 \times \ldots \times S_N \rightarrow \mathbb{U}$, kde $\mathbb{U}$ odpovídá universu všech možných užitků. Často klademe $\mathbb{U} = \mathbb{R}$, universum tedy odpovídá reálným číslům. Výsledkem rozhodování hráče $i \in Q$ je volba ryzí strategie $s_i^* \in S_i$. Jinými slovy, jedná se o model chování hráče. Nashovo ekvilibrium je pak taková volba strategií (profilu) $s^* = (s_1^*, s_2^*, \ldots, s_N^*)$ z hry Γ, pro níž platí $\forall i \in Q : \forall\, s_i \in S_i : U_i(s_1^*, s_2^*, \ldots, s_N^*) \geq U_i(s_1^*, s_2^*, \ldots, s_{i-1}^*, s_i, s_{i+1}^*, \ldots s_N^*)$. Toto ekvilibrium je nazýváno jako ryzí ekvilibrium a označujeme jej zkratkou PNE (podle anglického Pure Nash Equilibrium) [1].

Mějme hru $\Gamma = (Q, S_1, S_2, \ldots, S_N, U_1, U_2, \ldots, U_N)$. Vektor pravděpodobností $\sigma_i = (\sigma_i^1, \sigma_i^2, \ldots, \sigma_i^{|S_i|})$ se nazývá smíšená strategie hráče $i \in Q$ ve hře Γ, pokud platí: $\forall j \in \{1, 2 \ldots, |S_i|\} : \sigma_i^j \in \langle 0, 1\rangle$ a zároveň $\sum_{j=1}^{|S_i|} \sigma_i^j = 1$. Hra $\Gamma^m = (Q, \Delta_1, \Delta_2, \ldots, \Delta_N, \pi_1, \pi_2, \ldots, \pi_N)$ se nazývá smíšeným rozšířením hry Γ. Δ_i je dán vztahem: $\Delta_i = \{\sigma_i \in \langle 0, 1\rangle^{|S_i|} | \sum_{s_i \in S_i} \sigma_i(s_i) = 1\}$, což je množina smíšených strategií hráče $i \in Q$. Δ_i je vektor délky $|S_i|$. Číslo $\sigma_i(s_i)$ odpovídá pravděpodobnosti přiřazené ryzí strategii $s_i \in S_i$ ve smíšené strategii σ_i. Užitková funkce je definována jako $\pi_i(\sigma) = \sum_{s \in S} U_i(s) \prod_{i \in Q} \sigma_i(s_i)$. Smíšené Nashovo ekvilibrium je taková volba smíšených strategií (profilu) $\sigma^* \in \Delta$ ve hře Γ, pro které platí $\forall i \in Q : \forall \sigma \in \Delta : \pi_i(\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*) \geq \pi_i(\sigma_1^*, \sigma_2^*, \ldots, \sigma_{i-1}^*, \sigma_i, \sigma_{i+1}^*, \ldots, \sigma_n^*)$. Smíšené Nashovo ekvilibrium označujeme zkratkou MNE (z angl. Mixed Nash Equilibrium). Každá konečná hra má vždy alespoň jedno ekvilibrium ve smíšených strategiích [1].

R_A	1	0	0	0	0	1	$\rightarrow$	$potomek_1$	1	0	0	1	1	0
R_B	0	1	0	1	1	0		$potomek_2$	0	1	0	0	0	1

Tabulka 1: Příklad operátoru křížení a tvorba dvou potomků.

R_A	1	0	0	0	0	1	$\rightarrow$	$potomek$	1	1	0	0	0	1

Tabulka 2: Příklad operátoru mutace.

3 Genetické algoritmy

Genetické algoritmy vychází z principů Darwinovy teorie evoluce a různých teorií neodarwinismu. Podle Charlese Darwina je hlavní hnací silou evoluce tzv. přirozený výběr, díky němuž slabší jedinci umírají a ti silnější přežívají a rozmnožují se.

Genetické algoritmy pracují analogicky. Nejdříve se vytvoří konečná množina reprezentující počáteční populaci $P_1 = \{I_1, I_2, \ldots, I_N\}$, kde index 1 populace P značí generaci a prvek populace I_i značí jedince. Samotného jedince můžeme prozatím definovat jako neprázdný řetězec bitů $I_i \in \Sigma*$, kde abeceda je $\Sigma = \{0, 1\}$. Jednotlivý prvek jedince se pak nazývá gen. V každé generaci n se z P_n vybere množina nejčastěji dvou rodičů $R \subset P_n$, $|R| = 2$. Výběr probíhá pomocí selektivních operátorů turnaj či ruleta. Aplikací genetických operátorů křížení a mutace na rodiče vznikne potomek či více potomků. Tímto způsobem vytvoříme množinu potomků O_n. Další generaci přeživších vybereme pomocí selektivních operátorů z množiny, která se vytvoří sjednocením populace a potomky $P_{n+1} \subset O_n \cup P_n$, kde platí $|P_n| = |P_{n+1}|$.

Genetické operátory slouží k tvorbě nového jedince – potomka. Mezi nejčastěji používané operátory v GA jsou křížení a mutace. Nejdříve se oba rodiče zduplikují, tyto kopie budou představovat nové dva potomky. Křížením se myslí výměna části genetické informace mezi duplikáty, viz tabulka. Ke křížení potomků dochází s určitou pravděpodobností.

Formálně můžeme operaci jednobodového křížení definovat vztahem

$$potomek = \begin{cases} xy & \text{když } r \leq p_c \\ R_A & \text{když } r > p_c \end{cases} \tag{1}$$

kde p_c je pravděpodobnost, že nastane křížení. Hodnota r značí náhodné číslo, R_A a R_B jsou rodiče přičemž xy je nově vytvořený řetězec. Ten vznikl pomocí operace konkatenace mezi částí řetězce x z R_A a částí řetězce y z R_B. Z řetězce $R_A = a_1 a_2 \ldots a_n, a_i \in \Sigma$ pro $i = 1, \ldots, n$ získáme řetězec $x = a_1 \ldots a_k$, kde $1 \leq k < n$. Analogicky z řetězce $R_B = b_1 b_2 \ldots b_n, b_i \in \Sigma$ pro $i = 1, \ldots, n$ získáme řetězec $y = b_k \ldots a_n$, kde $1 < k \leq n$. Konkatenací x s y získáme kříženého jedince.

Aby byla zachována pestrost jedinců, použije se na potomky operátor mutace. Mutace mají velký vliv na výsledného jedince. Nepovolíme-li je, potom algoritmus nemusí konvergovat. Tento fakt zároveň platí pro křížení. Aplikace operátoru mutace je znázorněna v tabulce 2.

Formálně můžeme mutaci definovat vztahem

$$potomek = \begin{cases} M & \text{když } r \leq p_m \\ R_A & \text{když } r > p_m \end{cases} \tag{2}$$

kde p_m je pravděpodobnost křížení, r značí náhodné číslo, R_A představuje rodiče a M je mutovaný jedinec. V řetězci $R_A = a_1 a_2 \ldots a_n$ vybereme symbol a_j s náhodným indexem $j, 1 \leq j \leq n$. Potom nový symbol je vytvořen podle $c_j \in \Sigma \setminus \{a_j\}$. Mutovaný jedinec potom odpovídá řetězci $M = a_1 \ldots a_{j-1} c_j a_{j+1} \ldots a_n$.

Pro vybrání nových množin, ať už rodičů $R \subset P_n$ nebo nové populace $P_{n+1} \subset O \cup P_n$, musíme každému jedinci z P_n a O přiřadit tzv. fitness hodnotu. Fitness hodnota nám udává kvalitu jedince – kandidátního řešení.

Fitness funkce se implementují různými způsoby. Záleží totiž na problému, který chceme vyřešit. Rovněž záleží zda-li chceme fitness hodnotu v průběhu evoluce maximalizovat anebo minimalizovat.

K výběru jedinců z množiny nám poslouží operátory turnaj nebo ruleta. Při turnaji se nejdříve vyberou dva náhodní jedinci, ale může jich být více. V turnaji vyhraje ten, který má větší fitness hodnotu a je vybrán do nové množiny. Selekce pomocí rulety probíhá takovým způsobem, že se každému jedinci vypočte tzv. podíl na ruletě. Ten může být určen podle vzorce:

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i} \tag{3}$$

kde f_i je fitness hodnota jedince, $\sum_{i=1}^{N} f_i$ je suma fitness hodnot jedinců. Podíly vytvoří množinu intervalů: $I = \{I_1, I_2, \ldots, I_N\}$, kde $I_K = \langle \sum_{i=1}^{K-1} p_i, \sum_{i=1}^{K} p_i)$ pro odpovídající indexy $K \in 1, 2, \ldots, N$. Tyto intervaly se nepřekrývají a jejich sjednocení nám dá množinu reálných čísel v intervalu $\langle 0, 1)$. Mějme náhodnou hodnotu r, potom pro výsledného jedince s indexem n platí: $r \in I_n$.

Celá tato kapitola byla čerpána z [2].

4 Použití GA ve vyhledání ryzího a smíšeného ekvilibria

Problémem při používaní GA je definování jedince (chromozómu) a jeho ohodnocení. Autoři článku [3] použili bitové kódovaní chromozomu. Každou strategii s_j^* kódovali na 7 bitech, chromozom tedy obsahoval $7 * N$ bitů[1]. Muselo se provádět zakódovaní a rozkódování strategií. Tato vlastnost může mít negativní vliv na rychlost programu. Vhodnější je zakódovat strategii s_j^* pomocí datového typu integer.

Fitness funkce $D(s^*) : S \to \mathbb{R}^+$ nám udává vzdálenost mezi strategickým profilem s^* a ekvilibriem. Lze jej definovat jako:

$$D(s^*) = \sum_{i=1}^{N} [\max_{s_i \in S_i} U(s_1^*, s_2^*, \ldots, s_{i-1}^*, s_i, s_{i+1}^*, \ldots, s_N^*) - U(s^*)] \tag{4}$$

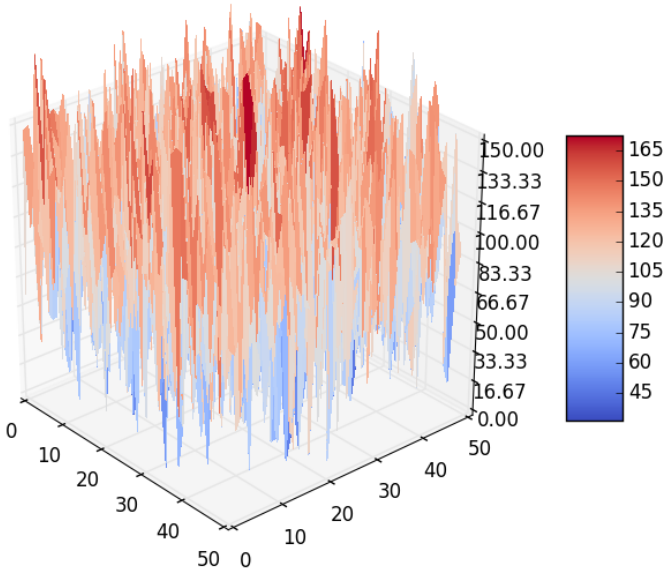
Theorem 1 *Funkce D nabývá kladných hodnot, když s^* není Nashovo ekvilibrium. Je-li výsledek D roven 0, potom s^* je Nashovo ekvilibrium [3].*

V průběhu evoluce se tedy snažíme fitness hodnotu jedinců minimalizovat.

Pro vyhledání smíšeného Nashova ekvilibria můžeme vycházet z článku [4]. Chromozóm zakódovali vektorem reálných hodnot, reprezentující profil $\sigma^* = (\sigma_1^*, \sigma_2^*, \ldots \sigma_N^*)$, kde σ_i^* je pravděpodobnostní vektor hráče i. Fitness funkce odpovídala výše uvedené rovnici s tím rozdílem, že ji aplikovali pro hru o smíšených strategiích. Protože jsme změnili kódovaní genů na reálná čísla, musíme změnit i genetické operátory křížení a mutace. Mutace může být chápana jako součet vektoru chromozomu s vektorem, jehož hodnoty genů odpovídají Gaussovskému rozložení pravděpodobnosti. Křížením může vzniknout jedinec, jehož hodnoty genu odpovídají průměru daného genu obou rodičů. Více informací je dostupných v textu [2].

Porovnáme-li prohledávací prostory MNE s PNE, potom prostor MNE závisí na počtu strategií a rozložení jejich pravděpodobnosti. Takže ve výsledku můžeme získat dlouhý chromozóm s velkým (a členitým) prohledávacím prostorem kandidátních smíšených ekvilibrií. Ukážeme si to na příkladě. Mějme hru deseti hráčů a každý z nich má na výběr deset různých strategií, potom chromozóm pro PNE je reprezentován deseti geny, kde každý z nich nabývá deseti různých hodnot. Pro PNE existuje tedy 10^{10} řešení. Chromozóm pro MNE obsahuje sto genů, kde každá desítka genů reprezentuje pravděpodobnostní vektor σ_k^*. Tento vektor obsahuje rozložení pravděpodobnosti v intervalu $\langle 0, 1\rangle$, viz kapitola 2. Problémem při vyhledávání MNE je velikost množiny $\langle 0, 1\rangle$ – je nespočetná. Tudíž můžeme tvrdit, že prohledávací prostor PNE je rovněž nespočetný. Evoluční algoritmy pro MNE se snaží aproximovat pravděpodobností profil tak, aby se co nejvíce blížil smíšenému ekvilibriu. Tento problém částečně řeší GA podporující koevoluci a multikriteriální GA (NSGA II), kde jednotlivá kritéria jsou hráči.

[1] Pozor na notaci v publikaci [3] strategie hráče představovala $u_i \in U$ a užitková funkce byla označována písmenem J.



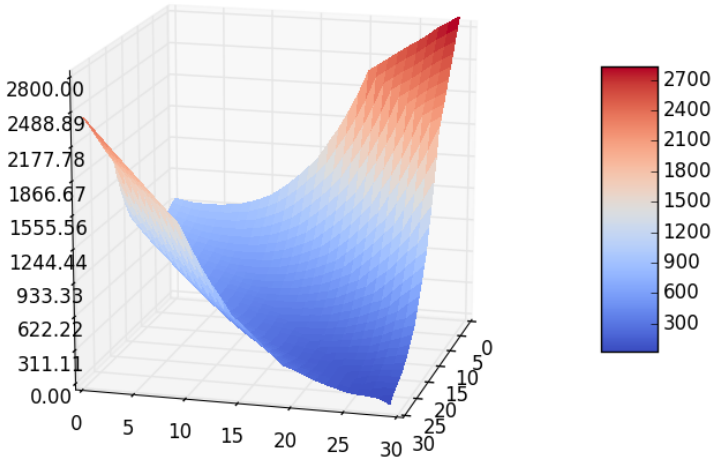
Obrázek 1: Pseudonáhodně vygenerovaný prohledávací prostor pro dva hráče, kde každý hraje 50 strategií. Graf odpovídá fitness funkci definované vztahem 4.

5 Implementace a experimentální vyhodnocení

Jako programovací jazyk byl zvolen jazyk Python. Program byl implementován podle výše zmíněných algoritmů pro vyhledání PNE. Ověření funkčnosti fitness funkce proběhlo na vězňově dilematu a na hře o třech hráčích, kde $|S_1| = 3, |S_2| = 2$ a $|S_3| = 2$, viz zdrojový kód. Nalezená ekvilibria odpovídala ekvilibriím nalezených pomocí programu Gambit [5]. Je nutné podotknout, že kódovaní buněk v matici v Gambitu je ve tvaru s_1, s_3, s_2 a v implementovaném programu je použita posloupnost argumentů popořadě, tedy: s_1, s_2, s_3, kde s_i odpovídá hrané strategii hráče $i \in Q$. Funkčnost GA (konkrétně selektivní operátory, genetické operátory) byla úspěšně otestována na základě výpisů populace a jejich fitness hodnot.

Pro otestování na větším prohledávacím prostoru byla zvolena užitková funkce pro dva hráče, kde každý z nich hraje 50 strategií. Výsledná hodnota užitkové funkce odpovídala pseudonáhodně vygenerovanému číslu v intervalu $\langle 0, 100 \rangle$ s rovnoměrným rozložením pravděpodobnosti. Následně byl zvolen jeden profil, který byl označen za dominantní – byla mu přiřazena hodnota 120 pro všechny hráče. Je zřejmé, že takto vytvořená hra má alespoň jedno ryzí Nashovo ekvilibrium. Na tuto hru bylo aplikováno 10 nezávislých běhů GA s následujícími parametry: velikost populace (počet rodičů) $|P| = 10$, počet potomků $|O| = 10$, pravděpodobnost mutace $p_m = 20\%$, počet mutovaných genů $p_g = 1$, pravděpodobnost křížení $p_k = 80\%$, počet bodů křížení $p_b = 1$ a počet generací $gen = 100$. Z 10ti běhů našel PNE pouze jeden běh, zbytek běhů skončil hodnotě blízké 0. Důvodem je špatně zvolený vyhledávací prostor, protože obsahuje hodně lokálních minim, jak můžete vidět na obrázku 1.

V článku [3] zvolili cellModel podle trhů s elektřinou, jehož graf měl jeden globální extrém tudíž jedno ryzí Nashovo ekvilibrium. V této práci představím vlastní cellModel, který by měl reprezentovat hru o N výrobcích alkoholu. Mějme obchodníka, který chce koupit L lahví kvalitního alkoholu za rozumnou cenu. Výrobci chtějí utržit, co největší částku, avšak jsou částečně omezeni cenou za plnou láhev C_i. Zároveň platí fakt, že čím více alkoholu producent vyrobí, tak tím méně je kvalitní. Tedy je vyšší pravděpodobnost, že se v něm bude nacházet methanol. Každý výrobce má možnost dát na trh cenu svého výrobku, což je strategie, kterou bude



Obrázek 2: Graf funkce $D(profit_i(s) - kvalita_i(s))$ pro vyhledání MNE v trhu s alkoholem.

daný výrobce hrát. Počet koupených lahví obchodníka závisí na ceně daného výrobce i a hrané strategii s:

$$koupi\_lahvi_i(s) = L * \frac{p(s_i - \min(s))}{\sum_{\forall j \in Q} p(s_j - \min(s))}$$

$$p(x) = \begin{cases} 1.0 - x0.1 & \text{když } x \leq 10 \\ 0.0 & \text{když } x > 10 \end{cases}$$

Podle počtu zakoupených lahví se určí výdělek a kvalita, kde ω značí faktor kvality:

$$profit_i(s) = koupi\_lahvi_i(s) * (s_i - C_i)$$

$$kavlita_i(s) = koupi\_lahvi_i(s) * \omega_i$$

Celkově tedy hodnotu užitku U_i vypočteme podle

$$U_i(s) = profit_i(s) - kvalita_i(s)$$

Na obrázku 2 je znázorněn výsledný graf funkce $D(U_i(s))$, což je výsledná ohodnocená hra dvou výrobců s alkoholem, kde každý hraje jednu z 30-ti strategií. Můžeme vidět globální minimum (na souřadnicích $[29, 29]$), které představuje Nashovo ekvilibrium. Tento model byl vygenerován těmito parametry: $L = 150$, $C = (15, 20)$ a $\omega = (2, 1)$.

Je pravděpodobné, že tento model neodpovídá realitě, avšak pro otestování GA je takto vytvořený vyhledávací prostor vhodný. Na vygenerovaném modelu proběhlo 10 nezávislých běhů s parametry $|P| = 20$, $|O| = 10$, $p_m = 20\%$, $p_g = 1$, $p_k = 80\%$, $p_b = 1$ a $gen = 100$. Úspěšně skončilo 7 běhů a bylo pro to zapotřebí v průměru 29 generací. Neúspěšné běhy se blížily k hodnotě PNE, nejčastěji byly nalezeny souřadnice $[28, 28]$, či $[27, 27]$ s fitness funkcí 30. Pro zlepšení úspěšnosti nalezení PNE je potřeba lépe nastavit parametry běhu GA (např. zvýšit počet generací).

Další experiment se prováděl na hře o deseti hráčích a zkoumal počet volání procedury cellModel. Jako model byla vybrána suma strategií, neboť model s alkoholem fungoval pouze pro správně zvolené parametry a ve hře o třech hráčích neměl vůbec PNE, což bylo zjištěno iterativní metodou. Suma strategií je definována vztahem: $U_i(s) = \sum_{\forall j \in Q} s_j$. Je zřejmé, že se ryzí Nashovo ekvilibrium se nachází na souřadnicích $(|S_1|, |S_2|, \ldots, |S_N|)$. Bylo zvoleno deset hráčů ($|Q| = 10$), kde každý z nich hraje 30 strategií ($\forall i \in Q : |S_i| = 30$). Je zřejmé, že vyhledávací prostor je dostatečně velký, existuje celkem $30^{10} = 590\,490\,000\,000\,000$ řešení. Bylo vytvořeno 15 nezávislých běhů GA s parametry $|P| = 40, |O| = 20, p_k = 80\%, p_b = 2, p_m = 20\%, p_g = 3, g$. Každý běh dopadl úspěšně, průměrný počet generací pro nalezení PNE činil 276 a průměrný počet volání funkce cellModel byl $444\,035$.

6 Závěr

Tato práce shrnula vědomosti několika článků zabývajících se evolučním přístupem k vyhledání ryzího a smíšeného Nashova ekvilibria. Na základě článku [3] byl implementován algoritmus pro vyhledání ryzího ekvilibria ve velkých prohledávacích prostorech. Experimenty bylo zjištěno, že evoluční přístup může snadno uváznout v lokálním minimu, byla-li ohodnocovací matice inicializovaná náhodně. Testování dále probíhalo na vlastním modelu trhu s alkoholem. Díky použití GA a cellModelu jsme nemuseli vygenerovat celou skórovací matici, ale pouze její část.

Pokračování tohoto projektu vidím ve dvou větvích. Za prvé zaměřil bych se na implementaci GA pro vyhledání MNE za využití multikriteriálních algoritmů. Za druhé vytvořil bych reálnější modely více hráčů, na kterých by se dalo naleznout PNE.

Reference

[1] Hrubý M., *Prezentace a skripta k předmětu Teorie her*

[2] Schwarz J., *Prezentace a skripta k předmětu Evoluční algoritmy*

[3] Beck, E.V., Cherkaoui, R., Minoia, A. and Ernst, D., *Nash equilibrium as the minimum of a function. Application to electricity markets with large number of actors*, Power Tech, 2007 IEEE Lausanne, s. 837-842, July 2007

[4] Liu W., Li J., Yue K. and Song N., *Reduction of Graphical Model and Genetic Algorithm for Computing Approximate Nash Equilibrium in Static Games,* Natural Computation, 2007. ICNC 2007. Third International Conference on , vol.3, s. 466-470, August 2007

[5] Turocy T., *Gambit: Software Tools for Game Theory*, URL: `http://gambit.sourceforge.net/`

A Epilog

Program byl úspěšně testován na operačním systému Windows 7 a školním serveru Merlin, kde program spustíte příkazem `python2 ./nash_eq.py`. Jako interpret Pythonu je potřeba zvolit Python2. Pro úpravu parametrů GA (případě i modelu) je potřeba změnit řádky v programu, konkrétně řádky 15–42.