

EVA I

NAIL 025 – 2016/17

Roman Neruda

CZECH VERSION, 13-01-2017



Témata, literatura, osnova.

ÚVOD

Literatura

- Mitchell, M.: *Introduction to Genetic Algorithms*. MIT Press, 1996.
- Eiben, A.E and Smith, J.E.: *Introduction to Evolutionary Computing*, Springer, 2007.
- Michalewicz Z.: *Genetic Algorithms + Data Structures = Evolution Programs* (3ed), Springer, 1996
- Holland, J.: *Adaptation in Natural and Artificial Systems*, MIT Press, 1992 (2nd ed).
- Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

Témata

- **Modely evoluce** - základní přístupy a pojmy. Populace, rekombinace. Ohodnocení úspěšnosti individua.
- **Genetické algoritmy.** Zakódování řešeného problému do chromozómu. Základní genetické operace, selekce, křížení, mutace.
- **Selekce** - simulace přirozeného výběru. Účelová funkce. Dynamická vs. statická selekce, mechanismus rulety, turnaje, elitismus.
- Reprezentační schémata, vlastnosti, věta o schématech. **Hypotéza o stavebních blocích.** Věta o implicitním paralelismu.
- Vězňovo dilema, iterované vězňovo dilema, strategie, ekvilibria, evoluční stabilita.
- Evoluční strategie, kooperace individuí, (1+1) ES, (m+1) ES, rychlosť konvergencie, metaparametry.
- Diferenciální evoluce, CMA-ES.
- **EA a kombinatorické problémy.** Řešení NP-úplných úloh, problém obchodního cestujícího, problém batohu.
- Strojové učení a datamining. Evoluce expertních systémů, vnitřní reprezentace, Michiganský vs. Pittsburghský přístup.
- **Klasifikační systémy.** Učení pravidel "if-then", makléřský algoritmus, Q-učení, produkční systémy.

Úvod, biologická inspirace, základní algoritmus, vývoj a části.

EVOLUČNÍ ALGORITMY

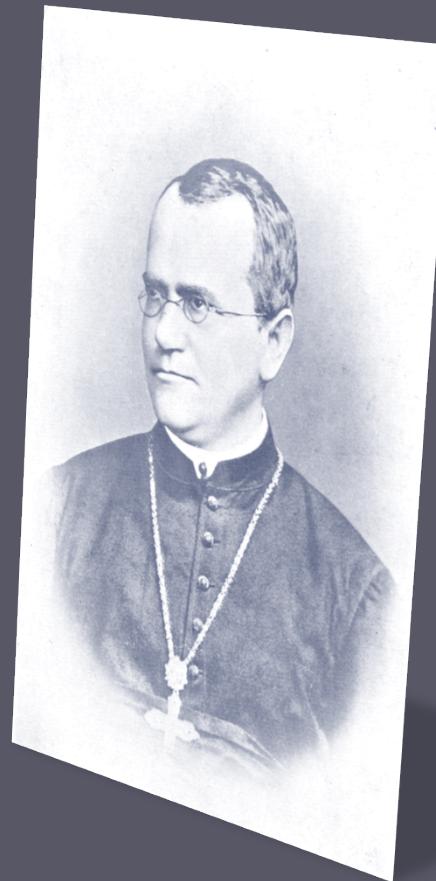
Darwinova evoluční teorie

- 1859 – O původu druhů
- omezené zdroje prostředí
- Základem života je reprodukce
- Jedinci lépe přizpůsobení prostředí mají větší šanci se reprodukovat
- Úspěšné fentotypické rysy se více reprodukují, rekombinují, náhodně mění



Mendelova genetika

- 1856 - Versuche über Pflanzenhybriden
- Gen (původně faktor) jako základní jednotka dědičnosti
- Každý diploidní jedinec má dva páry alel, jedna z nich se přenese do potomka, nezávisle na ostatních
- Je to složitější:
 - Polygenie – více genů ovlivňuje jeden znak
 - Pleiotropie – jeden gen ovlivňuje více znaků
 - Mitochondriální DNA
 - Epigenetika



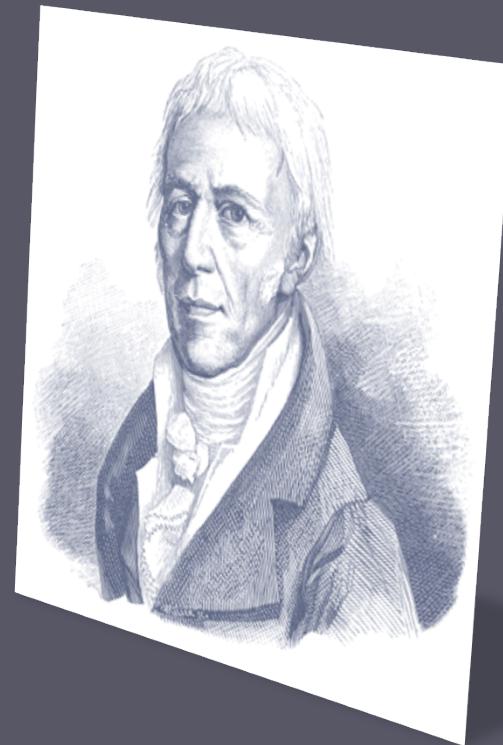
DNA

- 1953 – Watson&Crick – dvojitá šroubovice DNA
- Molekulárně-biologické vysvětlení:
 - jak je genetická informace uložena v organismu
 - jak se dědí
- DNA je tvořena 4 nukleotidy/bazemi – adenin, guanin, cytosin, thymin
- Kodon - trojice nukleotidů kódující 1 z 23 aminokyselin (redundance)
- Těchto 23 aminokyselin je základní stavební složkou bílkovin ve všech živých organismech



Molekulární genetika

- Křížení
- Mutace
- Transkripce: DNA->RNA
- Translace: RNA->protein
- GENOTYP->FENOTYP
- Jednosměrné velmi komplexní zobrazení
- Lamarckismus:
 - existuje zpětný přenos fenotyp->genotyp
 - získané vlastnosti lze dědit

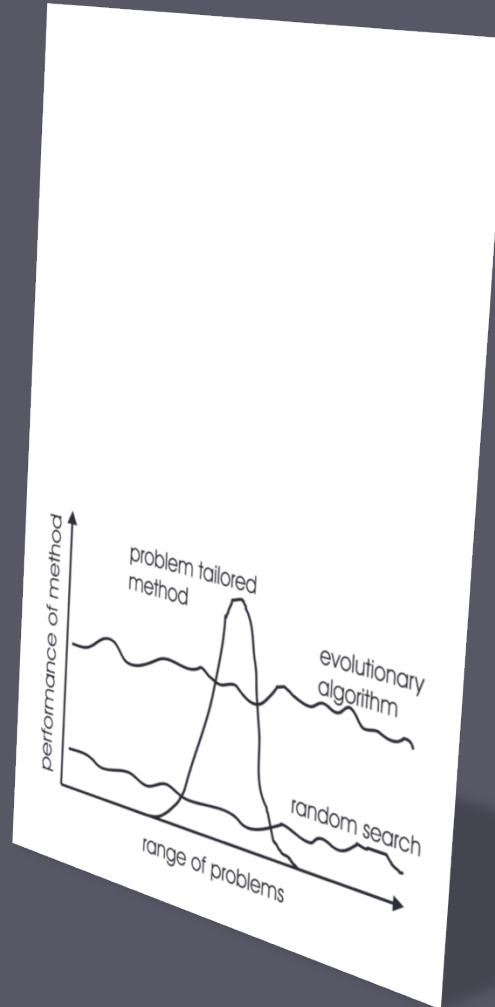


EA - shrnutí

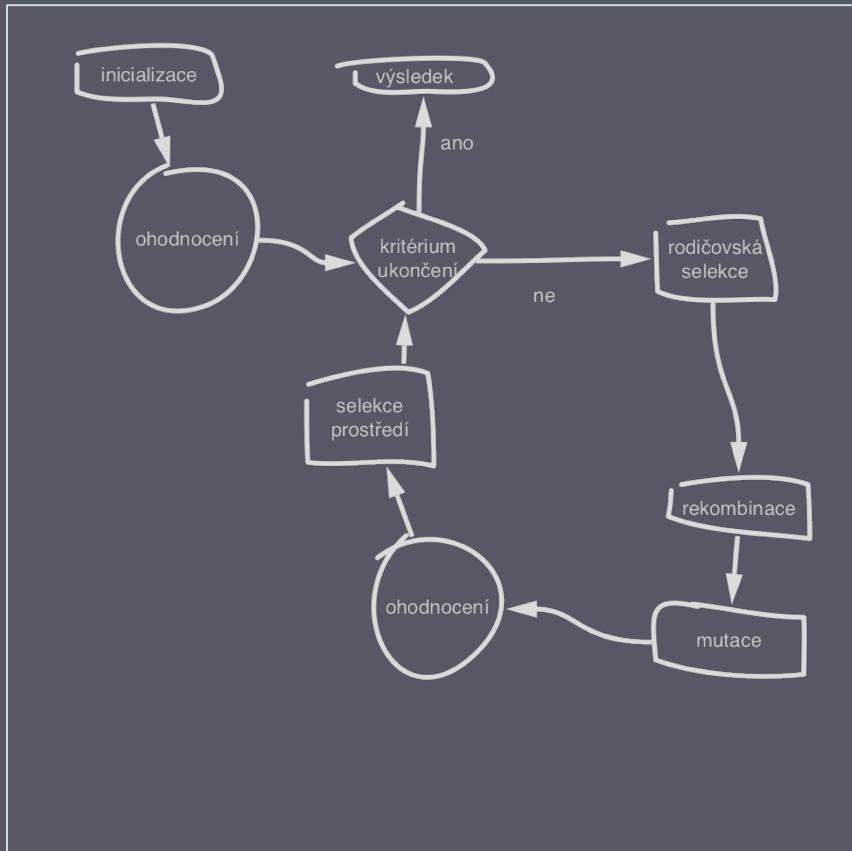
- Přírodní evoluce: prostředí, jedinci, fitness
- Umělá evoluce: problém, kandidáti řešení, kvalita řešení
- EA jsou populační stochastické prohledávací algoritmy
- Rekombinace a mutace zajišťují variabilitu
- Selekce vede prohledávání správným směrem

Obecný EA

- EA jsou robustní meta-algoritmus
- No free lunch theorem
 - neexistuje jeden nejlepší algoritmus
- Vyplatí se vytvářet doménově specifické varianty EA
 - Reprezentace
 - Operátory



Obecný EA



- Náhodně vytvoř iniciální populaci $P(0)$
- V cyklu vytvářej $P(t)$ z $P(t+1)$:
 - Výběr rodičů
 - Rekombinace a mutace
 - Tím vznikají noví jedinci
 - Environmentální selekce vybere $P(t+1)$ na základě $P(t)$ a nových jedinců

Genetické algoritmy

- 1975 - Holland
- Binárně zakódovaní jedinci
- Ruletová selekce
- Operace 1-bodového křížení
- Bitové mutace
- Inverze
- Teorie schémat k vysvětlení funkce GA

Evoluční programování

- 1965 – Fogel, Owens a Walsh
- Evoluce konečných automatů
- Smazán rozdíl mezi genotypem a fenotypem
- Důraz na mutace
- Většinou neexistuje křížení
- Turnajové selekce

Evoluční strategie

- 1964 - Rechenberg, Schwefel
- Optimalizace vektorů reálných čísel
- Floating point zakódování jedince
- Základní operací je mutace
- Krok mutace je určován heuristikou nebo adaptací (evoluje se)
- Deterministická selece prostředí

Genetické programování

- 1992 – Koza
- Evoluce jedinců zakódovaných jako (LISPovské) stromy
- Použití (nejen) k evoluci programů
- Specifické operátory křížení, mutace, inicializace
- Další aplikace (neuroevoluce, evolving hw, ...)

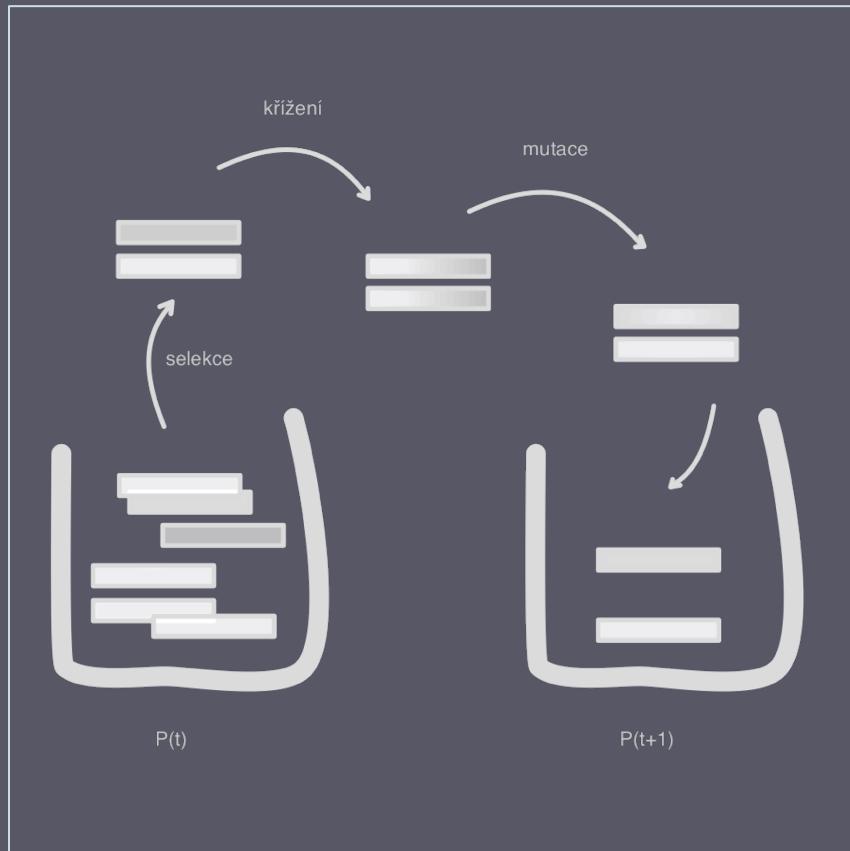
Hollandův JGA, binární reprezentace, varianty operátorů

JEDNODUCHÝ GENETICKÝ ALGORITMUS

GA

- Genetické algoritmy – 70. léta USA, Holland, DeJong, Goldberg, ...
- Původní návrh se dnes označuje SGA (jednoduchý GA)
 - minimální sada operací, co nejjednodušší kódování, teoretický výzkum
- Postupem času se GA obohatily o další zakódování i operátory

SGA - princip



- $t=0$; Náhodně vygeneruj populaci $P(0)$ n l-bitových jedinců
- Přechod od $P(t)$ k $P(t+1)$:
 - Spočti $f(x)$ pro každé x z $P(t)$
 - Opakuj $n/2$ krát:
 - selekcí vyber páry rodičů x, y z $P(t)$
 - s pravděpodobností p_C aplikuj křížení na x, y
 - s pravděpodobností p_M mutuj každý bit x, y
 - Vlož x, y do $P(t+1)$

Selekce

- *Ruletová selekce:*
 - selekční mechanismus závislý na hodnotě fitness jedince
 - očekávaný počet vybrání jedince by měla být závislá na podílu jeho fitness a průměrné fitness populace
 - Ruletová selekce: každý jedinec má alokovánu kruhovou výseč rulety odpovídající fitness, n-krát ruletou zatočím.

Křížení

- V GA je křížení hlavním operátorem
- Rekombinuje vlastnosti rodičů
- Doufáme, že rekombinace povede k lepší fitness
- *Jednobodové křížení:*
 - náhodně zvolíme bod křížení,
 - vyměníme odpovídající části jedinců
 - Pravděpodobnost pC typicky v rozsahu desetin

Mutace

- V GA hraje spíše podřadnou roli pojistky proti uvíznutí v lokálních extrémech.
- (V EP nebo raných ES to byl naopak jediný zdroj variability)
- Bitová mutace:
 - S pravděpodobností p_M změním každý bit jedince
 - p_M je malá (např. tak, aby průměrně změnila 1 bit v jedinci)

Inverze apod

- V původním návrhu SGA Holland používal další genetický operátor – *inverzi*
- *Inverze*
 - Obrácení části řetězce
 - Ovšem se zachováním významu bitů
 - Složitější technické řešení
 - Inspirace v přírodě
 - Neukázala se jako výhodná

Věta o schémetech, hypotéza o stavebních blocích, implicitní paralelismus, k-ruký bandita

TEORIE SCHÉMAT

Schéma

- *Jedinec* je slovo v abecedě {0, 1}
- *Schéma* je slovo v abecedě {0, 1, *}
 - (* = don't care)
- Schéma reprezentuje množinu jedinců
- Schéma s r * reprezentuje 2^r jedinců
- Jedinec délky m je reprezentován 2^m schématy
- Je 3^m schémat délky m
- V populaci velikosti n je 2^m až $n \cdot 2^m$ schémat

Vlastnosti schémat

- *Řád schématu S:* $o(S)$
 - Počet 0 a 1 (*pevných pozic*)
- *Definující délka:* $d(S)$
 - Vzdálenost mezi první a poslední pevnou pozicí
- *Fitness schématu:* $F(S)$
 - Průměrná fitness odpovídajících jedinců v populaci
 - Takže závisí na populaci.

Věta o schématech

- Krátká nadprůměrná s malým řádem schémata se v populaci během GA exponenciálně množí. (Holland)
- Hypotéza o stavebních blocích:
 - GA hledá suboptimální řešení problému rekombinací krátkých, nadprůměrných s malým řádem schémat (building blocks).
 - “just as a child creates magnificent fortress through arrangement of simple blocks of wood, so does a GA seek near optimal performance ...”

Důkaz VoS

- Populace $P(t)$, $P(t+1)$, ... n jedinců délky m
- Co se děje s konkrétním schématem S při:
 - Selekcí
 - Křížení
 - Mutaci
- $C(S,t)$... četnost schématu S v populaci $P(t)$
 - (počet jedinců v reprezentovaných S v $P(t)$)
- Postupně odhadujeme $C(S,t+1)$

Důkaz VoS

- Selekcce:
 - Řetězec v má pravděpodobnost vybrání:
 $p_s(v)=F(v)/F(t)$, kde $F(t) = \sum F(u)$, $\{u \in P(t)\}$
 - Schéma S má pravděpodobnost vybrání:
 $p_s(S)=F(S)/F(t)$
 - Tedy: $C(S,t+1) = C(S,t) \cap p_s(S)$
 - Jinými slovy: $C(S,t+1)=C(S,t) \cdot F(S)/F_{\text{prum}}(t)$
 - $F_{\text{prum}}(t)=F(t)/n$... průměrná fitness v $P(t)$

Důkaz VoS

- ... ještě selekce:
 - Shrňme si to: $C(S,t+1) = C(S,t) F(S)/F_{\text{prum}}(t)$
 - Kdyby bylo schéma “nadprůměrné” o $e\%$:
 - $F(S,t) = F_{\text{prum}}(t) + e F_{\text{prum}}(t)$, pro $t=0, \dots$
 - $C(S,t+1) = C(S,t) (1+e)$
 - $C(S,t+1) = C(S,0) (1+e)^t$
 - Četnost nadprůměrných schémat roste geometrickou řadou (v populacích (po selekci)).

Důkaz VoS

- Křížení:
 - Pravděpodobnost, že schéma křížení (ne)přežije:
 - $p_d(S) = d(S)/(m-1)$
 - $p_s(S) = 1 - d(S)/(m-1)$
 - Křížení má pravděpodobnost aplikace p_c :
 - $p_s(S) \geq 1 - p_c \cdot d(S) / (m-1)$
- Selekce a křížení dohromady:
 - $C(S,t+1) \geq C(S,t) \cdot F(S)/F_{\text{prum}}(t) [1 - p_c \cdot d(S) / (m-1)]$

Důkaz VoS

- Mutace:
 - 1 bit nepřežije: p_m ;
 - 1 bit přežije: $1 - p_m$
 - Schéma přežije ($p_m << 1$):
 - $p_s(S) = (1 - p_m)^{o(S)}$
 - $p_s(S) = \dots$ asi tak $\dots = 1 - p_m \cdot o(S)$, pro malé p_m
- Selekce, křížení a mutace dohromady:
- $C(S,t+1) \geq C(S,t) \cdot F(S) / F_{\text{prum}}(t) [1 - p_c \cdot d(S) / (m-1) - p_m \cdot o(S)]$
- QED.

Důsledky VoS a HoSB

- Na zakódování záleží
- Na velikosti záleží
- Předčasná konvergence škodí
- Co GA najde:
 - $(111*****)$ a $(*****11)$ jsou nadprůměrní
 - Ale $F(111*****11) << F(000*****00)$
 - Ideál je (1111111111) ; GA ho těžko najde
- Podmínka na výběr je divná

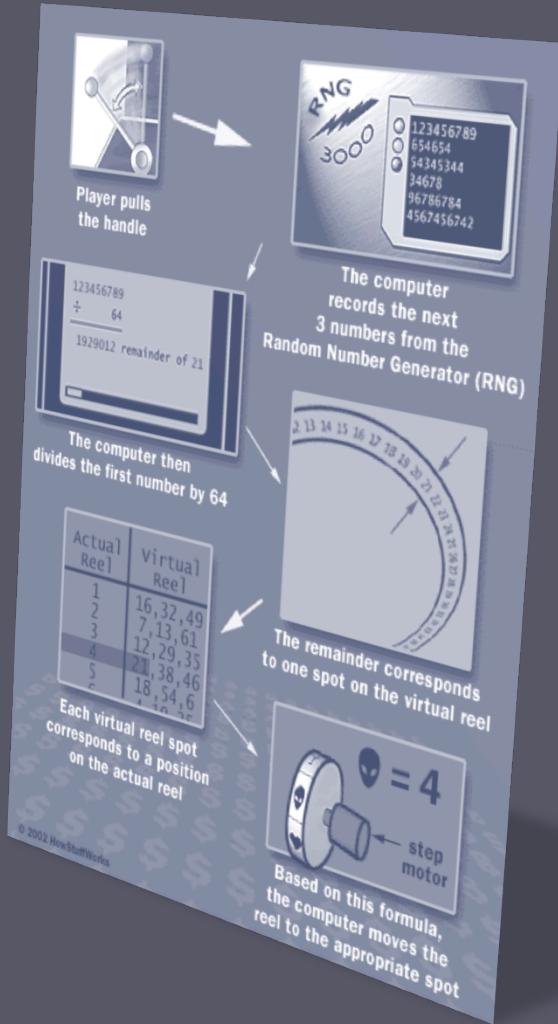
Implicitní paralelismus

- GA pracuje s n jedinci, ale “implicitně” vyvíjí (mnohem více) schémat: 2^m až $n \cdot 2^m$.
- Kolik schémat se v GA zpracuje efektivně:
 - Holland (a další): (Za různých přepokladů, např., že $n = 2^m$, schémata zůstávají nadprůměrná, ...) Počet schémat, kterým se v GA dostává exponenciálního růstu, je úměrný n^3 .
- Jediný příklad kombinatorické exploze na straně dobra. (Asi ani to ne.)

Exploration vs. Exploitation

- Původní Hollandova motivace: GA je “adaptivní plán” vyvažující při hledání řešení napětí mezi
 - *explorací* (nacházení nových oblastí hledání)
 - *exploatací* (využití stávajících znalostí)
- Jen explorace: náhodné procházky, nevyžívá se předchozích znalostí
- Jen exploatace: uvíznutí v lokálních extrémech, rigidita

1-ruký bandita



2-ruký bandita

- Má m N mincí, stojím před dvourukým banditou (ruce vyplácejí se střední hodnotou m_1, m_2 a rozptylem s_1, s_2). N-n alokuji lepší ruce, n horší.
- Cíl: maximalizovat zisk / minimalizovat ztrátu.
- Analytické řešení: alokovat exponenciálně více pokusů pro právě vyhřávající ruku.
- $N-n^* = O(\exp(c n^*))$; c závisí na m_1, m_2, s_1, s_2

Bandita a SGA

- GA taky alokuje exponenciálně mnoho nadějným schématům
- Řeší tedy optimálně problém explorace vs. exploatace
- Schémata spolu hrají mnohoruké bandity
 - Výhra je počet pozic v populaci
 - Je odhadnut hodnotou fintess schématu
 - Nejdříve se myslelo: GA hraje 3^m rukého banditu,
 - Všechny schémata jsou konkurenční ruce

... ale

- Hraje se paralelně mnohem více her
- Schémata “soutěží” o “konfliktní” pevné pozice
- Schémata řádu k soutěží vždy o těch konkrétních k pevných pozic – hrají spolu 2^k rukého banditu
- Takže vždy nejlepší z této hry dostane exponenciálně mnoho míst v populaci
- A to ještě jen tehdy, když GA může v populaci správně odhadnout fitness schémat

Takže, 'blbá' úloha pro GA je ...

- $f(x) = 2$; pro $x \sim 111*...*$
- $f(x) = 1$; pro $x \sim 0*...*$
- $f(x) = 0$; jinak
- Pro schémata platí:
 - $F(1*...*) = 1/2$;
 - $F(0*...*) = 1$
- Jenže GA odhadnou $F(1*...*) \sim 2$,
- Protože $111*...*$ v populaci převáží
- GA zde nesampluje schémata nezávisle, takže neodhadne jejich skutečnou fitness.

Problémy

- V banditovi jsou ruce nezávislé, GA ale nesampluje schémata nezávisle
- Selekce nepracuje “ideálně” jako ve VoS, je dynamická.
- GA maximalizuje on-line performance, jsou velmi vhodné pro adaptivní případy. (Škoda je zastavovat ;-)
- (Paradoxně) se nejčastěji používají “jen” pro hledání nejlepšího řešení.

Statická HoSB

- *Grafenstette, 91: Lidi předpokládají, že GA konverguje k řešením se skutečně nejlepší statickou průměrnou fitness; a ne k těm existujícím v populacích s nejlepší pozorovanou fitness.*
- To pak lidi může zklamat:
- Kolaterální konvergence
- Velký rozptyl fitnees

Kolaterální konvergence

- Když už se někam začne konvergovat, nemusejí být schémata samplována stejnouměrně.
- Je-li např. schéma 111***...* dobré, převází v populaci po pár generacích (skoro všechny řetězce tak začínají).
- Pak ale skoro všechny samplý schématu ***000...* jsou i samplý schématu 111000*...*.
- Takže GA neodhadne $F(***000*...*)$ správně.

Velký rozptyl fitness

- Má-li statická průměrná fitness schématu velký rozptyl, GA ji zase nemusí odhadnout dobře.
- Viz schéma 1*...* z našeho blbého příkladu.
- Rozptyl jeho fitness je velký, takže GA nejspíš bude konvergovat jen na těch částech prostoru, kde má fitness velkou hodnotu.
- To ale zatíží další samplování schématu chybou, takže se neodhadne správně jeho statická fitness.

Celočíselné a floating point reprezentace a jejich standardní operátory, selekce

REPREZENTACE A OPERÁTORY

Kódování

- Binární
 - Odjakživa (od Hollanda)
 - Máme pro něj teoretické výsledky (a nejen teorii schémat, jak uvidíme příští semestr)
 - *Holland argumentuje: binární řetězce délky 100 jsou lepší než desítkové délky 30, protože mají více schémat ($2^{100} > 2^{30}$).*
 - ale to nestačí, dnes víme, že schémata nejsou to nejdůležitější
 - Důležité je, že binární kódování je často nepřirozené a neefektivní pro daný problém.

Jiná kódování

- Víceznakové abecedy
- Celočíselné
- Floating point
- Ještě jiná:
 - Permutace,
 - Stromy (programy),
 - Matice,
 - Neuronové sítě (různými způsoby),
 - Konečné automaty,
 - Grafy,
 - Zvířátka...

Selekce - přehled

- **Ruletová selekce**
 - tradiční, pravděpodobnost výběru úměrná fitness
- **SUS (stochastic universal sampling)**
 - jen jedna náhodná pozice v ruletovém kole, další pozice se dopočítají posunem o $1/n$
 - „spravedlivější ruleta“ – proč?
- **Turnajová selekce**
 - k-turnaj, porovná se k jedinců, vítěz je vybrán
 - Typicky k je malé číslo
 - Možno použít i tam, kde fitness není explicitně dána, např. když se opravdu hraje nějaká hra nebo se simuluje

Celočíselné kódování

- Mutace:
 - Buď „nezatížená“ – nová hodnota z celého rozsahu
 - Nebo „zatížená“ – nová hodnota posunem od aktuální hodnoty
- Křížení:
 - Jednobodové, vícebodové, ...
 - Uniformní – u každé položky jedince házíme mincí, z kterého rodiče ho vezmeme
- Pozor na ordinální reprezentace tam, kde uspořádání nemá smysl (pak ani zatížená mutace nemá smysl)

Floating point kódování

- Historicky první pokusy o zakódování reálných čísel do bitové reprezentace, dnes se nepoužívá pokud není opravdu dobrý důvod (komprese prohledávacího prostoru, explicitní ovládání přesnosti reprezentace).
- Dnes se typicky reálná čísla kódují přirozenou floating-point reprezentací a odpovídající operace to pak repsectují (neřeže se do čísel)

Floating point operátory

- Mutace
 - Zatížená
 - Nezatížená
- Křížení
 - Strukturální
 - Jednobodové, ... uniformní
 - Aritmetické
 - Kříží se hodnoty

Aritmetická křížení

- Nejčastěji se počítá průměr hodnot obou rodičů
- Varianty:
 - místo průměru nějaká jiná konvexní kombinace:
 - $z = a*x + (1-a)*y$, kde $0 < a < 1$
 - Kolik hodnot z jedince se kříží:
 - Typicky všechny
 - Někdy jen 1 náhodně zvolená
 - Někdy kombinace s 1 bodovým křížením, kříží se část

Vězni a jejich dilemata, von Neumann, Nash, Axelrod, Dawkins

EVOLUCE KOOPERACE

Altruismus vs. darwinismus

- Darwinismus je inherentně kompetitivní – survival of the fittest
 - sociální darwinismus - odůvodnění *laissez-faire* („nech to bejt“) kapitalismu
 - Andrew Carnegie, The Gospel of Wealth, 1900 *While the law of competition may be sometimes hard for the individual, it is best for the race, because it ensures the survival of the fittest in every department. We accept and welcome, therefore, as conditions to which we must accommodate ourselves, great inequality of environment; the concentration of business, industrial and commercial, in the hands of the few; and the law of competition between these, as being not only beneficial, but essential to the future progress of the race.*
- Jak se s tím slučuje altruismus a kooperace v přírodě i společnosti?
- Hlavní problém evoluční (socio)-biologie: **Jak může evolucí vzniknout altruistické chování, které (z definice) snižuje fitness jedince?**

Teorie evoluce altruismu

- Skupinová selekce
 - evoluce může působit po skupinách (již Darwin)
 - jak vysvětlit jedince, co podvádějí a ničím nepomáhají
- Příbuzenská selekce
 - Zachování kopií genů u blízkých příbuzných
 - Ale co altruismus u cizích, či dokonce u jedinců různých druhů
- Dawkins, sobecký gen
 - jednotkou evoluce není jedinec ale gen
 - Wilson: „the organism is only DNA's way of making more DNA.“
- Trivers, 1971: reciproční altruismus
 - výhodný pro oba organismy, někdy i různého druhu
 - stín budoucnosti, paralela s iterovaným vězňovým dilematem
 - Krakonošova sojka, upíří netopýři, péče o srst u makaků

Vězňovo dilema

i/j	D	C
D	2 / 2	0 / 5
C	5 / 0	3 / 3

i/j	D	C
D	P / P	S / T
C	T / S	R / R

- *Temptation > Reward > Penalty*
• *> Suckers payoff*
- *R>P: vzájemná spolupráce je lepší než vzájemná zrada*
- *T>R a P>S: zrada je dominantní strategií pro oba hráče*
- *(50.Léta - RAND corp.)*

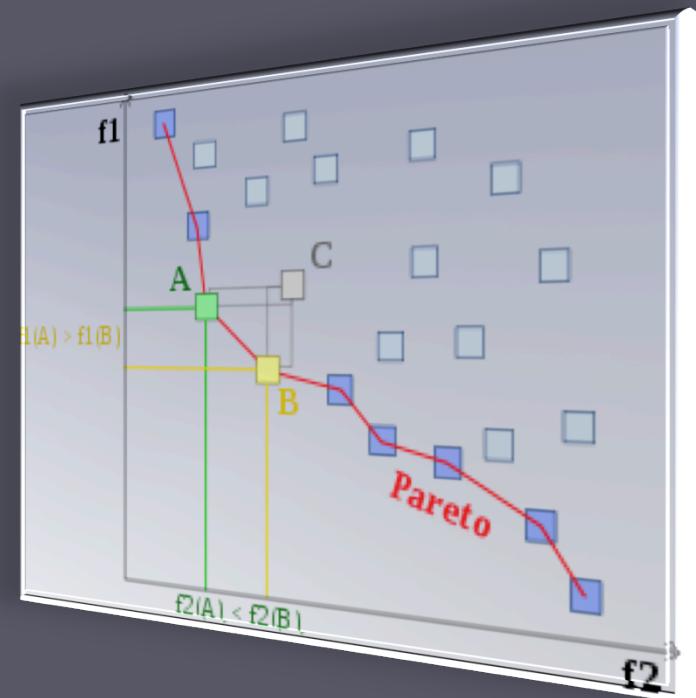


Nash

- Strategie s je *dominantní* pro agenta i , když dá lepší nebo stejný výsledek než jakékoliv jiná strategie agenta i pro všechny strategie agenta j
- Strategie s_i a s_j jsou v *Nashově ekvilibriu*, když:
 - Hraje-li agent i strategii s_1 , agent j dosáhne nejlepšího výsledku s s_2
 - Hraje-li j s_2 , pro i je nejlepší hrát s_1
- Neboli, s_1 a s_2 jsou na sebe vzájemně tou nejlepší odpovědí.
 - Tohle je Nashovo ekvilibrium ryzích strategií
 - Ale ne každá hra má Nashovo ekvilibrium v ryzích strategiích
 - A některé hry mají víc Nashových ekvilibrií

Nash a Pareto

- *Smíšené strategie* – náhodný výběr mezi ryzími strategiemi
 - *Nashova věta*: Každá hra s konečným počtem strategií má Nashovo ekvilibrium ve smíšených strategiích.
- Řešení je *Pareto-optimální/efektivní*,
 - když neexistuje jiná strategie, která by zlepšila agentův výnos bez zhoršení výnosu jiného agenta.
 - Řešení je neefektivní podle Pareta: - lze zlepšit výnos jednoho, aniž by se zhoršil výnos druhého



Takže ...

- Pro racionální agenty to dilema není/je:
 - DD je Nashovo ekvilibrium
 - DD je jediné řešení, které není Pareto-optimální
 - CC je řešení, které maximalizuje společný užitek
- Tragedy of the commons
- Co to je racionální a jsou lidé racionální?
- Stín budoucnosti – iterované verze – Axelrod

Iterované vězňovo dilema

- Soupeři hrají více her, pamatují si výsledky/akce protivníka a podle toho upravují svou strategii
- $T > R > P > S$,
- $2R > T + S$ – nevyplatí se pravidelně střídat C a D
- Pokud se hra hraje přesně N krát (a ví se to N), indukcí lze dokázat, že nejlepší strategií je „pořád zrazuj“.



Axelrodovy turnaje

- První turnaj
 - 14 strategií plus RANDOM, 200 kroků, každý s každým (i se sebou), 5x opakovaně
- TFT = Tit For Tat („půjčka za oplátku“)
 - Začni kooperací, a v dalších tazích kopíruj tahy soupeře
- Druhý turnaj
 - 62 strategií – všichni věděli, jak dopadl předchozí turnaj – vítěz opět TFT
- Třetí – „ekologický“ turnaj
 - Něco jako generace GA, iniciální generace byly strategie z 2. turnaje, hrálo se na 1000 generací
 - Množství jedinců v další generaci záviselo na počtu vítězství v předchozí generaci
 - A zase TFT!

Co to znamená pro strategie?

- 4 důležité vlastnosti úspěšných strategií:
 - Niceness – nezrazuj první
 - Provability – rychle oplácej zradu
 - Forgiveness – ale rychle se uklidni
 - Clarity – bud' jednoduchá, aby tvým akcím ostatní rozuměli
- Neexistuje jedna strategie, která by vyhrála proti všem strategiím
- Je třeba být úspěšný proti velmi různorodým typům hry (ALL-D, TFTT, RANDOM, TRIGGER)
- Je také dobré naučit se hrát sama se sebou
- Pokusy vyzrát nad TFT pomocí více zrad nepomohly

Co to znamená pro kooperaci?

- V prostředích, která podporují kooperaci ...
 - nastavení odměn,
 - velká pravděpodobnost iterace PD (shadow of the future)
- ... se kooperace většinou vyvine
 - Ale ne jen v např. ALL-D světě
- Racionalita, inteligence, vědomí, ... není pro vzájemnou kooperaci důležité, jen výhoda větších hodnot fitness
- Počáteční kooperace může dokonce vzniknout náhodně a pak se už udržet

Po dvaceti letech

- V prostředích s šumem je výhodná strategie Pavlov (win-stay, lose-shift)
 - Pokud byla odměna R nebo P => C,
 - if T or S => D
- Po 20 letech zopakovali turnaj s více strategiemi od jednoho týmu
 - Vítězné strategie kooperovaly
 - Několik (10) tahů na rozpoznání soupeře, pak všechny pomohly jedné kopii (taťkovi) získat větší skóre
 - Účastníci dokonce bojovali s organizátory (falešní účastníci, boj o místa v populaci ...)

Motivace, populační cyklus, floating point mutace, meta-evoluce

EVOLUČNÍ STRATEGIE

Evoluční strategie

- Rechenberg, Schwefel, 60.léta
- optimalizace reálných funkcí mnoha parametrů
- 'evoluce evoluce'
- evolvovaný jedinec:
 - *Genetické parametry* ovlivňující chování
 - *Strategické parametry* ovlivňující evoluci
- nový jedinec akceptován jen, je-li lepší
- na vzniku se může podílet více jedinců
- Dnes nejkomplexnější verzí je CMA-ES (correlation matrix adaptation-ES)

ES notace

- Důležité parametry:
 - M počet jedinců v populaci
 - L počet vznikajících potomků
 - R počet 'rodičů'
- Zvláštní notace souvisí se selekcí:
 - $(M+L)$ ES – M jedinců do nové populace je vybráno z $M+L$ starých i nových jedinců
 - (M,L) ES – M nových jedinců je vybráno jen z L nových potomků
 - Ukazuje se, že (M,L) je většinou robustnější k uvíznutí v lokálních extrémech.
- Jedinec $C(i)=[G_n(i), S_k(i)]$, $k=1$, nebo n , nebo $2n$

ES populační cyklus

- $n=0$; Náhodně inicializuj populaci P_n M jedinců
- Ohodnotí jedince P_n pomocí fitness
- Dokud není řešení dostatečně dobré:
 - Opakuj L krát:
 - vyber R rodičů,
 - zkřiž, mutuj, ohodnoť nového
 - Vyber M nových (podle typu ES)
 - $++n$

ES jedinec a mutace

- $C(i) = [G_n(i), S_k(i)]$
- S_k jsou standardní odchylky floating point mutací
- $k=1$:
 - jedna společná odchylka pro všechny parametry
- $k=n$:
 - nekorelované mutace
 - Každý parametr má svou odchylku
 - Geometricky se mutuje po elipse rovnoběžné s osami
- $k=2n$:
 - vylepšené o rotace, nemutuje se jen podle os dimenzí
 - korelované mutace, odpovídají mutování z n-rozměrného normálního rozdělení
 - Násobí se maticí rotace pro určení optimálních směrů
 - (stačí vektor: n-rozměrný), takže metaparametrů je 2n

ES mutace

- Genetické parametry:
 - Přičtení náhodného čísla z normálního rozdělení (s příslušnou odchylkou (a příp. rotací))
- Odchylky:
 - Zvětšovat nebo zmenšovat podle úspěšnosti mutace 1/5 pravidlo (heuristika, „nejlepší je, když mutace má 20% úspěšnost“, takže se při menší úspěšnosti odchylka zvětší a při větší úspěšnosti zmenší)
 - Anebo tradičně přičtení náhodného čísla z $N(0,1)$
- Rotace:
 - Přičtení náhodného čísla z $N(0,1)$

ES křížení

- Uniformní
- Gang bang více rodičů
 - Lokální ($R=2$)
 - Globální ($R=M$)
- Dvě verze
 - Diskrétní
 - Aritmetické (průměr)

Alternativní geometricky motivovaný floating-point evoluční algoritmus

DIFERENCIÁLNÍ EVOLUCE

DE – schéma a inicializace

- **Inicializace:** náhodné hodnoty parametrů
- **Mutace:** „posun“ podle ostatních
- **Křížení:** uniformní „s pojistkou“
- **Selekece:** porovnání a případné nahrazení lepším potomkem

Mutace

- Každý jedinec v populaci projde mutací, křížením a selekcí
- Pro jedince $\mathbf{x}_{i,p}$ zvolme tři různé jedince $\mathbf{x}_{a,p}$, $\mathbf{x}_{b,p}$, $\mathbf{x}_{c,p}$
- Definujme donora \mathbf{v} : $\mathbf{v}_{i,p+1} = \mathbf{x}_{a,p} + F \cdot (\mathbf{x}_{b,p} - \mathbf{x}_{c,p})$
- F je parametr mutace, konstanta z intervalu $<0;2>$

Křížení

- Uniformní křížení původního jedince s donorem
- Parametr C určuje pravděpodobnost změny
- Ve výsledku zajištěno aspoň 1 prvek z donora
- Pokusný vektor $ui,p+1$:
- $u_{j,i,p+1} = v_{j,i,p+1}$; iff $rand_{ji} \leq C$ or $j = l_{rand}$
- $u_{j,i,p+1} = x_{j,i,p+1}$; iff $rand_{ji} > C$ and $j \neq l_{rand}$
- $rand_{ji}$ je pseudonáhodné číslo z $<0;1>$
- l_{rand} náhodný int z $<1;2; \dots ; D>$

Selekce

- Porovnáme fitness \mathbf{x} a \mathbf{v} a lepšího vezmeme:
 - $\mathbf{x}_{i,p+1} = \mathbf{u}_{i,p+1}$; iff $f(\mathbf{u}_{i,p+1}) \leq f(\mathbf{x}_{i,p})$
 - $\mathbf{x}_{i,p+1} = \mathbf{x}_{i,p}$; jinak
 - pro $i=1, 2, \dots, N$
- Mutaci, křížení a selekci opakujeme dokud není splněno nějaké ukončovací kriterium, (typicky např. fitness nejlepšího už je dost dobrá)

Jedinec je částice a pluje si v hejnu nad krajinou fitness

PARTICLE SWARM OPTIMIZATION

PSO

- Populační prohledávací algoritmus
- Eberhart, Kennedy, 1995
- Inspirace hejny hmyzu/ryb
- Jedinec je typicky vektor reálných čísel
- Říká se mu *částice*
- Nejsou zde operace křížení,
- Ani mutace tak, jak ji známe
- Jedinci se pohybují v hejnu prostorem parametrů

PSO algoritmus

- Inicializuj každou částici
- Do
- Foreach částice
- Spočítej fitness částice
- Je-li fitness lepší než dosud nejlepší dosažená fitness částice (pBest)
- pBest := fitness;
- End
- Nastav gBest jako nejlepší fitness nejlepší částice
- Foreach částice
- spočítej rychlosť částice dle rovnice (a)
- aktualizuj pozici částice dle rovnice (b)
- End
- While maximum iterations or minimum error není splněno

PSO rovnice pohybu

- $v := v +$
 $+ c1 * rand() * (pbest - present) +$
 $+ c2 * rand() * (gbest - present)$ (a)
- $present = present + v$ (b)
- v je rychlosť častice, $present$ je pozícia častice.
- $pbest$ najlepšia pozícia častice v histórii
- $gbest$ najlepšia globálna pozícia v histórii
- $rand()$ náhodné číslo z $(0,1)$.
- $c1, c2$ konstanty (učíci faktory) často $c1 = c2 = 2$.

PSO diskuse

- Společné s GA:
 - Začínají z náhodné konfigurace, prohledávají prostor, mají fitness jako ohodnocení, používají stochastické metody
- Odlišné:
 - Nejsou zde genetické operace
 - Částice mají paměť
 - Výměna informací je jen od nejlepších částic ostatním

Michigan vs. Pittsburg, machine learning i reinforcement learning

EVOLUČNÍ STROJOVÉ UČENÍ

Strojové učení

- Učení pravidel na základě předkládaných dat
 - Data mining
 - Expertní systémy
 - Učení agentů, robotů (posilované učení)
- Základní přístupy:
 - **Michiganský** (Holland): pravidlo je jedinec
 - Hollandovy LCS: učící se klasifikační systémy (vhodné pro posilované učení)
 - **Pittsburský**: jedinec je množina pravidel

Michigan

- Holland v 80.letech: learning classifier systems
- Jedinec GA je pravidlo, celá populace pak funguje jako řídící či expertní systém
- Jednoduchá pravidla:
 - Pravá strana: příznak nastal/ne/don't care
 - Levá strana: kód akce či klasifikace kategorie
- Pravidla mají váhu (úspěšnost)
- Ta určuje fitness v evoluci
- Evoluce nemusí probíhat v generacích

Michigan - LCS

- Evoluce probíhá jen občas a jen na části populace
- Problém reaktivnosti (absence vnitřní paměti)
 - Pravidla mohou mít na pravé straně – kromě klasifikačního výstupu – další vnitřní příznaky - „zprávy“
 - a na levé straně „receptory“ na jejich příjem,
 - Systém má pak frontu zpráv a musí realizovat algoritmus odměn pro pravidla

LCS – bucket brigade

- Jen některá pravidla vedou k akci, za kterou následuje odměna/trest od prostředí,
- Rozdělení odměny – pro celý řetěz úspěšných pravidel
- Pravidla musejí dát část své síly (jakoby peněz), když chtějí soupeřit o možnost být v cestě k řešení
- Bucket brigade algoritm, v praxi komplikované a těžkopádné, ekonomika odměn se těžko vyvažuje

Z(ero)CS

- (Wilson, 1994) zjednodušené LCS
 - Žádné interní zprávy
 - Žádný složitý mechanismus alokace odměny
- **Pravidla** jsou jen reprezentace bitových map:
 - IF(vstupy) THEN (výstupy)
- Cover operátor:
 - Pokud se nenajde žádné pravidlo pro danou situaci, je vygenerováno ad hoc
 - Přidají se náhodně * a vybere se náhodný výstup

ZCS pokr.

- Jak se přiděluje odměna – upravuje síla pravidel:
 - Pravidla, která neodpovídají dané situaci, tak nic
 - Pravidla, která odpovídají vstupu, ale mají jiný výstup: síla se zmenší násobením konstantou $0 < T < 1$
 - Všem pravidlům se zmenší síla o malou část B
 - Tahle síla se rozdělí rovnoměrně mezi pravidla, která *minule* odpověděla správně, zmenšená o faktor $0 < G < 1$
 - Nakonec, odpověď od systému se zmenší o B a rozdělí rovnoměrně mezi pravidla, která *ted'* odpověděla správně

XCS – vylepšení ZCS

- Nevýhody ZCS:
 - ZCS nemá nutkání vyvíjet kompletní pravidlový systém pokrývající všechny případy
 - Pravidla na začátku řetězu akcí jsou málo odměňována (a nepřežijí)
 - Pravidla vedoucí k akcím s malou odměnou nemusejí přežívat (i když jsou důležitá)
- XCS:
 - Oddělit fitness od předpokládaného výdělku pravidla
 - Fitness založit na přesnosti pravidla, ne na množství

Pitt

- Jedinci jsou množiny pravidel
- Ohodnocení komplikovanější
 - Priority pravidel, konflikty
 - False positives, false negatives
- Genetické operátory komplikovanější
 - Typicky vede k desítkám operátorů pracujících na úrovni celých množin, jednotlivých pravidel, termů v pravidlech
- Důraz na bohatou reprezentaci domén (množiny, výčtové typy, intervaly, ...)

Pitt, například GIL

- Binární klasifikační úloha
- Jedinec klasifikuje implicitně do jedné třídy (nemá pravou stranu)
- Každý jedinec je disjunkce *komplexů*
- Komplex je konjunkce *selektorů* (*z 1 proměnné*)
- Selektor je disjunkce hodnot z domény proměnné
- Reprezentuje se bitovou mapou:
 - $((X=A1) \text{AND} (Z=C3)) \text{ OR} ((X=A2) \text{AND} (Y=B2))$
 - $[001|11|0011 \text{ OR } 010|10|1111]$

GIL pokr.

- Operátory na úrovni jedince:
 - výměna pravidel, kopírování pravidel, generalizace pravidla, smazání pravidla, specializace pravidla, zahrnutí jednoho pozitivního příkladu
- Operátory na úrovni komplexů:
 - rozdělení komplexu na 1 selektoru, generalizace selektoru (nahrazení 11...1), specializace generalizovaného selektoru, zahrnutí negativního příkladu
- Operace na selektorech:
 - Mutace $0<->1$, rozšíření $0->1$, zúžení $1->0$,

Multi-Objective Evolutionary Algorithms (MOEA), Paretova fronta, NSGA II

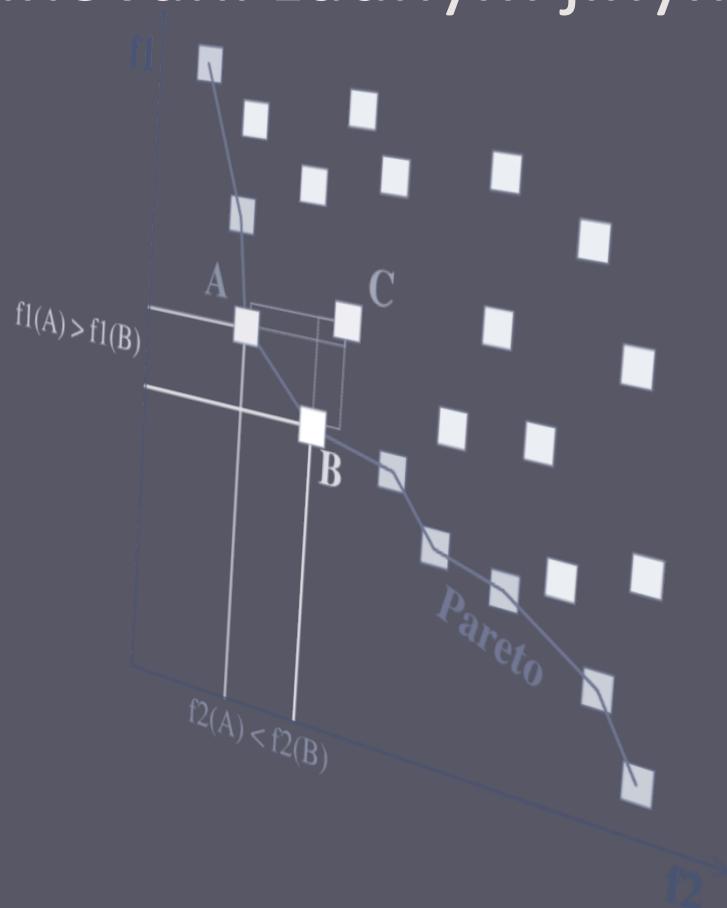
VÍCEKRITERIÁLNÍ OPTIMALIZACE

Problém

- Místo jedné fitness (účelové funkce) jich máme celý vektor $f_i, i=1\dots n$
- Uvažujeme minimalizační formulaci, takže se snažíme o co nejmenší hodnotu ve všech f_i , to je ale těžké.
- Definice dominance (jedinec, neboli řešení):
 - Jedinec x **slabě dominuje** jedince y , právě když $f_i(x) \leq f_i(y)$, pro $i=1..n$
 - x **dominuje** y , když ho slabě dominuje a existuje j : $f_j(x) < f_j(y)$
 - x a y jsou **nesrovnatelní**, když x nedominuje y , ani y nedominuje x
 - x **nedominuje** y , když buď y slabě dominuje x anebo x a y nesrovnatelní

Paretova fronta

- Paretova fronta je množina jedinců, kteří nejsou dominováni žádným jiným jedincem.



Jednoduše

- Jak řešit MOEA jednoduše:
- Agregovat fitness:
 - tzn. udělat ze všech f_i váženou sumou jednu f
 - a řešit to jako normální jednokriteriální optimalizaci
 - té se někdy v kontextu MOEA říká SOEA (single objective EA, to jsme až do teď dělali, nelekněte se)
- ale nevíme, jak nastavit váhy u jednotlivých f

VEGA (Vector Evaluated GA)

- Jeden z prvních MOEA algoritmů, 1985
- Idea:
 - populaci N jedinců utřídím podle každé z n účelových funkcí
 - pro každé i vyberu N/n nejlepších jedinců dle f_i
 - ty křížím a mutuju a vyberu do další generace
- Má to mnoho nevýhod:
 - těžko se udržuje diverzita populace
 - má to tendenci konvergovat k extrémům jednotlivých f_i

NSGA (non-dominated sorting GA)

- 1994, používá ideu dominance pro fitness
- To ale nezaručuje rozptýlení populace, které se musí řešit jinak (niching)
- Algoritmus:
 - Populace P je rozdělena na postupně konstruované fronty F_1, F_2, \dots
 - F_1 je množina všech nedominovaných jedinců z P
 - F_2 je množna všech nedominovaných jedinců z $P - F_1$
 - F_3 je m. v. n. j. z $P - (F_1 \cup F_2)$
 - ...

NSGA pokr.

- Pro každého jedince i se spočítá **niching faktor** jako součet $sh(i,j)$ přes všechny jedince j ze stejné fronty, kde:
 - $sh(i,j) = 1-[d(i,j)/dshare]^2$, pro $d(i,j) < dshare$
 - $sh(i,j) = 0$ jinak
 - $d(i,j)$ je vzdálenost i od j
 - $dshare$ je parametr, který je nutno určit předem
- Jedinci z první fronty dostanou nějakou „dummy“ fitness, která se dělí niching faktorem
- Jedinci z druhé fronty dostanou dummy fitness menší než fitness nejhoršího z první fronty, opět se dělí jejich niching faktorem
- ... pro všechny fronty

NSGA II

- 2000, odstraňuje chyby NSGA:
 - Nutnost určení dshare
 - Neexistenci elitismu
- **Niching**
 - *Dshare* a *niche count* je nahrazeno **crowding distance**:
 - To je součet vzdáleností k nejbližším sousedům.
 - Nejlepší jedinci vzhledem k jednotlivým *f_i* mají crowding distance nekonečno
- **Elitismus**
 - Stará a nová populace jsou spojeny, utříděny a vybere se lepší část do další generace

NSGA II pokr.

- **Fitness:**
 - Každý jedinec má přiřazeno číslo nedominované fronty a crowding distance
 - Když se porovnávají dva jedinci, tak nejdřív se bere v potaz číslo fronty (menší je lepší), při stejné frontě se bere crowding distance (větší je lepší)
 - A většinou se žádná fitness opravdu nepočítá, jenom se v turnajích porovnávají tahle dvě čísla

EVA řeší NP-težké úlohy, problém obchodního cestujícího, specifika permutační reprezentace

KOMBINATORICKÁ OPTIMALIZACE

EVA řeší těžké úlohy

- Batoh (0-1 knapsack problem)
 - Jednoduché zakódování
 - Jednoduchá (ale problematická) fitness
 - Standardní operátory
- Obchodní cestující - POC (TSP)
 - Jednoduchá fitness
 - Problematické kódování a příslušné op (křížení)
 - rozvrhování, plánování, dopravní problémy ...

Batoh

- Máme:
 - batoh kapacity CMAX
 - a N věcí,
 - každá má cenu $v(i)$
 - a objem $c(i)$
- Úkolem je vybrat z nich takové, abychom
 - maximalizovali sumu $v(i)$
 - a zároveň se vešli do batohu, t.j. suma $c(i) \leq CMAX$

Batoh

- Zakódování přímočaré – bitová mapa:
 - 0110010 – bereme věci 2,3 a 6
 - je to až triviální
 - ale jedinci nemusejí splňovat podmínu o CMAX
- Operátory:
 - Přímočaré křížení, mutace, selekce
- Fitness: dva členy:
 - $\max [\text{suma } v(i)]$ vs. $\min [\text{CMAX} - \text{suma } c(i)]$

Batoh

- Problém s víceúčelovou optimalizací lze řešit různě:
 - Oba členy fitness zvážit a sečíst
 - Použít 1 z obecných EA pro víceúčelové funkce
 - Anebo chytře změnit zakódování:
 - 1 znamená: DEJ věc do batohu POKUD se nepřeplní
 - takhle dosáhneme i toho, že všechny řetězce jsou přípustná řešení

Obchodní cestující

- Máme N měst, chceme je objet s minimálními náklady
- Fitness je zřejmá – náklady na cestu
- Reprezentace jsou různorodé
 - Varianty reprezentací pomocí vrcholů
 - Reprezentace pomocí hran, ...
- Operátory jsou závislé na reprezentaci
 - Křížení umožňuje využít heuristiky k řešení POC

Reprezentace sousednosti

- Cesta je seznam měst, město j je na pozici i, vede-li cesta z i do j
- Příklad
 - (248397156) odpovídá cestě 1-2-4-3-8-5-9-6-7
- Každá cesta má jen 1 reprezentaci, ale některé seznamy negenerují přípustnou cestu
- Klasické křížení nefunguje (různé opravy)
- Ale schémata ano:
 - Např (*3*...) značí všechny cesty s hranou 2-3

Reprezentace bufferem – ordinální

- Motivována tím, aby šlo použít normální 1-bodové křížení (ale má smysl?)
 - Mějme buffer vrcholů (třeba uspořádaný) a reprezentace představuje pořadí města v bufferu
 - Města se z bufferu po použití mažou
- Příklad:
 - Buffer (123456789) a cesta 1-2-4-3-8-5-9-6-7 je reprezentována jako (112141311)

Reprezentace cestou – permutací

- Každého asi napadne první,
- Má význam i pro kódování jiných úloh.
 - cesta 5-1-7-8-9-4-6-2-3 je tedy repr. (517894623)
- Nefunguje křížení, proto byly a jsou vyvíjeny operátory, které produkují správné reprezentace a navíc obsahují nějakou ideu o tom, jak udělat dobré řešení
 - PMX, CX, OX, ...

PMX

- Partially mapped crossover (Goldberg)
- Snaží se zachovat co nejvíce měst na pozicích z daných měst, jde-li to.
- 2-bodové křížení
- (123|4567|89) PMX (452|1876|93) :
 - (...|1876|..) (...|4567|..) a zobr. 1-4 8-5 7-6 6-7
 - lze doplnit (.23|1876|.9) (..2|4567|93)
 - a dle zobr. (423|1876|59) (182|4567|93)

OX

- Order crossover (Davis)
- Zachovává relativní pořadí měst dle jednotl.cest
- (123|4567|89) OX (452|1876|93) :
 - (...|1876|..) (...|4567|..) a přeuspořádáme cestu 2 od druhého bodu křížení 9-3-4-5-2-1-8-7-6
 - vyhodíme překřížená města z 1, zbyde: 9-3-2-1-8
 - doplníme potomka 1: (218|4567|93)
 - obdobně potomek 2: (345|1876|92)

CX

- Cyclic crossover (Oliver)
- Snaží se zachovat absolutní pozici města v cestě
- (123456789) CX (412876935)
 - první pozice, náhodně třeba z prvního $P1=(1.....)$,
 - teď musíme vzít 4, $P1=(1..4....)$, pak 8, 3 a 2
 - $P1=(1234...8.)$, dál už nelze, doplníme z druhého
 - $P1=(123476985)$
 - Obdobně $P2=(412856739)$

ER

- Edge recombination (Whitley et al)
- Předchozí křížení zachovají max 60% z obou rodičů,
- snaha ER je zachovat jich co nejvíce.
 - Pro každé město si udělám seznamy hran,
 - Začnu někde (1. městem),
 - Vybírám města s méně hranami,
 - V případě shod počtu hran se mezi nimi rozhodnu náhodně

(123456789) ER (412876935)

- 1: 9 2 4
- 2: 1 3 8
- 3: 2 4 9 5
- 4: 3 5 1
- 5: 4 6 3
- 6: 5 7 9
- 7: 6 8
- 8: 7 9 2
- 9: 8 1 6 3

- Začnu v 1, následníci jsou 9, 2, 4
- 9 vypadává, má 4 násl., z 2 a 4 náhodně vyberu 4
- násl. 4 jsou 3 a 5, beru 5, neboť 5 má 3 násl, zatímco 3 má 4 násl. (to je ale nepřehledné)
- Teď máme (145.....), podobně pokračujeme
- ... (145678239)
- Může dojít k tomu, že nelze hranu vybrat, a křížení selže, ale jen zřídka (1-1.5% případů)

(123456789) ER2 (412876935)

- 1: 9 #2 4
 - 2: #1 3 8
 - 3: 2 4 9 5
 - 4: 3 #5 1
 - 5: #4 6 3
 - 6: 5 #7 9
 - 7: #6 #8
 - 8: #7 9 2
 - 9: 8 1 6 3
- ER2 – vylepšení ER
 - Zachovává více společných hran
 - Označím si ty hrany, které jsou 2x - #
 - Dám jím přednost při výběru
 - To se projeví jen u vrcholů se třemi následníky, že

Inicializace pro TSP

- Nejbližší sousedi:
 - Začni s náhodným městem,
 - vybírej další jako nejbližší z dosud nevybraných
- Vkládání hran:
 - K cestě T (začni hranou) vyber nejbližší město c mimo
 - Najdi hranu $k-j$ v T tak, že minimalizuje rozdíl mezi $k-c-j$ a $k-j$
 - Vyhod' $k-j$, vlož $k-c$ a $c-j$ do T

Mutace pro TSP

- Inverze (!)
- Vkládání města do cesty
- Posun podcesty
- Záměna 2 měst
- Záměna podcest
- Heuristiky 2-opt apod.

Další přístupy

- (Binární) maticová reprezentace:
 - Bud' 1 jen pro hranu z i do j
 - Nebo 1 pro předcházení v cestě (častější)
- Specifické operátory křížení matic –
 - Průnik – AND a náhodné doplnění hran
 - Sjednocení – rozdělit na kvadranty, 2 vyhodit, odstranit kontradikce, doplnit náhodně
- Kombinace s lokálními heuristikami
 - ES, která vylepšuje cesty v populaci pomocí "mutací" - heuristik jako je 2-opt, 3-opt

Další úlohy - rozvrhování

- Rozvrhování, víme, že je NP těžké:
 - Jedinec je rozvrh, přímočaré kódování maticí
 - Řádky odpovídají učitelům, sloupce hodinám, hodnoty jsou kódy předmětů
 - Mutace – zamíchej předměty
 - Křížení – vyměň „lepší“ řádky z jedinců
 - Fitness
 - fitness řádku (jak je spokojen učitel)
 - další soft kritéria kvality rozvrhu
 - Hard omezující podmínky
 - nutno respektovat v operátorech, jinak se generuje mnoho nepřípustných řešení
 - kdy mohou učitelé, co a kde lze učit, ...

Další úlohy – job shop scheduling

- Plánování výroby
 - Výrobky o1...oN, sestávající z částí p1...pK, pro každou část více plánů, jak jí vyrobit na strojích m1...mM, stroje mají různé doby nastavení na jiný výrobek
 - Fitness – čas výroby
- Zakódování je kritické:
 - Permutační – plán je jen permutace pořadí výrobků. Dekodér pak musí zvolit plány pro části. Výhody – jednoduchá reprezentace, možnost požít křížení z TSP. Ale ukazuje se, že to není efektivní. Dekodér řeší tu komplikovanou část úlohy, TSP operátory nejsou vhodné.
 - Přímá reprezentace jedince jako celého plánu – klade nároky na specializované (komplexí) evoluční operátory.

Jen základní principy, současné přístupy až v EVA II

NEUROEVOLUCE

Učení NS pomocí EVA

- První pokusy od 80.let
- Učení parametrů (vah)
- Učení struktury (spoje, architektura)
- Učení vah i architektury najednou
- Použití v úlohách reinforcement learning – není možné učit metodami učení s učitelem (robotika)
- Hybridní metody – kombinace EA s lokálním prohledáváním apod

Učení vah

- Přímočaré
 - Zakódování vah do vektoru,
 - floating point GA,
 - standardní operátory
- Většinou je pomalejší než specializované gradientní algoritmy (často řádově)
- + Lze ho paralelizovat
- + Lze ho použít i pro úlohy, kde mám fitness, ale ne chybu v každém kroku, takže gradientní algoritmy nemohu použít

Učení struktury

- Fitness = postavit síť, inicializovat, zkusit učit, nejlépe víckrát
- Přímé kódování
 - Realizuji strukturu sítě např. jako binární matici,
 - pracuji s evolucí linearizovaných dlouhých binárních vektorů
- Gramatické kódování
 - Kitano navrhl vyvíjet 2D formální gramatiky, které jsou „programem“ pro vytvoření matice