

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Semestrální práce

KMA/MM Genetické algoritmy (a jejich aplikace)

Obsah

1	Úvod	2
2	Genetické algoritmy	3
2.1	Inspirace	3
2.2	Motivace	3
2.3	Trocha historie	3
2.4	Korespondence s Darwinovou teorií	4
2.5	Genetický algoritmus	4
2.6	Reprezentace jedinců	4
2.7	Inicializace	5
2.8	Cenová funkce	5
2.9	Optimum	5
2.10	Křížení	6
2.11	Mutace	6
2.12	Interpolace	6
2.13	Reálné použití	6
2.14	Nevýhody	6
2.15	Zrychlení konvergence	7
3	Aplikace GA	8
3.1	Lineární regrese (L2 aproximace)	8
3.2	Regrese goniometrickou funkcí	9
3.3	Výsledky	9
4	Závěr	13

1 Úvod

Cílem této práce je seznámit čtenáře s problematikou týkající se genetických algoritmů a ukázat aplikaci tohoto aparátu na vyhledávání aproximační funkce vrcholů, která je buď přímka (lineární regrese) nebo funkce, která je složená z goniometrických funkcí (něco na způsob Fourierových řad), jejíž střední kvadratická chyba je od zadaných bodů co nejmenší (v podstatě numerická L2 aproximace).

2 Genetické algoritmy

V této kapitole si povíme něco o genetických algoritmech. Měli bychom se dozvědět jak fungují a v jakých oblastech je vhodné jejich použití.

2.1 Inspirace

Genetické algoritmy byly inspirovány při jejich vzniku přirozenou evolucí v přírodě. Přírodní evoluce je robustní a úspěšná metoda adaptace v biologických systémech. Mezi hlavní pilíře patří:

- potomci dědí vlastnosti svých rodičů
- lepší jedinci přežívají déle, mají tedy více potomků
- potomci, díky přirozenému výběru, předávají dále dobré vlastnosti

V přírodě evoluce vyžaduje hodně času, ale pomocí počítače jsme schopni ohodnotit tisíce umělých bytostí během krátkého časového úseku.

2.2 Motivace

Genetické algoritmy jsou aparát, který (při dobrém návrhu) umí řešit různé problémy, aniž by mu bylo přesně popsáno, jak je má řešit. Jediné, co je třeba umět, je ohodnotit nějaký přibližný výsledek cenou (tzv. fitness). Pak už jen vybíráme výsledek s výhodnější cenou.

Genetické algoritmy tedy můžeme použít tehdy, pokud se např. chceme vyhnout složitým výpočtům. Ale neznamená to, že o řešené úloze nemusíme znát nic my. Pro optimální návrh genetického algoritmu je vhodné vědět o daném problému co nejvíce.

2.3 Trocha historie

- 1960 - Ingo Rothenberg představuje myšlenku evolučních výpočtů.
- 1975 - John Holland poprvé popisuje genetický algoritmus.
- 1992 - John Koza použil genetické algoritmy k vývoji programů, které mají plnit určité zadané úlohy.

2.4 Korespondence s Darwinovou teorií

V genetických algoritmech můžeme hovořit o těchto pojmech z oblasti Darwinovy teorie:

- **Jedinec** - řetězec hodnot, představující genetickou informaci daného jedince.
- **Přírodní výběr** - výběr vhodných jedinců podle cenové funkce.
- **Křížení** - kombinace dvou genetických řetězců za účelem vzniku nového (a pokud možno lepšího) jedince.
- **Mutace** - úprava hodnoty v genetickém řetězci jedince.

2.5 Genetický algoritmus

Struktura genetického algoritmu vypadá přibližně takto:

1. **Inicializace algoritmu** - vygenerování náhodné populace.
2. **Ohodnocení jedinců** - spočítání cenového ohodnocení (dle cenové funkce) pro každého jedince.
3. **Vyhledání optima** - pokud jsme našli optimálně ohodnoceného jedince (dle stanoveného kritéria), končíme algoritmus a jedince s tímto ohodnocením prohlásíme za výsledek.
4. **Křížení** - zkřížíme určité procento jedinců v aktuální populaci a vzniklými potomky nahradíme ty nejhorší jedince.
5. **Mutace** - náhodně provedeme mutaci jedinců v aktuální populaci.
6. Opakujeme bod 2.

2.6 Reprezentace jedinců

Jedince pro genetické algoritmy většinou navrhujeme tak, aby co nejlépe popisovali řešený problém. V praxi se většinou používají tyto varianty:

- **Binární reprezentace** - každý gen jedince je reprezentován jedním bitem (popř. skupinou bitů).

- Reálná reprezentace - každý gen představuje skutečná data, která jedince charakterizují (např. parametry funkce, vrcholy geometrického útvaru, atp.).

Oba způsoby mají své výhody a nevýhody. Např. u binární reprezentace jedinců je velmi jednoduché křížení a mutace (neboť hodnoty nabývají pouze jedničky a nuly), ale pro praktické užití se dost často hodí spíše reprezentace reálná, neboť lépe popisuje řešený problém.

2.7 Inicializace

Inicializace obvykle probíhá tak, že všem genům se nastaví nějaká náhodná (pseudonáhodná) hodnota z rozsahu, ze kterého mohou jednotlivé geny nabývat své hodnoty. U binární reprezentace je to snadné, tam jsou jen dvě možnosti (jedna nebo nula). U reálné reprezentace je rozumné si stanovit nějaký omezený interval (generovat např. reálná čísla z rozsahu $\pm\infty$ není úplně dobré). Interval by měl být volen tak, aby bylo možné (pokud možno) vyjádřit všechna řešení, blížíci se k optimu (určení takového intervalu může být někdy opravdový problém).

2.8 Cenová funkce

Cenová funkce je taková funkce, která nějak ohodnocuje řetězec genů jedince. Je nutné, aby šlo ohodnotit všechny potencionální jedince, kteří mohou během výpočtu genetického algoritmu vzniknout. Např. kdybychom počítali L2 aproximaci pomocí genetických algoritmů, byla by cenová funkce definovaná jako součet kvadratických odchylek aproximovaných vrcholů, s regresní křivkou, pro každé nastavení paramterů křivky (v případě přímky $ax + b$ by to pak byla funkce dvou proměnných - a a b).

2.9 Optimum

Hledání optima závisí na zvolené cenové funkci. Některá cenová funkce přiřazuje lepším výsledkům malá čísla, horším ty velká (střední kvadratická chyba). Jiné mohou třeba lepším výsledkům přiřazovat ohodnocení vysoké a těm horším zase malé (např. když bude cena jedince záviset na tom, jak dlouho jedinec přežil v nějaké virtuální simulaci).

2.10 Křížení

Křížení (nebo také rekombinace) obvykle probíhá tak, že si část genů vezmeme z jednoho jedince a část genů z druhého jedince. Budto nalezneme nějakou dělicí úroveň a geny před touto úrovní bereme z prvního rodiče, vše za touto hranicí pak z druhého rodiče. Nebo také můžeme střídavě (náhodně) brát geny od obou rodičů. Většinou závisí na typu úlohy.

2.11 Mutace

Mutace probíhá tak, že občas (s nějakou malou pravděpodobností) vybereme jeden gen nově vzniklého jedince a trochu ho upravíme. V případě binární reprezentace změníme nulu na jedničku (a obráceně), v případě reálné hodnoty bychom jí lehce modifikovali (v rámci nějaké tolerance). Někdy je ale také dobré gen opět náhodně vygenerovat z celého rozsahu.

Mutace slouží především k tomu, aby se genetický algoritmus nezasekl v lokálním optimu.

2.12 Interpolace

Interpolace je druhem křížení, které zastánci čisté podoby genetických algoritmů příliš neuznávají. Pravdou ale je, že u některých úloh (převážně u těch geometrických) to může značně zrychlit konvergenci. Jak už název napovídá, jde o to, že se místo rekombinace vzájemně interpolují korespondující geny rodičů (pokud jsou geny čísla, provedeme klasickou lineární interpolaci s náhodným interpolačním koeficientem z rozsahu $\langle 0; 1 \rangle$).

2.13 Reálné použití

Jak už bylo řečeno, genetické algoritmy použijeme tam, kde buď neumíme přesně nastavit nějaké parametry (neboť je nejde ani třeba dost dobře odhadnout) nebo také tam, kde je vyžadován nějaký složitý matematický postup (třeba také numericky velmi nestabilní).

2.14 Nevýhody

Hlavní nevýhodou genetických algoritmů je fakt, že nemusí vždy konvergovat (hlavně z důvodů špatně zvolených rozsahů, reprezentace nebo způsobu

křížení a mutace). Další nevýhodou je určitě jejich pomalá konvergence. Existují sice postupy, jak konvergenci zrychlit, ale i tak nikdy nepoběží v reálném čase.

2.15 Zrychlení konvergence

Vyšší rychlost konvergenci genetických algoritmů můžeme zajistit tak, že použijeme ještě nějaký další iterační postup, který ale nemusí nutně hledat globální optima. Algoritmus se pak změní do takovéto podoby:

1. **Inicializace algoritmu** - vygenerování náhodné populace.
2. **Ohodnocení jedinců** - spočítání cenového ohodnocení (dle cenové funkce) pro každého jedince.
3. **Vyhledání optima** - pokud jsme našli optimálně ohodnoceného jedince (dle stanoveného kritéria), končíme algoritmus a jedince s tímto ohodnocením prohlásíme za výsledek.
4. **Křížení** - zkřížíme určité procento jedinců v aktuální populaci a vzniklými potomky nahradíme ty nejhorší jedince.
5. **Mutace** - náhodně provedeme mutaci jedinců v aktuální populaci.
6. **Urychlení** - použijeme několik málo iterací jiného algoritmu (na aktuální geny jedince) a upravíme tak genetickou strukturu všech jedinců populace.
7. Opakujeme bod 2.

Pokud např. počítáme kořeny nějaké funkce, můžeme jako **urychlovací** metodu použít třeba několik iterací *metody tečen* či *regula falsi*.

3 Aplikace GA

V této kapitole si povíme jak modelovat některé problémy pomocí genetických algoritmů. Uvedené problémy jsou geometrického typu, aby bylo možné výsledky snadno prezentovat. Většina úloh praktické využití nemá, ale mohou inspirovat k nalezení řešení pro složitější problémy.

3.1 Lineární regrese (L2 aproximace)

Lineární regresi určitě každý zná. Máme množinu bodů a snažíme se proložit přímkou tak, aby odchylka od každého vrcholu byla co nejmenší. V lineární algebře jsme se setkali s *normální rovnicí* (metoda nejmenších čtverců), díky níž šlo pomocí maticových operací spočítat parametry přímky. Pokud máme parametrů málo (v případě přímky jsou pouze dva), tak je možné *normální rovnici* použít. My se však budeme zabývat numerickým výpočtem, který je vhodný pro velké množství parametrů.

Funkce přímky má tento tvar:

$$f(x) = ax + b \quad (3.1)$$

Abychom mohli ohodnotit danou konfiguraci paramterů a a b , potřebujeme cenovou funkci. Ta je kupodivu jednoduchá. Stačí nasčítat kvadratické odchylky od *regresní* přímky (N je počet vrcholů):

$$cost(a, b) = \sum_{i=1}^N (f(x_i) - y_i)^2 \quad (3.2)$$

Po rozepsání tedy:

$$cost(a, b) = \sum_{i=1}^N (ax_i + b - y_i)^2 \quad (3.3)$$

Genetický algoritmus se tedy pokusí hledat takové řešení, které má cenové ohodnocení pro parametry a a b co nejmenší (tehdy je totiž v součtu kvadratická odchylka od každého vrcholu co nejmenší).

Reprezentace funkce přímky je velice jednoduchá. Každý gen jedince představuje jeden parametr přímky (a a b). Křížení a mutace pak probíhá podle pravidel uvedených v kapitole 1.

3.2 Regrese goniometrickou funkcí

Abych byl přesný, není to regrese pouze jednou goniometrickou funkcí, ale regrese součtem různých sinusovek (s různou amplitudou, frekvencí a fází). Hledaná funkce má pak tento tvar (K je počet sinusovek v součtu):

$$f(x) = \sum_{i=1}^K a_i \sin(b_i x + c_i) \quad (3.4)$$

Abychom mohli ohodnotit hledanou funkci, potřebujeme cenovou funkci:

$$\text{cost}(a_1, b_1, c_1, \dots, a_K, b_K, c_K) = P + \sum_{i=1}^N (f(x_i) - y_i)^2 \quad (3.5)$$

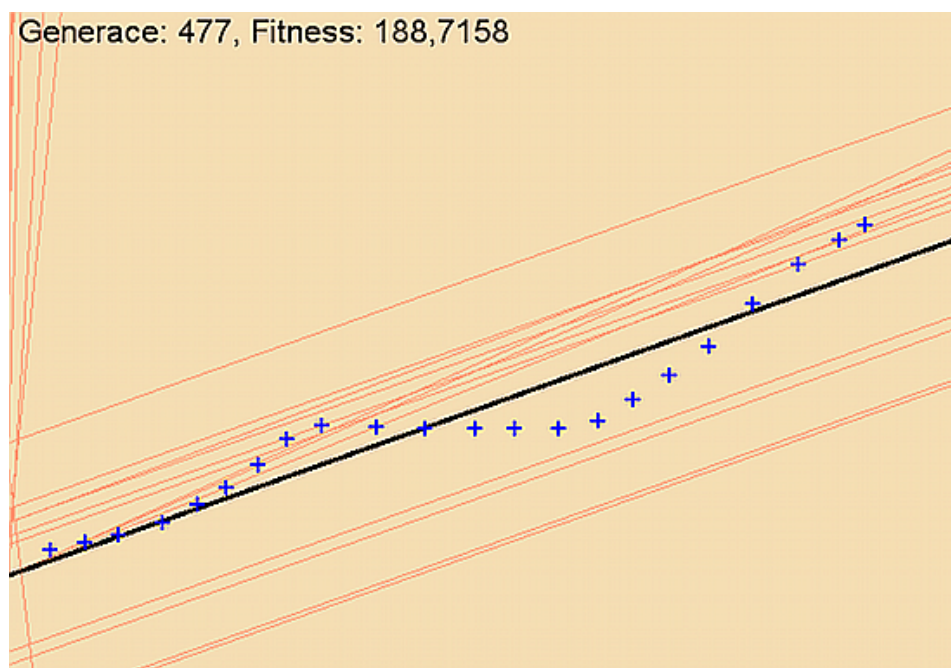
$$P = 50 \cdot K + 50 \cdot \max(a_0, \dots, a_K) \cdot \max(b_0, \dots, b_K) \quad (3.6)$$

Hodnota P nám pomáhá v tom, aby se potlačili sinusovky s vysokou amplitudou (parametr a) a rovněž s vysokou frekvencí (parametr b). Fázový posun (parametr c) nám v ničem nevádí, takže ho do cenové funkce nezahrneme.

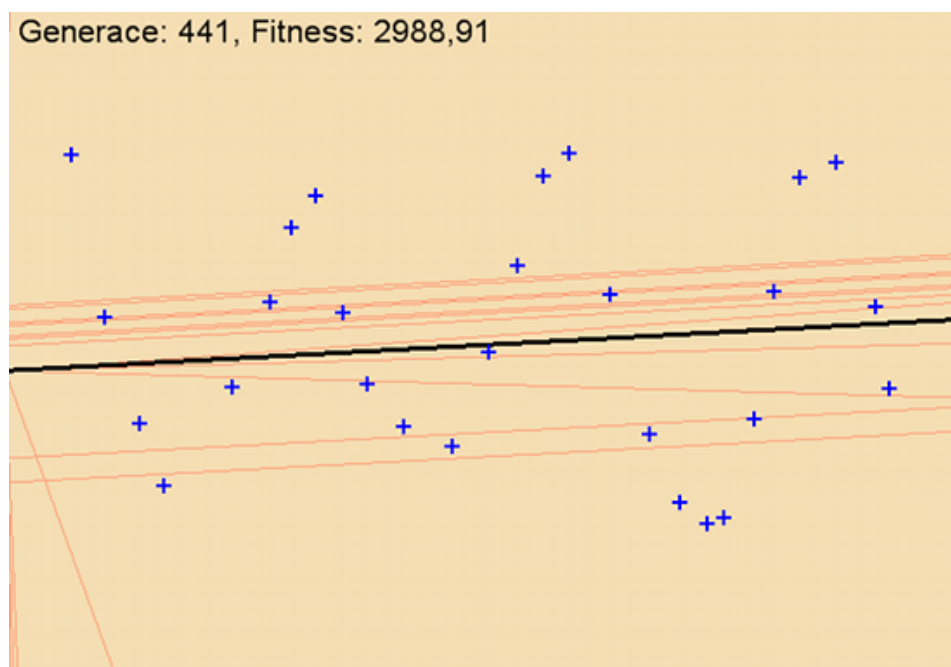
Kdyby se někdo pokoušel najít analytické řešení této úlohy, určitě by se u toho pořádně zapotil. Nejblíže se dostal pan Fourier ☺.

3.3 Výsledky

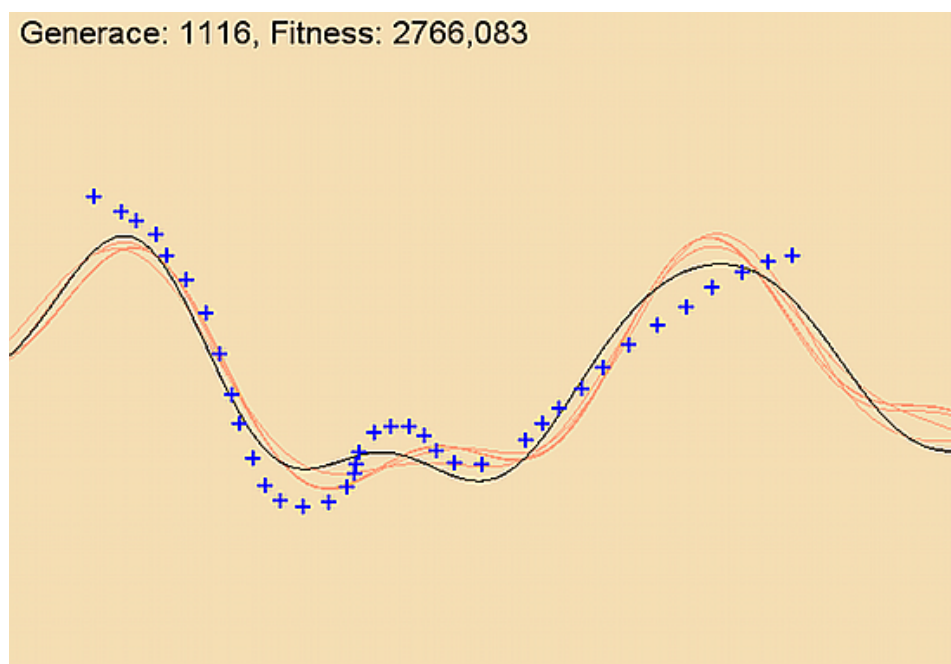
Nyní si ukážeme některé výsledky řešených problémů.



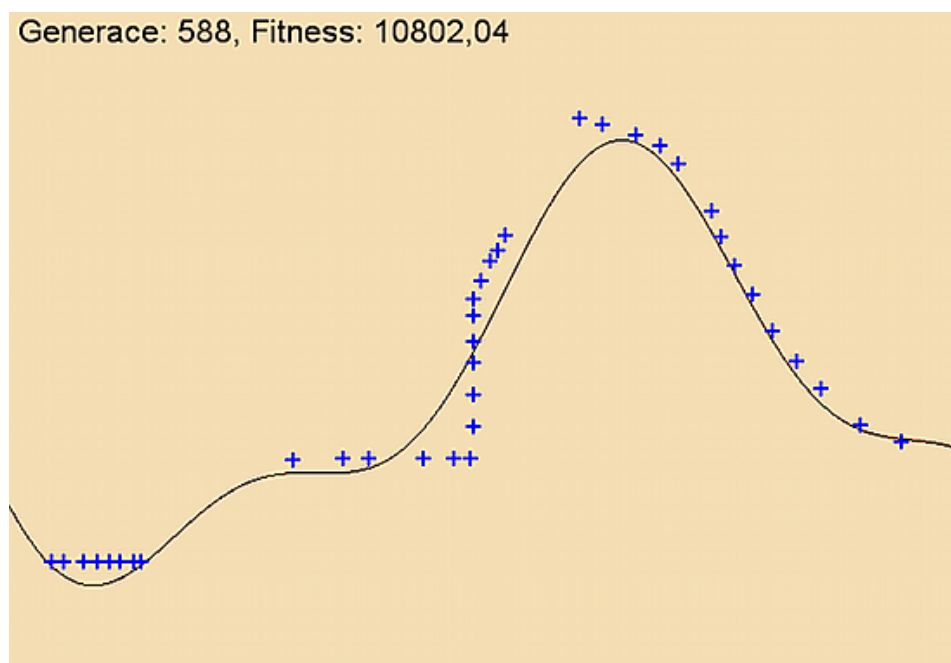
Obrázek 3.1: Výsledek lineární regrese řešené pomocí GA.



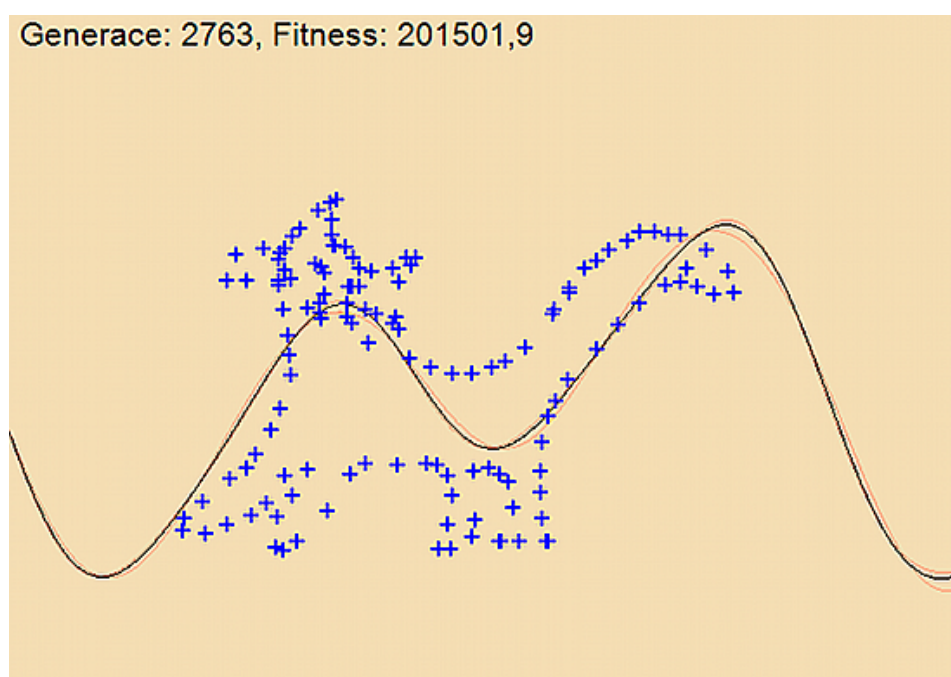
Obrázek 3.2: Výsledek lineární regrese řešené pomocí GA.



Obrázek 3.3: Goniometrická regrese řešená pomocí GA.



Obrázek 3.4: Goniometrická regrese řešená pomocí GA.



Obrázek 3.5: Goniometrická regrese kočičky řešená pomocí GA.

4 Závěr

Genetické algoritmy, spolu s neuronovými sítěmi, jsou asi jedno z nejzajímavějších odvětví moderní informatiky. Díky genetickým algoritmům je možno řešit problémy, které se na první pohled zdají buď neřešitelné nebo je jejich řešení velice komplikované, popř. náročné na výkon počítače. Genetické algoritmy je možné nasadit téměř na jakýkoliv problém, pokud se tedy spokojíme s tím, že výsledek nedostaneme okamžitě.

Osobně se genetickými algoritmy intenzivně zabývám a rád bych v jejich zkoumání pokračoval dále na doktorském studiu, neboť je to obor, který se stále rozvíjí a určitě má mnoho co nabídnout.