Autor: **Pavol Palka**
e-mail: 7palka@st.fmph.uniba.sk

Téma 7. Riadenie robota

Robot je primitívny stroj, ktorý sa môže pohybovať iba na sever, juh, výhod a západ. Nemá senzory a podmienené pohyby, jeho správanie sa je dané sekvenciou pohybov, ktoré má urobiť v bludisku. Keď narazí do múra, taký pohyb je jednoducho zbytočný a neaplikovaný. Bludisko má jeden vchod, kde robot začína a jeden východ (ciel robota). Čím bližšie sa robot dostane k cieli pri čo najmenšom počte krokov, tým lepšie má skóre. Robot je jednoducho reprezentovaný ako sada pohybov:

```
(s v v v j j)
```

Bludisko môže vyzerať napr. nasledovne:

```
######
#   x#
### ##
#e  R#
######
```

kde x je východ, e je vchod, a R je momentálna pozícia robota (na ktorú sa dostane, pokiaľ bude sledovať vyše uvedenú sadu pohybov a štartuje z e). Robot sa pohybuje, dokiaľ nedosiahne východu (zvyšok riadiacej sekvencie je ignorovaný) alebo nevyčerpá pohyby. Chromozóm može vyzerať ako binárny reťazec s move (00 = v, 01 = z, 10 = s, 11 = j). Vyhodnoťte rôzne fitness funkcie.

Riešenie:

Imšpiráciu ako aj teoretické pozadie k tejto práci som čerpal s prednášok "Evolučné algoritmy" /FMPH 2000/ s knihy "Evolučné algoritmy" / Prof. Ing. Vladimír Kvasnička, DrSc, Doc. RNDr. Jiří Pospíchal, CSc, a Ing. Peter Tiňo, CSc. 2000 / a s práce Kyle Wagnera, ktorý sa zaoberal implementáciou tohoto problému v jazyku LISP /PC Scheme/.

Teretický úvod

Genetický algoritmus patrí medzi základné stochastické optimalizačné algoritmy s výraznými evolučnými črtami. Metódy numerickej matematiky používajú pri optimalizácií funkcie poznatky o jej tvare v danom bode (gradientové metódy), alebo sú schopné pomocou predchádzajúcich riešení indikovať s veľkou efektívnosťou nájdený smer minimalizácie funkcie(napr. Simplexová metóda). V biologických systémoch prebiehajúce elvolučné procesy nám poskytujú alternatívny prístup ku generovaniu náhodných bodov k hodnotám blízkym optimálnym. Evolučná stratégia sa preto stala základom genetického algoritmu, ktorý vylepšuje astochastický algoritmus natoľko, že je schopný v reálnom čase nájsť optimálne riešenie.
Darwinova teória evolúcie sa zakladá na myšlienky prirodzeného výberu. Teda prežívajú len najlepší jedinci. Reprodukcio dvoch dobrých jedincov dostávame s vysokou pravdepodobnosťou jedincov, ktorí budú dobre prispôsobení na prežitie. Pri podrobnejšej matematickej analýze sa však ukáže, že samotná reprodukcia nám nedostatočne zaručuje vzik nových pre prežitie významných vlastností. Preto je nutné zaviesť mutáciu, ktorá bude ovplyvňovať genetický materiál jedincov.
Biologická evolúcia je progresívna zmena obsahu genetickej informácie (*genotypu*) populácie v priebehu mnohých generácií.Pri evolúcii tvorí klúčový prvok schopnosť jedinca prežiť a reprodukovať sa. V teórii genetických algoritmov je reprezentovaná funkciou *fitness*. Teda je to kladné reálne číslo, priradené genetickej informácií reprezentujúcej jedinca, ktoré vyjadruje jeho schopnosť plniť v danom prostredí úlohy (v našom prípade nájsť cestu z labyrintu) a vstupovať do reprodukcie.
Ďalším významným pojmom v evolučnej terminológií je jedinec. V „umelom živote" abstrahujeme od jeho fenotypu a reprezentujeme ho chromozómom, ktorý lineárnym spôsobom reprezentuje informačný obsah jednotlivca. Teda hovoríme o populácií chromozómov, ktoré sa reprodukujús pravdepodobnosťou úmernou ich fitness, pričom integrálnou súčasťou tejto reprodukcie sú mutácie. Mutácie vnášajú do chromozómov novú informáciu, ktorá môže zvýšiť fitness chromozómov vznikajúcich reprodukciou pôvodných chromozómov. Nové chromozómy vytesňujú z populácie chromozómy s malou fitness. Tento prces sa v danej populácii neustále opakuje. Teda po určitom čase sa s veľkou pravdepodobnosťou objavia chromozómy s novými vlastnosťami, ktoré zvyšujú ich fitness.
Keďže populácia je zložená z jedincov reprezentovaných chromozómami, môžeme ju teda definovať ako množinu týchto chromozómov:

$$P = \{\alpha_1, \alpha_2, \ldots, \alpha_p\} \subseteq \{a, b, \ldots\}^k$$

Populácia P obsahuje p chromozómov, ktoré sú realizované ako reťazce dĺžky *k* zložené zo znakov a, b, … . Obvykle sa tieto znaky rovnajú 0 a 1., chromozómy sú potom binárne reťazce, $P \subseteq \{0, 1, \ldots\}^k$ (aj v našom prípade je táto konvencia svojim spôsobom zachovaná, pretože aj napriek tomu, že chromozómom kódujeme smer (V, Z, S, J), pouzívame na jeho kódovanie chromozómy ako reťazce 0 a 1). Každý chromozóm je ohodnotený fitness, ktorá sa interpreturje ako zobrazenie chromozómu na kladné reálne čísla

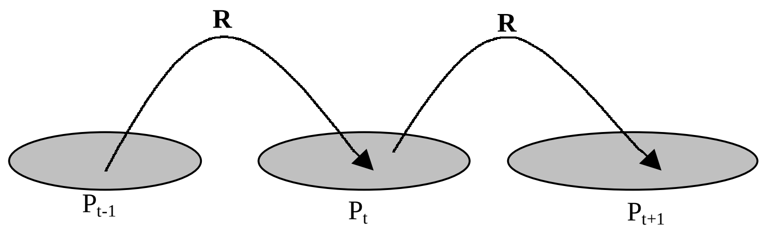
$$F: P \rightarrow R_+$$

Proces reprodukcie sa začína tým, že sa z populácie kvázináhodne vyberú dva chromozómy v závislosti od ich fitness. Pod reprodukciou rozumieme proces, v ktorom dva pôvodné chromozómy reprodukujú, pričom vzniknú dva nové chromozómy

$$(\alpha'_1, \alpha'_2) = O_{repro}(\alpha_1, \alpha_2)$$

Operátor reprodokcie obsahuje dve časti: *mutáciu* a *kríženie*. Vzhladom na náhodnosť obydvoch operácií, má aj operátor reprodukcie stochastický charakter.
Evolúciu môžeme chápať aj ako zmenu populácie realizujúcu sa v čase. Nech P_t je populácia chromozómov v čase t, potom je populácia Purčená rekurentne pomocou „reprodukčného operátora“ operátora R

$$P_{t+1} = R(P_t)$$



Evolučný algoritmus je založený na simultánnej optimalizácií celej populácie chromozómov, zatiaľ čo numerické metódy optimalizujú len jeden objekt, čo v podstate vo väčšine prípadov zvyšuje efektivitu ale zároveň to znižuje šance nájsť globálne minimum. Z čoho nám vyplýva aj náhodný resp. kvázináhodný výber chromozómov, ktoré vstupujú do reprodukcie.

Základné pojmy genetického algoritmu – populácia a fitness chromozómov

Základné princípy genetického algoritmu sú určitou špecifikáciou všeobecných princípov evolúcie a jej algoritmizácie. Pod chromozómom α teda rozumieme binárny vektor fixnej dĺžky *k*

$$\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_k) \in \{0,1\}^k$$

Populácia P obsahuje chromozómy – binárne vektory α

$$P = \{\alpha_1, \alpha_2, \ldots \alpha_p\}$$

Kde p je kardinalita populácie P, vyjadruje počet chromozómov v P, p=|P|. Nech *f* je účelová funkcia definovaná nad množinou binárnych vektorov dĺžky *k*

$$F : \{0,1\}^k \rightarrow R$$

Ktorá ohodnotí každý binárny vektor $\alpha \in \{0,1\}^k$ reálnym číslom. Našou úlohou bude nájsť globálne minimum tejto funkcie nad množinou $\{0,1\}^k$

$$\alpha_{opt} = \arg \min_{\alpha \in \{0,1\}^k} f(\alpha)$$

Funkcia *f* „reprezentuje prostredie", v ktorom existujú chromozómy populácie. V biologickej terminológii chromozóm a reprezentuje *genotyp* organizmu, zatiaľ čo funkčná hodnota *f*(α) reprezentuje jeho *fenotyp*. Mierou úspešnosti chromozómu je jeho funkčná hodnota. Pretože hľadáme minimum účelovej funkcie, chromozóm je tým lepší, čím je jeho funkčná hodnota, teda genotyp , menšia. Na základe týchto úvah môžeme zaviesť ohodnotenie fitness chromozómov ako zobraaenie, ktoré vyhohuje nasledujúcej podmienke

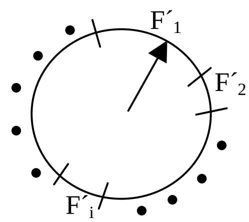
$$\forall\ \alpha_1,\alpha_2 \in \boldsymbol{P} : \boldsymbol{f}(\alpha_1) \leq\ \boldsymbol{f}(\alpha_2) \Rightarrow \mathbf{F}(\alpha_1) \geq\ \mathbf{F}(\alpha_2) \geq 0$$

Z praktických dôvodov je vhodné, aby numerické hodnoty fitness boli z otvoreného intervalu (0, 1), preto sa zavádza tzv. *Renormalizovaná fitness*

$$\mathbf{F}'(\alpha)=\frac{\mathbf{F}'(\alpha)}{\sum\limits_{\alpha'\in P} \mathbf{F}(\alpha')}$$

$$\sum_{\alpha'\in P} \mathbf{F}(\alpha)=1$$

kde 0<F'(a) <1 pre každé a e P.
(V našom algoritme sa táto normalizácia vypočítava pri inicilizácii rulety. Teda fitness môže vracať aj nenormalizované hodnoty, poľa pomerov týchto hodnôt sa priradia jednotlivým chromozómom rôzne velké políčka rulety.)
Pomocou renormalizovaných fitmess môžeme jednoducho realizovať kvázináhodnosť výberu chromozómov (čo patrí medzi základné imperatívy evolučných algoritmov) pomocou pojmu ruleta.



Ruleta –grafická reprezentácia

Chromozóm s indexom *I* sa vyberie (kvázináhodne), ak pre náhodne generované číslo random z intervalu [0,1) s rovnomernoou distribúciou pravdepodobnosti platí

$$\sum_{k=1}^{i-1} \mathbf{F}'_k\mathbf{=1} \leq \mathbf{random} < \sum_{k=1}^{i} \mathbf{F}'_k\mathbf{=1}$$

Kríženie chromozómov

Operátor mutácie stochasticky transformuje binárny vektor a na nový binárny vektor a´ , pričom stochastičnosť tohoto procesu je určená pravdepodobnosťou Pmut

$$\alpha' = \boldsymbol{O}_{\mathbf{mut}}(\alpha)$$

Kde α a α´ sú dva binárne vektory rovnakej dĺžky *kn*

$$\alpha' = (\ \alpha_1, \alpha_2, \ldots , \alpha_{kn}\ )\ \ \mathbf{a}\ \ \alpha' = (\ \alpha'_1, \alpha'_2, \ldots , \alpha'_{kn}\ )$$

Kde jednotlivé komponenty sú určené nasledovne

$$\alpha'_i = \begin{cases} 1-\alpha_i\ (\text{ pre random} < {}_{\text{Pmut}}\ ) \\ \alpha_i\ (\text{ v ostatných prípadoch} \end{cases}$$

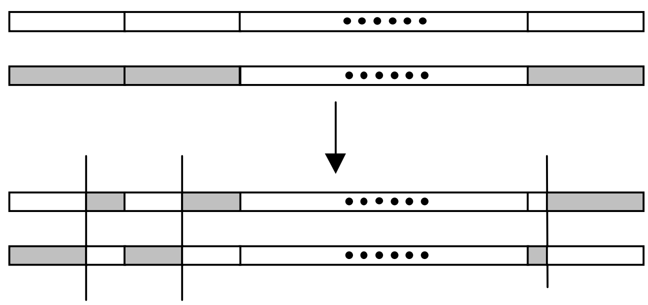
kde *random* je náhodné číslo z intervalu [0,1) generované rovnomernou distribúciou pravdepodobnosti. Pričom pravdepodobnosť P_{mut} určuje stochastičnosť operátora mutácie, v limitnom prípade ak $P_{mut} \to 0$, potom operátor O_{mut} nemení binárny vektor

$$\lim_{P_{mut} \to 0} \boldsymbol{O}_{\mathbf{mut}}(\alpha) = \alpha$$

Majme dva chromozómy $\alpha,\beta \in \{0,1\}^k$, operácia kríženia sa bude formálne interpretovať ako operátor – zobrazenie, ktorý tejto dvojici chromozómov priradí dva nové chromozómy s rovnakou dĺžkou ako pôvodné

$$(\alpha',\beta') = \boldsymbol{O}_{cross}(\alpha,\beta)$$

Tento operátor má, podobne ako operátor mutácie, stochastický charalter. Táto stochastičnocť je založená na tom, že pri aplikovaní operátora kríženia na dvojicu chromozómov z populácie sa generuje náhodne tzv. bod (body) kríženia a potom je už aplikácia kríženia úplne deterministická. V prípade že chromozóm kóduje *n* reálnych premenných pomocou binárnych reťazcov dĺžky *k*, operácia kríženia sa môže aplikovať zvlášť pre každý binárny podreťazec reprezentujúci jednotlivú premennú účelovej funkcie.



Schématiské znázornenie kríženia chromozómov dĺžky *kn*

Pomocou operátorov mutácie a kríženia môžeme pristúpiť ku konštrukcii operátora reprodukcie. Nech $\alpha,\beta \in P$ sú dva chromozómy z populácie *P*, pomocou operátora reprodukcie zostrojíme z týchto dvoch „rodičovských“ chromozómov dva nové chromozómy „potomkov“, formálne $(\alpha',\beta') = O_{repro}(\alpha,\beta)$. V prvom kroku zostrojíme pomocou kríženia dva nové chromozómy

$$(\hat{\alpha},\hat{\beta}) = \boldsymbol{O}_{cross}(\alpha, \beta)$$

V nasledujúcom kroku sa tieto dva nové chromozómy modifikujú mutáciou na výsledné chromozómy – potomkov

$$\alpha' = \boldsymbol{O}_{mut}(\hat{\alpha}) \quad \text{a} \quad \beta' = \boldsymbol{O}_{mut}(\hat{\beta})$$

Kríženie patrí medzi základné črty genetického algoritmu, ak ho odstránime, potom sa genetizký algoritmus zredukuje na niečo podobné horolezeckému algoritmu. Používanie kríženia odlišuje genetický algoritmus od ostatných stochastických algoritmov evolučných algoritmov, ktoré v rámci populácie objektov – riešení – chromozómov tiež používajú reprodokciu založenú na ich fitness a aplikujú mutácie na jednotlivé objekty.

Kombinátorické optimalizačné problémy

Jeden z prvých veľkých úspechov evolučných algoritmov bolo ich použitie na riešenie ťažkých kombinátorických problémov, ktorých čas CPU rastol buď faktoriálovo alebo exponenciálne s rastom dimenzie problému (sú to tzv NP-ťažké problémy). Problém robota patrí medzi tieto problémy. Jej formalizácia môže vyzerať nasledovne.
V nasledujúcich odsekoch budú definované hlavne numerické vlastnosti funkcie *f* ktoré majú demonštrovať možnosť využitia evolučných algoritmov pri riešení kombinátorických úloh. Napriek tomu, že nieje možné použiť numerické metódy na optimalizáciu výpočtu funkcie *f* , pri praktickejšom náhlade na problém zistíme, že je vhodné použiť

dynamické programovanie. Teda napriek tomu, že problém robota ilustruje určité črty kombinátorických problémov, sám *ťažkým kombinátorickým problémom nieje.*
Formalizácia problému môže vyzerať nasledovne.

Nech množina R_{set}={ E, W, S, N } je množina, ktorá obsahuje možné smery pohybu robota. Jej priamy súčin R_{set}^N obsahuje N-tice

$$\pi = (\pi_1, \pi_2, \ldots, \pi_N) \in R_{set}^N = R_{set} \times R_{set} \times \ldots \times R_{set}$$

Jednotlivé zložky tohto výrazu sú rôzne smery pohybu. Funkcia

$$F: R_{set}^N \rightarrow R$$

Zobrazuje N- násobný kartézsky produkt R_{set}^N na celé čísla. Funkciu f môžeme chápať ako simuláciu chodu robota, ktorej výsledok je počet krokov po ktorých robot našiel cestu von z labyrintu. Analógia optimalizačného problému má potom tvar

$$\pi_{opt} = \arg\min_{\pi \in R_{set}^N} f(\pi)$$

V dôsledku toho, že kardinalita množiny R_{set}^N je 4^N, optimalizačný s exponenciálnym priestorom riešení.. Vyplýva to z toho, že množina R_{set}^N musí obsahovať všetky postupnosti krov ktoré by potenciálne mohli viesť k cielu. Pri bludisku o rozmere m x n je najdlhšie možné riešanie dlhé (m x n)/2. Kedže veľkosť bludiska určuje veľkosť problému, tak pri probléme velkosti m x n má množina R_{set}^N veľkosť

$$|R_{set}^N| = 4^{(m \times n)/2} = (4^{1/2})^{(m \times n)} = 2^{(m \times n)}$$

Na výpočet globálneho minima neexistujú žiadne numerické metódy, ktoré by vedeli lokálne predpokladať tvar funkcie f a tak urýchliť výpočet minima. Je to hlavne preto, že v ktoromkoľvek kroku simulácie nevieme predpokladať či nasledujúci krok bude viesť do steny alebo nie.

Popis algoritmu

Robot je implementovaný ako klasický genetický algoritmus. Vzhľadom na to, že optimálna hodnota, teda výsledok algoritmu nieje vopred známa, je beh programu ohraničený iba maximálnym počtom iterácii. Program je v podstate realizáciou operátora O_{repro}., ktorý je zložený s operátora kríženia a operátora mutácie. Nasleduje jeho pseudopascalovský kód

```
procedure GenAlg_Robot(input tmax,bludisko)
begin
      t:=0;
      P:={náhodne vygenerovaná populácia chromozómov};
      while (t<tmax) do
      begin
            t=t+1;
            odsimuluj beh robota pre každý chromozóm;
            while |Q|<|P| do
            begin
                  vyber ruletou dva chromozómy α1,α2∈P;
                  if random<Prepro then
                  Reproduction(α1,α2,α'1,α'2)
                  else
                  begin α'1:=α1 ; α'2:=α2 ; end;
                  Q:=Q∪{α'1,α'2};
            end;
            P:=Q;
```

```
            if (α1opt je horsí ako najlepší chromozóm z P)
                  α1opt,:= najlepší chromozóm z P;
      end;
end

procedure Reproduction(inputr:α1,α2,output:α'1,α'2)
begin
      Crossover(α1,α2,α"1,α"2)
      Mutation_Bin(α"1,α'1)
      Mutation_Bin(α"2,α'2)
end;

procedure Crossover(input: α,β; output: α',β');
begin cross_point:=1+random(k-1)
      for i:=1 to cross_point do
      begin α'i=α i ; β'i =βi end;
      for i:= cross_point+1 to k do
      begin α'i=βi ; β'i=α i end;
end;

procedure Mutation_Bin(input:α; output:α')
begin for i:=1 to kn do
      do random<Pmut then
            α'i=1-αi;
      else
            α'i=αi;
end;
```

Popis fitness funkcií

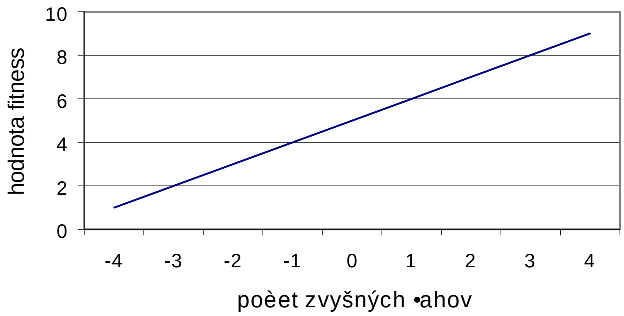
Kedže fitness funkcia je jediným cieleným prvkom v inak stochastickom algoritme, je jej správny výber základným predpokladom pre správny beh programu.
Pri návrhu fitness funkcie treba pamätať na všetky aspekty, ktoré ovplyvňujú hľadané minimum. V našom prípade je nevyhnutné zohladniť dva aspkty: -či robot nasiel labyrint, resp. ako daleko od východu z labyrintu skoncil
- ako rýchlo prisiel robot k východu

Význam prvého bodu je zrejmý. Pokiaľ nevieme posúdiť, ako blízko k východu sa robot dostal, nevieme „usmernit“ evolúciu požadovaným smerom. Dôležitosť druhého bodu ukazuje fitness funkcia 1, ktorá nájde cestu von a dĺžku cesty k východu mení stochasticky.Nasleduje popis jednotlivých fitness funkcií.
Vo všetkých fitness funkciách pokiaľ nebude uvedené inak, bude platiť nasledovné:

body za vzdialenosť = najdlhšia vzdialenosť – aktuálna vzdialenosť + 1

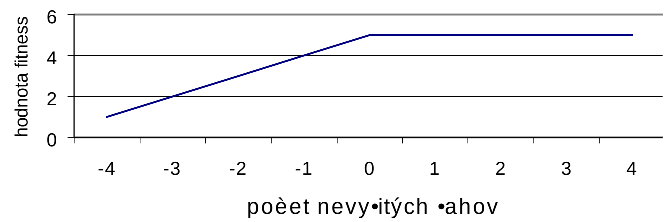
Fitness0

Táto fitness funkcia priradí chromozómu súčet vzdialenosti od východu do akej sa robot dostal a počtu ťahov, ktoré robotovi ostali. Graf fitness funkcie ilustruje fitness funkciu v labyrinte v ktorom je najväčšia vzdialenosť od východu štyri kroky a po nájdení východu ostanú maximálne štyri nevyužité ťahy.
Táto fitness funkcia je triviálnym príkladom zohladnenia obydvoch požadovaných aspektov.



Fitness0 = body za vzdialenosť + zvyšok chromozómu

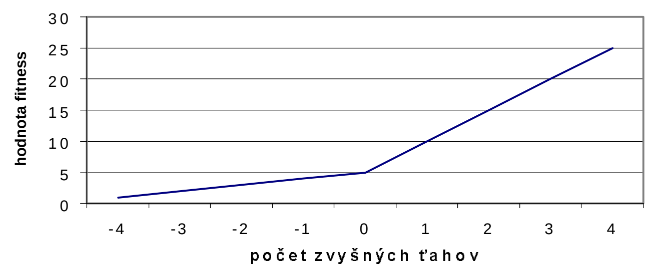
Fitness1



Táto fitness funkcia slúži na ilustráciu toho, aké dôležité je zohladniť všetky parametre. Ako vidno všetky chromozómy ktoré dovedú robota k východu sú s pohladu fitness rovnako dobré, preto táto funkcia dosahuje najhoršie výsledky.

Fitness1 = body za vzdialenosť

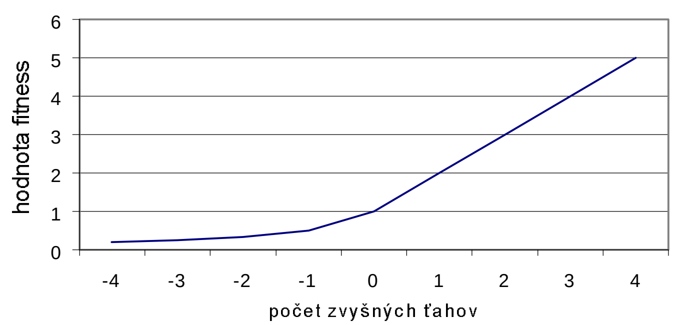
Fitness2



Fitness funkcia zavádza markantnejšie rozdiely medzi chromozómami, ktoré nevedú k východu a chromozómami ktoré vedú k východu skôr ako vyčerpajú všetky možné kroky.Nasleduje formalizácia fitness funkcie:

Fitness2 = (body za vzdialenosť) * (zvyšok chromozómu + 1)

Fitness3



Fitness3 funkcia zavádza tým väčšie rozdiely medzi chromozómami, čím sú bližšie k východu. Teda urýchluje konvergenciu k správnemu riešeniu. Medzi chromozómami, ktoré vedú k východu sú len lineárne rozdiely.

$$\textbf{Fitness3} = \frac{\textbf{zvyšok chromozómu +1}}{\textbf{vzdialenosť +1}}$$

Fitness4

Fitness funkcia 4 je obdobou predošlej fitness funkcie, ale namiesto skutočnej vzdialenosti od východu z labyrintu použije jej geometrickú vzdialenosť. Dosiahneme tým odstránenie zdĺhavého predvýpočtu na úkor toho, že chromozóm ktorý je v bludisku síce viac vzdialený od východu má vyžšiu fitness, ako ten, od ktorého síce vedie kratšia cesta ale má väčšiu geometrickú vzdialonosť. Čo môže v niektorých labyrintoch oddialiť nájdenie riešenia.

Fitness5

Táto fitness funkcia je obdobou fitness3. Pokiaľ chromozóm nevedie k východu, použije sa ako vzdialenosť minimálna vzdialenosť (geometrická), do akej sa robot dostal namiesto vzdialenosti, v ktorej robot skončil.

$$\textbf{Fitness5} = \frac{\textbf{zvyšok chromozómu +1}}{\textbf{min. vzdialenosť +1}}$$

Fitness6

Fitness6 je to isté ako fitness5, ale ako minimálnu vzdialenosť berie skutočnú vzdialenosť v bludisku, nie geometrickú vzdialenosť.

Výsledky:

Na výpočet nasledujúcich výsledkov, bola evolúcia zopakovaná sto krát a ako výsledky sú uvedené priemerné hodnoty.
Labyrint použitý v evolúcií je na Obr.1 .

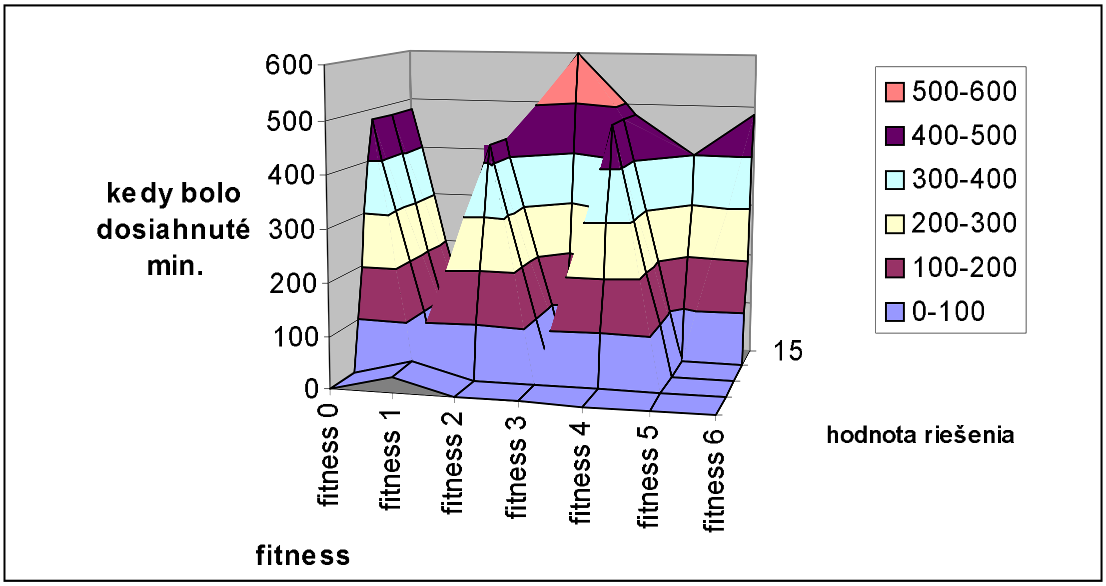
```
############
# x  #     #
# #    #   #
#    #     #
#    #     #
## #### ####
#    #     #
# #  #     #
#      # e #
#    #     #
#    #     #
############
```

Obr.1

Tabulka 1 porovnáva rýchlosť hodnotu nájdeného minima pre všekky fitness funkcie:

Fitness	populácia výskytu min.	nájdené minimum
Fitness 0	479	16
Fitness 1	31	27
Fitness 2	434	16
Fitness 3	600	15
Fitness 4	479	16
Fitness 5	399	15
Fitness 6	482	15

Graf 1 zobrazuje hodnoty z predchádzajúcej tabulky



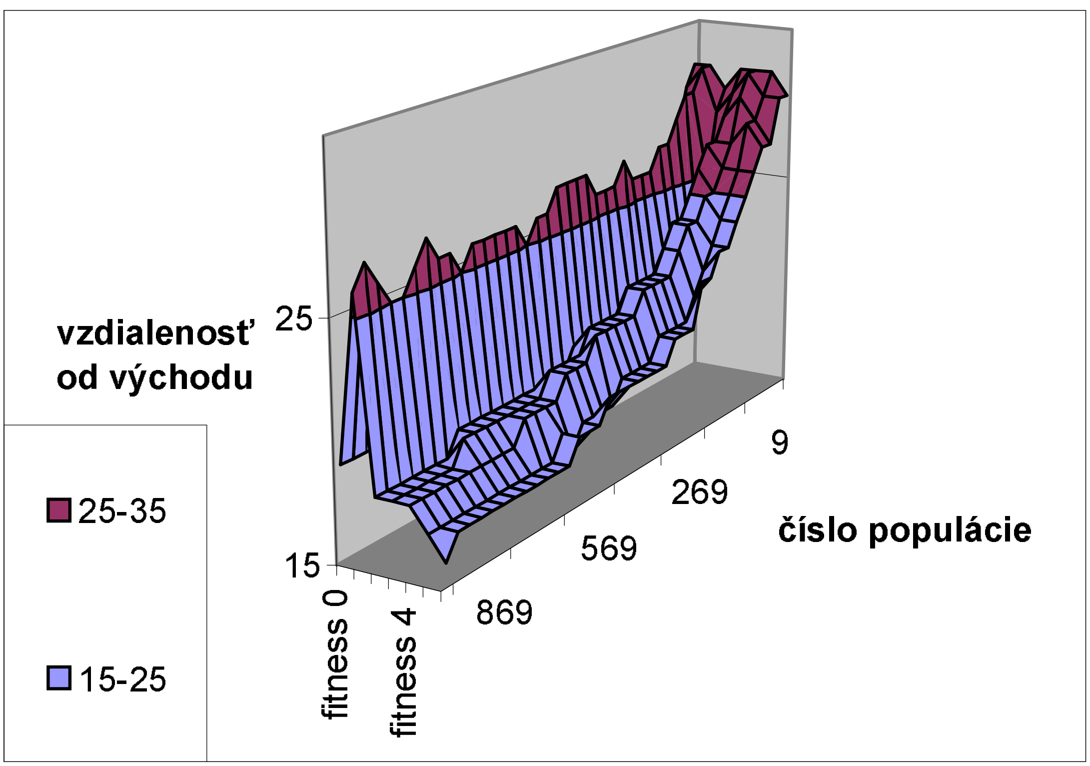
Tabulka 2 znázorňuje vývoj najlepších chromozómov v populácií pri rôznych fitness funkciách.

fitness populácia	fitness 0	fitness 1	fitness 2	fitness 3	fitness 4	fitness 5	fitness 6
9	30	30	29	30	30	30	29
19	29	30	28	29	30	29	29
29	28	29	26	28	30	28	28
39	27	29	25	28	29	28	27
49	26	28	23	28	28	28	27
59	26	28	23	27	28	28	27
69	25	29	22	27	27	28	27
79	25	28	22	27	27	27	26
89	25	28	22	27	26	26	26
99	24	27	22	27	26	25	25
109	24	27	22	26	25	25	25
119	24	27	22	26	25	25	25
129	23	27	21	26	25	24	25
139	24	27	22	26	25	24	24
149	23	27	22	25	25	24	24
159	23	26	21	25	24	23	24
169	23	26	21	24	24	23	24
179	23	26	22	24	24	23	23
189	23	26	21	24	24	22	24
199	23	26	21	24	24	22	23
209	23	26	21	24	24	22	23
219	23	26	21	24	23	21	23
229	23	27	21	24	23	21	22
239	23	26	21	23	23	21	22
249	22	26	21	23	22	21	22

259	22	26	21	23	22	20	22
269	23	27	21	22	22	20	22
279	23	26	21	22	22	20	21
289	23	26	21	22	22	20	21
299	22	26	21	22	22	20	20
309	23	26	20	22	22	20	20
319	22	27	20	22	22	20	20
329	22	26	20	22	22	20	20
339	22	26	20	22	22	19	20
349	22	27	21	22	22	20	20
359	22	26	20	22	22	20	20
369	22	27	20	22	22	19	20
379	22	26	20	22	21	19	19
389	22	27	20	21	21	19	19
399	22	26	20	21	21	19	19
409	22	27	20	21	21	19	20
419	22	27	20	21	21	19	19
429	22	27	20	21	21	19	19
439	22	28	20	21	21	19	19
449	22	27	20	21	21	19	19
459	22	27	20	21	21	19	19
469	22	27	20	21	21	19	19
479	22	27	20	21	21	19	19
489	21	26	19	20	21	19	19
499	21	26	19	20	21	18	19
509	21	26	19	20	21	18	19
519	21	25	19	20	20	18	19
529	22	26	19	20	20	18	19
539	21	26	19	20	20	18	19
549	21	26	19	20	20	18	18
559	21	26	19	20	20	18	19
569	20	25	19	20	20	18	19
579	20	26	19	20	20	18	18
589	20	26	19	20	20	18	19
599	20	26	19	20	20	18	18
609	20	26	19	19	20	18	18
619	20	25	19	19	19	18	18
629	20	26	19	19	19	17	18
639	20	25	19	19	19	18	18
649	20	26	19	19	19	17	18
659	20	26	19	19	19	18	18
669	20	26	19	19	19	17	18
679	19	26	19	19	19	17	18
689	20	26	19	19	19	17	17
699	20	26	19	19	19	17	17
709	20	26	19	19	19	17	17
719	20	26	19	19	19	17	17
729	20	26	19	19	19	17	17
739	20	26	19	19	18	17	17

749	20	25	19	19	18	17	17
759	20	26	19	19	18	17	17
769	19	25	19	19	18	17	17
779	19	26	19	19	18	17	17
789	20	26	19	19	18	17	17
799	20	26	19	19	18	17	17
809	19	26	19	19	18	17	17
819	19	25	19	18	18	17	17
829	19	25	18	19	18	17	17
839	19	27	18	19	18	17	17
849	19	26	18	18	18	17	17
859	19	26	18	19	18	17	17
869	19	26	18	18	18	17	17
879	19	26	18	18	18	17	17
889	19	26	18	18	18	17	17
899	19	25	18	18	18	17	17
909	19	25	18	18	18	17	17
919	19	26	18	18	18	17	17
929	19	25	18	18	18	17	17
939	19	26	18	18	18	17	17
949	19	25	18	18	18	17	17
959	19	26	18	18	18	17	17
969	19	26	18	18	18	17	17
979	19	26	18	18	18	17	17
989	19	27	18	18	18	17	17
999	19	26	18	18	18	17	16

Graf 2 znázorňuje výsledky z tabulky 2

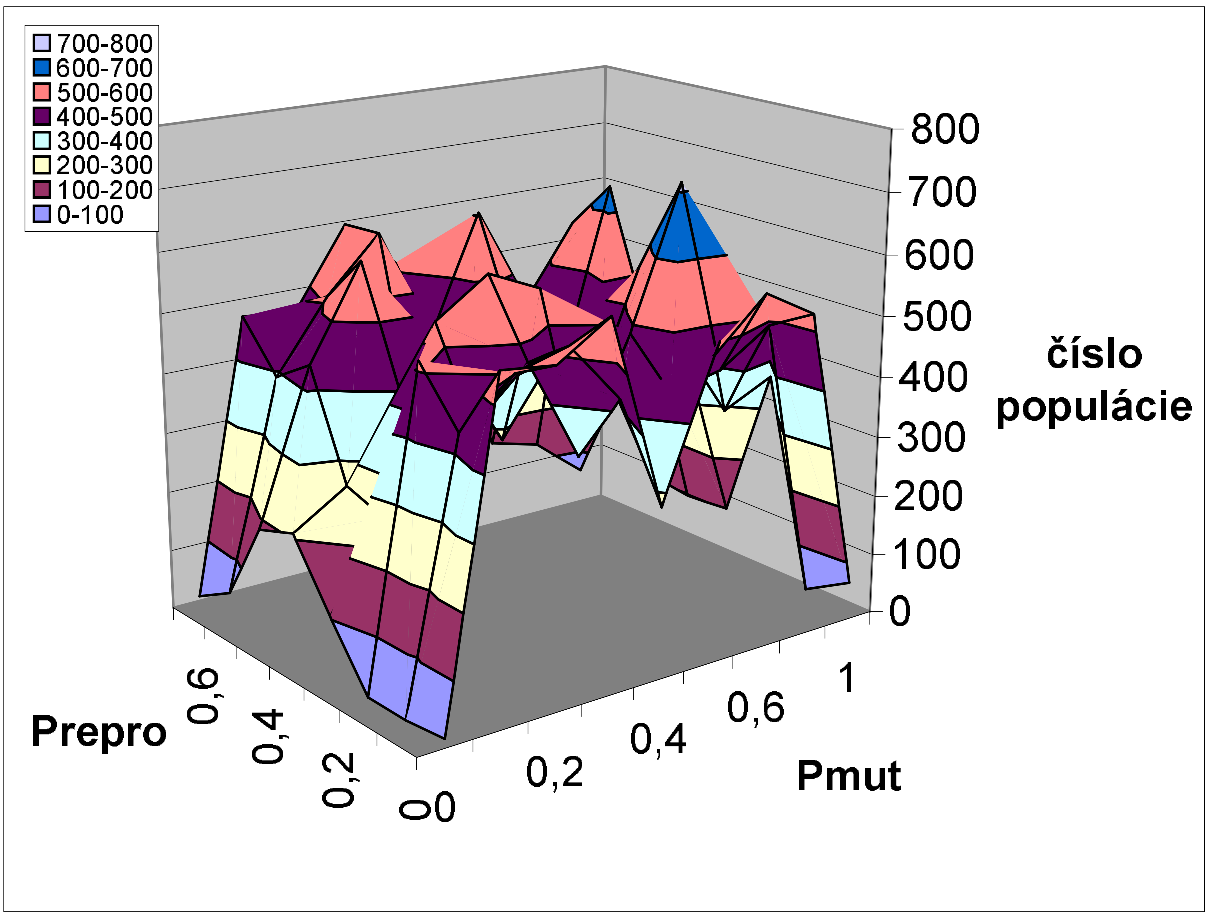


Ďalšie významné parametre, ktoré ovlyvňijú evolúciu sú pravdepodobnosť mutácie a pravdepodobnosť reprodukcie. Ich vplyv na nájdenie minima znázorňujú nasledovné tabulky.

Tabulka 3 ukazuje vplyv pravdepodobnosti na rýchlosť nájdenia minima.

Pmut\Prepro	0	0,1	0,2	0,3	0,4	0,5	0,6	0,8
0	15	7	7	104	208	185	46	8
0,1	549	430	515	242	264	436	392	470
0,2	538	501	395	535	406	589	510	462
0,3	594	526	476	594	431	299	599	594
0,4	280	424	305	555	279	342	406	514
0,5	501	415	487	348	400	422	605	421
0,6	579	433	714	558	432	344	437	153
0,8	530	489	323	404	424	640	559	130
1	63	19	362	103	97	108	232	61

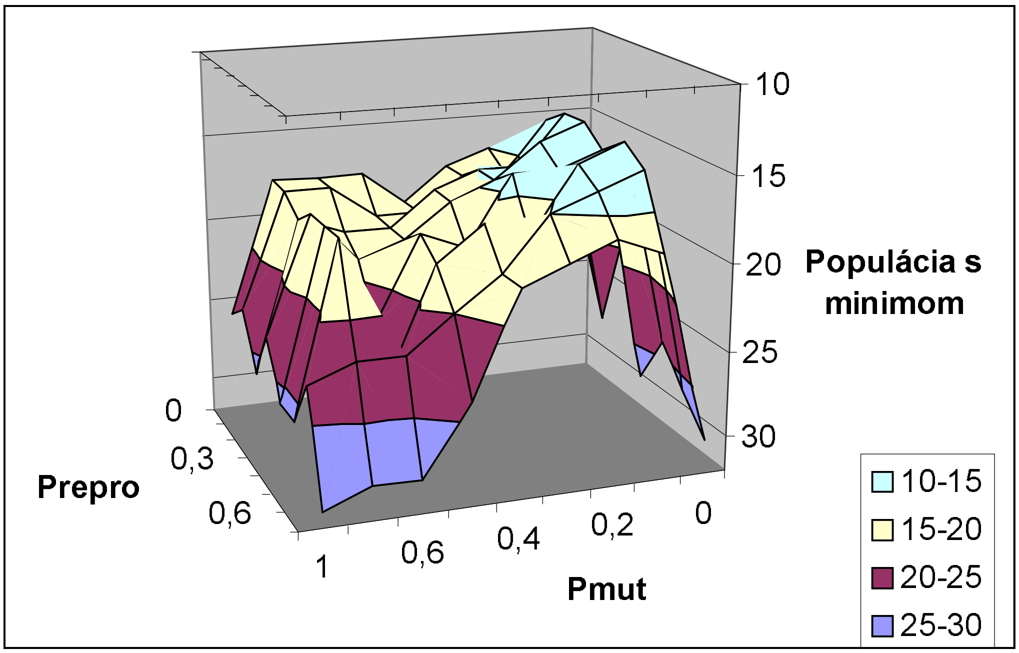
Graf 3



Tabulka 4 ukazuje vplyv pravdepodobností na nájdené minimum.

Pmut\Prepro	0	0,1	0,2	0,3	0,4	0,5	0,6	0,8
0	22	23	27	21	29	26	28	30
0,1	18	15	14	14	15	14	15	19
0,2	17	18	17	15	16	15	16	18
0,3	18	18	17	17	16	19	17	19
0,4	20	20	18	18	20	18	20	20
0,5	18	20	20	20	19	20	22	26
0,6	18	18	20	21	24	26	24	30
0,8	18	18	19	18	18	18	24	30
1	26	25	28	25	28	28	25	31

Graf 4



Záver:

Ako výsledky behu algoritmu ukázali, optimálna volba fitness funkcie, je fitness funkcia číslo 6, teda výhodné sú funkcie, ftoré robia väčšie rozdiely medzi optimálnejšími a menej optimálnymi riešeniami. Takisto markantný vplyv na kvalitu riešenia mapli parametre P_{mut} a P_{repro} , ktorých vhodná voľba má zabezpečiť možnosť pre riešenie s novými vlastnosťami dostať sa do populácie. Takáto voľba parametrov je P_{mut}=0,2 a P_{repro}=0,6. Dalším aspektom ovplyvňujúcim konvergenciu metódy je dĺžka chromozómu. Na nájdenie riešenia jeho optimalizáciu je vhodné použiť dĺžku chromozómu, ktorý kóduje približne 1,5 až 2-násobok dĺžky optimálneho riešenia. Keďže toto riešenia vopred nieje známe, je možné najprv zvoliť chromozóm obsahujúci $n^2/2$ ťahov, teda maximaálne možné riešenie a tento chromozóm postupne zmenšovať.

Odporúčania:

P_{mut}:	0,2
P_{repro}:	0,6
Fitness:	6
Pocet krokov robota:	30 (teda približne 2-násobok optimálneho riešenia)