Introdução a Modelos Teóricos de Algoritmos Genéticos

Alexandre Delbem
Universidade de São Paulo - USP
São Carlos - Brasil

6 de outubro de 2011

Sumário

Objetivo

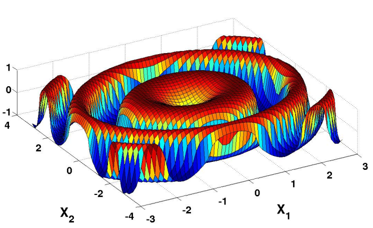
1. Construir modelos teóricos para estimar
   - Parâmetros de Algoritmos Genéticos
   - Tempo de convergência desses algoritmos
2. Propósitos
   - Garantir qualidade da solução
   - Determinar classes de problemas em que se garante isso
3. Efeitos colaterais dos modelos teóricos
   - Guia para o projeto de algoritmos genéticos
   - Suporte para o desenvolvimento de metaheurísticas em geral
   - Revisão do próprio processo de modelagem teórica de algoritmos genéticos

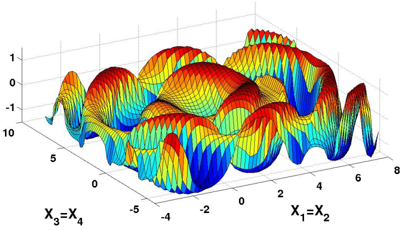
Técnica de Modelagem

1. A modelagem envolve
   - Escolha de um Algoritmo Genético
     - Minimal
     - Representativo de sua classe
     - Capaz de resolver problemas relativamente complexos
   - Escolha de problemas
     - Relativamente complexos
     - Com fácil controle da complexidade deles: Número de modos, Escala
     - Que enfatizam a característica de um algoritmo que os soluciona
2. 1) Quais seriam então as características do algoritmo ?
3. 2) Quais seriam esses problemas relativamente complexos ?
   - ! Começar pelos problemas ...

Problemas relativamente complexos

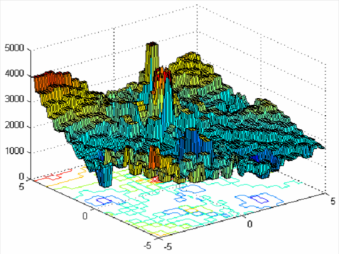
1. Funções multimodais

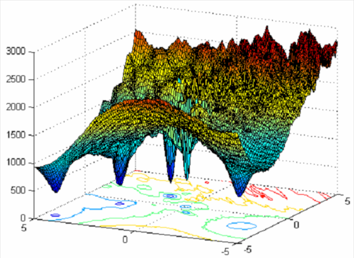




Problemas relativamente complexos

1. Essas superfícies podem ser bem mais complicadas





2. E considerando mais algumas dimensões ...

Problemas relativamente complexos

1. Nessas superfícies em geral o AG patina
2. Aumentar a taxa de crossover pode não resolver
   - Não explora fora das regiões já amostradas
   - Pode gerar muitas combinações de baixa qualidade
   - Aumentar a pressão de seleção para evitar isso
     - Restringe o espaço de busca a um conjunto menor de regiões promissoras
     - Perda do ótimo nesse caso não é baixa em superfícies multimodais

Problemas relativamente complexos

1. Por fim, pode-se aumentar a taxa de mutação
   - Também pode gerar muitas soluções de baixa qualidade
   - Nesse caso, pode-se aumentar a pressão de seleção
     - A mutação poderia reamostrar em regiões desimadas pela alta pressão de seleção
2. Porém, a taxa de mutação pode precisar ser alta
   - Ao ponto que o AG aproxima-se de uma busca aleatória

Problemas relativamente complexos

1. Há modelagens teóricas que descrevem as relações entre esses parâmetros para certas classes de problemas [1]

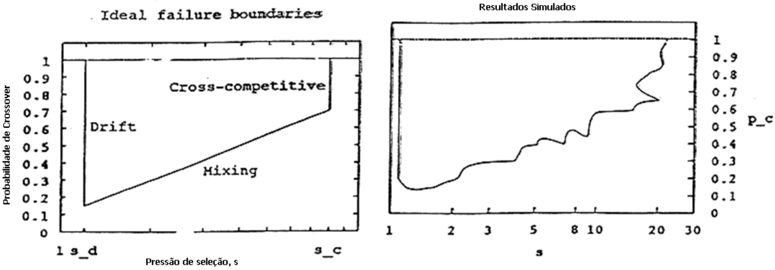


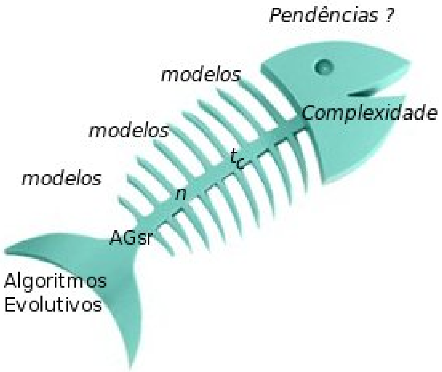
Figura: Mapas de controle de parâmetros teórico e experimental

2. **Não serão tratadas aqui !**

[1]  Goldberg, D. E. The Design of Innovation: Lessons from and for Competent Genetic Algorithms, 2002

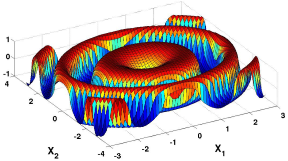
Problemas relativamente complexos

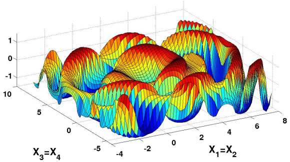
1. Aliás há vários outros modelos que não serão apresentados aqui !
2. Tentarei extrair o que seria a **espinha dorsal** de uma estrutura de modelos
    - Que seja simples para se começar a trabalhar

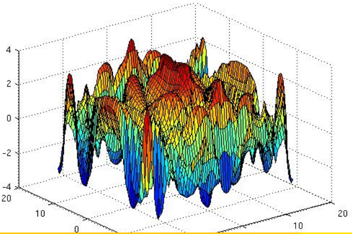


Relações entre problema e algoritmo

1. Além da briga entre parâmetros do AG e variações de operadores
   - Há outro caminho para lidar com problemas complexos é tentar identificar os seus suproblemas
2. Analizando novamente aquelas funções multimodais







Relações entre problema e algoritmo

1. Usando mutação baixa (ou sem) para não tornar a busca aleatória
   - É preciso amostrar muito bem de início
   - E/ou fazer um busca exaustiva para cada subproblema
2. Se os subproblemas forem pequenos, pode dar certo
3. Casos de subproblemas maiores, **fica para depois!**
4. Resolver os com subproblemas pequenos já tem alguma utilidade!
5. Seguindo esse caminho
   - O crossover não deve trabalhar dentro de um subproblema
   - Somente entre subproblemas
   - É o que chamarei de **crossover ideal!**

Relações entre problema e algoritmo

1. Observe que as variáveis de subproblemas são fortemente correlacionadas no sentido de Planejamento de Experimentos [2]
2. São dependentes
   - Conclusão: o crossover é adequado quando há certa independência entre as variáveis
3. No exemplo multimodal $\sqrt{seno(x_1^2 + x_2^2)}$
   - Chamando $x_1|x_2$ de y_1 e $x_3|x_4$ de y_2, então o crossover seria adequado
   - Trabalharia sobre **"metavariáveis"**

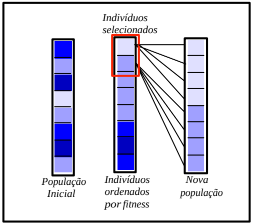
[2]  Montgomery, D.C., Design and Analysis of Experiments, 1976

Algoritmo genético selecto-recombinativo

1. Em síntese, esse AG resume-se em um AGsr
   - População de tamanho fixo
   - Seleção
   - Crossover
2. Toda diversidade deve ser gerada na população inicial
   - Consequentemente a Análise teórica será Espacial
     - Mapear adequadamente o espaço de busca em uma geração
   - Análise Temporal
     - Mapeamento ocorre também por mutações ao longo das gerações
     - Mais complicado, abortar!

Algoritmo genético selecto-recombinativo

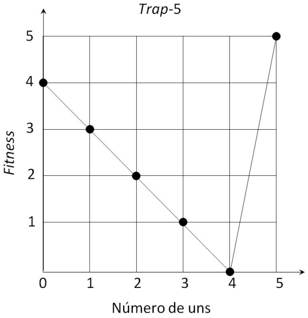
1. AGsr que usarei
   - População de tamanho fixo
   - Seleção por truncamento
   - Crossover ideal
2. Análise da pressão de seleção de forma independente do crossover
   - Ocorre truncamento
   - Nova população é composta por s cópias dos selecionados

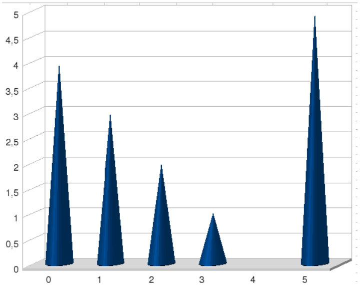


3. Considera-se que se tem um crossover ideal
   - trabalha sobre "metavariáveis"

Relações entre problema e algoritmo

1. Como modelar o efeito de funções multimodais de forma simples ?
   - Ótimos locais dados por variáveis fortemente correlacionadas
   - Uma função simples para criar tais correlações
   - Função deceptiva

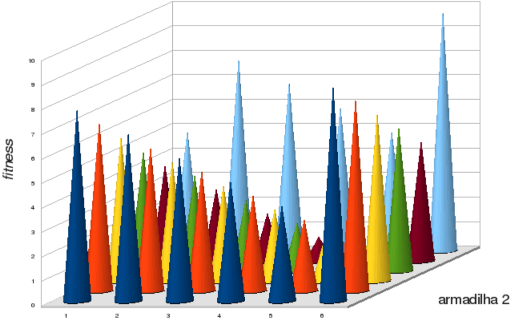




- ótimo: 11111
- ótimo local: 00000

Relações entre problema e algoritmo

1. Relação entre os modos pode ser
   - linear
   - não-linear
   - com sobreposição de variáveis
   - hirárquica
   - combinação dessas, etc.
2. Resultados com linear já é útil
   - Só linear então!



Suposições da modelagem

1. Aspectos assumidos
   - Sabem-se de antemão os subproblemas
   - Crossover sobre metavariáveis
   - Só crossover
   - Alta amostragem dos blocos construtivos
   - Critério de convergência
     - Garantir 100% de convergência correta
       Pode acarretar em parâmetros com valores infinitos para alguns modelos
     - Usa-se o critério de que $\frac{m-1}{m}100\%$ blocos convergiram corretamente
     - Observe que

$$\alpha = c\frac{1}{\ell},$$

$c > 0$ é uma constante
$\ell = km$

Pendências da modelagem

1. E as pendências ?
   - **Ficam para o final !**
2. Anotando-as
   - Basta crossover sobre "metavariáveis" ? Isto é, o que falta para evitar convergência para soluçõe subótimos ?
   - Como determinar os subproblemas ?
   - Como lidar com blocos não-pequenos ?
   - Como lidar com dependências a nível de "metavariáveis" ?

Convergência de múltiplos blocos

1. Supondo que cada subproblema foi bem amostrado, tem
   - 1 1 1 1 1 |##### e
   - ##### | 1 1 1 1 1
2. É preciso que o crossover combine-os gerando 1 1 1 1 1 | 1 1 1 1 1
   - **Antes do algoritmo convergir**
   - Indivíduos com blocos ótimos devem prevalecer dominando a população
3. Porém eles podem ter **algumas instâncias ruins** e perderem para indivíduos com **muitas instâncias subótimas**
   - 0 0 0 0 0 | 0 0 0 0 0 ganhe de ##### | 1 1 1 1 1
   - Com isso, instâncias ótimas podem ser eliminadas da população e o algoritmo convergir para um ótimo local: 0 0 0 0 0 | 0 0 0 0 0
4. **Como evitar esse risco** ?
5. **Em quantas gerações ocorre essa convergência** ?

Reduzindo o risco de ótimos locais

1. Escolha equivocada
   - Competição entre 2 instâncias de um mesmo bloco
     - instâncias ótima e ótima local
     - Há chance de escolhas equivocadas eliminando a instância ótima da população
   - Como esse risco varia ?
     - Com o número de amostras dessas instâncias na população
     - Mais instâncias ótimas, mais improvável errar
     - Amostras da ótima na população inicial: $a = n\frac{1}{2^k}$
   - O risco é alterado a cada geração pela mudança na proporção de instâncias ótimas
   - Qual é o risco de peder todas as instâncias ótimas em g gerações ?
2. Similar ao problema da Ruína do Jogador

Reduzindo o risco de ótimos locais

1. O Problema da Ruína do Jogador

Reduzindo o risco de ótimos locais

1. p (q) é probabilidade do Pedro (Bianca) ganhar uma jogada, sair cara
2. $p = 1 - q = 0,5$
3. Moeda viciada: $p > 0,5$
4. Moeda com 2 caras: $p = 1,0$

5. A probabilidade do sucesso (P_n) do Pedro é
   - $P_n = \frac{1-(q/p)^a}{1-(q/p)^n}$  [3]
   - a é a quantidade de notas de R$10 do Pedro
   - n é o total de notas de R$10 no jogo

[3] Christiaan Huygens, também Pascal, Bernoulli, Montmort, Hudde. Ver Edward, A.W.F., Pascal's Problem: The 'Gamblers's Ruin', International Statistical Review, 1983

Reduzindo o risco de ótimos locais

1. Retornando a competição de instâncias de blocos
   - Pedro = instância ótima: 00000
   - Bianca = instância ótima local: 11111
   - a = total de 00000s
   - n = total de instâncias na população, número de amostras
2. Para $p = q$

$$P_n = \frac{a}{n}$$

3. Na população incial, aleatória,

$$a = n\frac{1}{2^k}$$

4. A pressão de seleção deve fazer $p > q$ em poucas gerações

Reduzindo o risco de ótimos locais

1. Na verdade há mais que 2 jogadores
   - 00000, 00001, 00010, . . . , 11110, 11111
   - Nas primeiras populações
2. Existe solução tratável para certos casos do Problema da Ruína para n jogadores [4]
3. Mas a idéia é
   - Priorizar modelos simples
   - Revelando um aspecto/efeito importante do algoritmo
4. Em poucas gerações
   - 00000 e 11111 dominam o "jogo"
   - As demais instâncias podem ser desprezadas nos cálculos
5. A aproximação para 2 jogadores mostra-se suficiente

[4] Swan, Y.C., Bruss, F.T. A Matrix-Analytic Approach to the N-Player Ruin Problem. Journal of Applied Probability, 2006

Reduzindo o risco de ótimos locais

1. Qual a probabilidade de escolher 00000 (p) ou 11111 (q) ?
2. É afetada diretamente pela distribuição de *fitness* MÉDIO das amostras da população com 11111 (H_1) e 00000 (H_2)

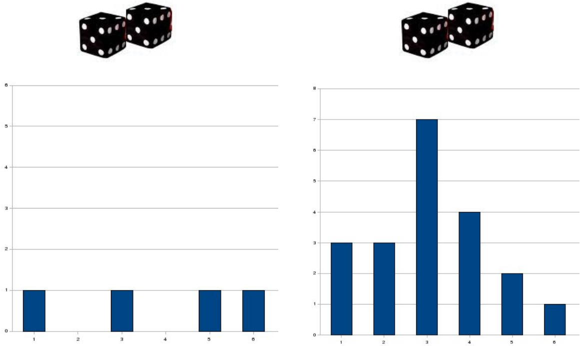


Figura: 4 amostras da MÉDIA de 2 dados (2 BBs): $\sigma = 2,2$
20 amostras da MÉDIA de 2 dados (2 BBs): $\sigma = 1,6$

Reduzindo o risco de ótimos locais

1. Considerando um dos dados viciado, dá 6 $\frac{1}{3}$ das vezes

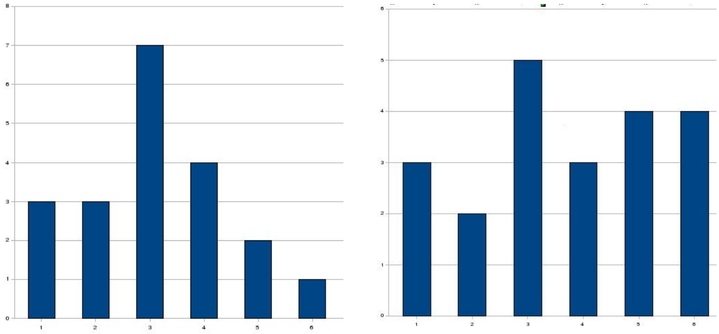
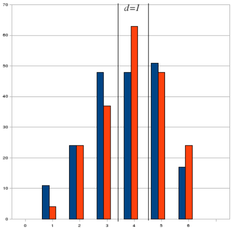
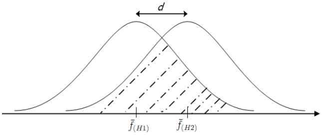


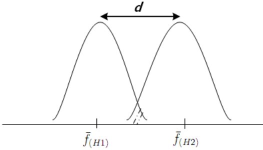
Figura: 20 amostras da MÉDIA de 2 dados (2 BBs): $\mu = 3,1$
20 amostras da MÉDIA de 2 dados (2 BBs): $\mu = 4,1$

Reduzindo o risco de ótimos locais

1. 200 amostras, aumentando, aumentando ...







2. Os indivíduos na área hachurada provocam escolhas equivocadas
3. Deve-se aumentar n para
   - Aumentar d
   - Reduzir σ
4. A probabilidade p de escolha correta aumenta com o aumento da probabilidade de que $\bar{f}_{H_2} > \bar{f}_{H_1}$
   - Depende de n
   - Depende do problema

Reduzindo o risco de ótimos locais

1. Então poderia-se utilizar um teste de hipótese de diferença entre médias, o que resultaria em
$$p = \mathbb{N}\left(\frac{d}{\sqrt{\sigma^2_{H_1} + \sigma^2_{H_2}}}\right)$$
2. Mas o desvio padrão ainda sofre o efeito de outros $m-1$ blocos, então
$$p = \mathbb{N}\left(\frac{d}{\sqrt{\sigma^2_{H_1^1} + \sigma^2_{H_2^1} + \quad \dots \quad + \sigma^2_{H_1^m} + \sigma^2_{H_2^m}}}\right)$$
3. Suponha que o desvio padrão de cada bloco aproxima-se do desvio padrão médio σ_{BB} em todos os blocos
$$p = \mathbb{N}\left(\frac{d}{\sqrt{2m\sigma^2_{BB}}}\right)$$
4. Usando uma expansão em série de potências para $\mathbb{N}$
$$p = \frac{1}{2} + \frac{1}{2}\frac{d}{\sigma_{BB}\sqrt{2m}}$$

Reduzindo o risco de ótimos locais

1. Voltando à equação

$$P_n = \frac{1-(q/p)^a}{1-(q/p)^n}$$

2. Substituindo $q = 1 - p$, $a = \frac{n}{2^k}$ e observando que o denominador aproxima-se de 1 antes que o numerador

$$P_n \approx 1 - (\frac{1-p}{p})^{\frac{n}{2^k}}$$

3. Lembrando que $p = \frac{1}{2} + \frac{1}{2}x$, $x = \frac{d}{\sigma_{BB}\sqrt{2m}}$,

$P_n \approx 1 - \left(\frac{\frac{1}{2}-\frac{1}{2}x}{\frac{1}{2}+\frac{1}{2}x}\right)^{\frac{n}{2^k}}$ ou

$$1 - P_n = \left(\frac{1-x}{1+x}\right)^{\frac{n}{2^k}}$$

Reduzindo o risco de ótimos locais

1. Sendo $\alpha = 1 - P_n$ a probabilidade de fracasso e aplicando ln

$$\ln \alpha \approx \frac{n}{2^k} \ln \left(\frac{1-x}{1+x} \right)$$

ou

$$\ln \alpha \approx \frac{n}{2^k} \left(\ln(1-x) - \ln(1+x) \right)$$

2. Usando que $\ln(1-x) \approx -x$ e $\ln(1+x) \approx x$

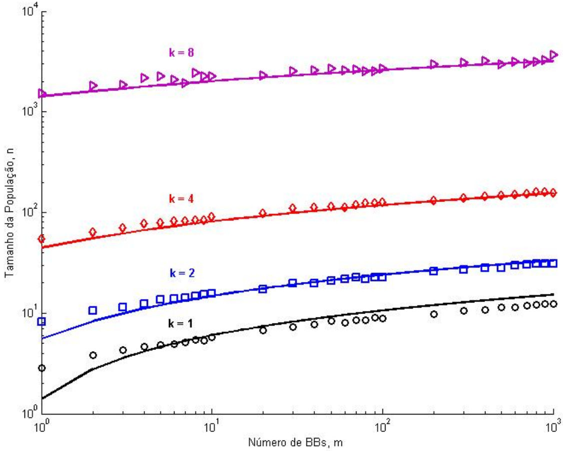
$$\ln \alpha \approx \frac{n}{2^k}(-2x)$$

3. $n \approx -\ln \alpha 2^{k-1} \frac{1}{x}$ ou

$$n \approx -\ln \alpha 2^{k-1} \frac{\sigma_{BB}\sqrt{2m}}{d}$$

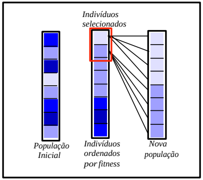
Reduzindo o risco de ótimos locais

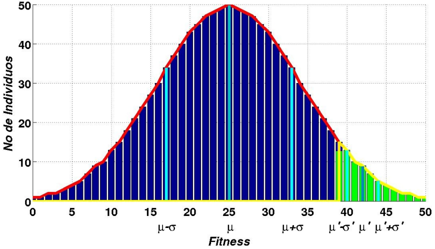
1. Comparação do modelo com resultados experimentais



Reduzindo o risco de ótimos locais

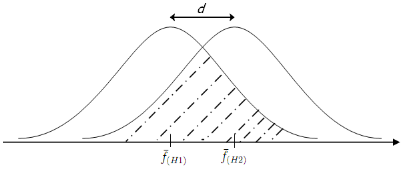
1. Ruído no *fitness*
   - O ruído pode ser modelado como interferência de sinais externos sobre o sinal de cada instância (ótima e ótima local) de cada bloco
   - Em processamento de sinais, esse efeito pode ser modelado pela soma das variâncias dos sinais devido a interferências externas, σ_x^2
   - $p = \frac{1}{2} + \frac{1}{2}\frac{d}{\sqrt{2m(\sigma_{BB}+\sigma_x^2)}}$
2. Efeito da pressão de seleção *s*

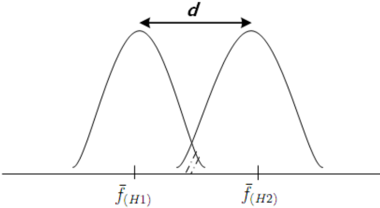




Reduzindo o risco de ótimos locais

1. Efeito da pressão de seleção s
   - s: número de cópias dos selecionados para compor a nova população
   - $1/s$: tamanho da fatia de selecionados
   - Conforme s diminui, d aumenta





   - Portanto, $d' = d + \mathbb{N}^{-1}(\frac{1}{s})\sigma_{BB}$

Tempo para convergir

1. Recordando as questões sobre convergência de múltiplos blocos
   - **1.Como evitar o risco da instância ótima local ganhar** ?

$$n \approx -\ln \alpha 2^{k-1} \frac{\sigma_{BB}\sqrt{2m}}{d}$$

   - **2.Em quantas gerações os blocos convergem** ?
2. Resposta para **2**
   - Teorema de Fisher
   - Teorema de Fisher adaptado para truncamento
   - Limitantes inferior e superior com base em problemas extremos
     - Mínima interferência entre blocos
       Problema UmMax
     - Interferência exponencial entre blocos
       Problema BinInt

Teorema fundamental de Fisher da seleção natural

1. "A taxa de crescimento do *fitness* de um organismo em qualquer tempo é igual ao efeito da variância genética sobre o *fitness* em qualquer tempo"

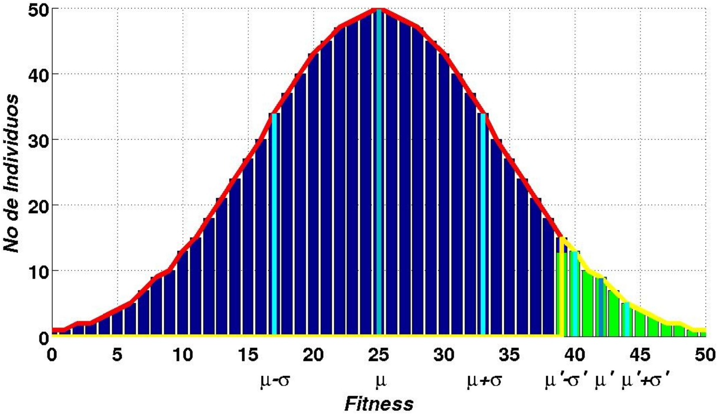
Figura: Fisher 1929, estatístico e biólogo

2. $\bar{f}_{t+1} - \bar{f}_t = \frac{\sigma_t^2}{\bar{f}_t}$ [5]
   - σ_t é a variância do *fitness* na população corrente

[5] Ver dedução simples em http://darwin.eeb.uconn.edu/eeb348/lecturenotes/quant-evolution/node4.html

Teorema fundamental de Fisher da seleção natural

1. Efeito de s por truncamento sobre $\mathbb{N}$



Teorema fundamental de Fisher da seleção natural

1. s gera praticamente uma $\mathbb{N}$ truncada pela esquerda
2. $\bar{f}_{t+1}$ fica próximo do limite esquerdo
3. A razão $\frac{\sigma_t}{\bar{f}_{t+1}}$ [6] aproxima-se de uma constante para cada s
4. Essa constante é chamada de intensidade de seleção, $\mathcal{I}$
5. Assim

$$\bar{f}_{t+1} - \bar{f}_t = \mathcal{I}\sigma_t$$

[6] Há estudos para modelar melhor essa relação

Tempo para convergir do UmMax

1. $fo(x) = \sum_i^l |x_i^k - x_i^O|$
2. Resolvido rapidamente por um algoritmo de busca local
3. Ótimo em tempo linear: $O(\ell)$
4. Considere $x_O = (1, \ldots, 1)$
    - O *fitness* cresce conforme o número de 1s aumenta

$$f(x) = \sum_{i=1}^{l} x_i$$

Tempo para convergir do UmMax

1. A variância de $f(x)$, σ_t^2 na geração t podem ser obtidos pelos respectivos momentos de uma distribuição de Benoulli
   - *string* x pode ser vista como um conjunto de ℓ experimentos com resultado aleatório (0 ou 1) para cada expeirmento x_i
   - $\bar{f}_t = \ell p_t$
   - $\sigma_t^2 = \ell p_t(1 - p_t)$
2. Como $f_t = \ell p_t$
$$f_{t+1} = \ell p_{t+1}$$
3. Aplicando a Equação de Fisher para truncamento
$$f_{t+1} - f_t = \mathcal{I}\sigma_t$$
$$\ell p_{t+1} - \ell p_t = \mathcal{I}\sigma_t$$
$$p_{t+1} - p_t = \frac{\mathcal{I}}{\ell}\sigma_t$$

Tempo para convergir do UmMax

1. Substituindo $\sigma_t = \sqrt{\ell p_t(1-p_t)}$

$$p_{t+1} - p_t = \frac{\mathcal{I}}{\ell}\sqrt{\ell p_t(1-p_t)}$$

$$p_{t+1} - p_t = \frac{\mathcal{I}}{\sqrt{\ell}}\sqrt{p_t(1-p_t)}$$

2. Aproximando a equação de diferenças por uma equação diferencial

$$\frac{\partial p_t}{\partial t} = \frac{\mathcal{I}}{\sqrt{\ell}}\sqrt{p_t(1-p_t)}$$

3. Uma solução aproximada é

$$p_t = Acos^2\left(\frac{\mathcal{I}}{\sqrt{\ell}}t\right)$$

4. Para $t = 0$, $p_0 = \frac{1}{2} \rightarrow A = \frac{1}{2}$ e

$$p_t = \frac{1}{2}cos^2\left(\frac{\mathcal{I}}{\sqrt{\ell}}t\right)$$

Tempo para convergir do UmMax

1. Então

$$cos\left(\frac{\mathcal{I}}{\sqrt{\ell}}t\right) = \pm\sqrt{2p_t}$$

$$\frac{\mathcal{I}}{\sqrt{\ell}}t = arccos\left(\sqrt{2p_t}\right)$$

2. $arccos(z) \approx \frac{\pi}{2} - z$

$$\frac{\mathcal{I}}{\sqrt{\ell}}t \approx \frac{\pi}{2} - \sqrt{2p_t}$$

$$t \approx \frac{\sqrt{\ell}}{\mathcal{I}}\frac{\pi}{2} - \sqrt{2p_t}$$

Tempo para convergir do UmMax

1. Usando o critério de convergência do AGsr $p_t = 1 - \frac{1}{n}$

$$t \approx \frac{\sqrt{\ell}}{\mathcal{I}}\frac{\pi}{2} - \sqrt{2(1 - \frac{1}{n})}$$

2. Portanto

$$t_g = \frac{\sqrt{\ell}}{\mathcal{I}}\frac{\pi}{2}$$

Tempo para convergir do BinInt

1. Maximizar a função

$$f(x_j) = \sum_{j=1}^{\ell} 2^{\ell-j} x_j$$

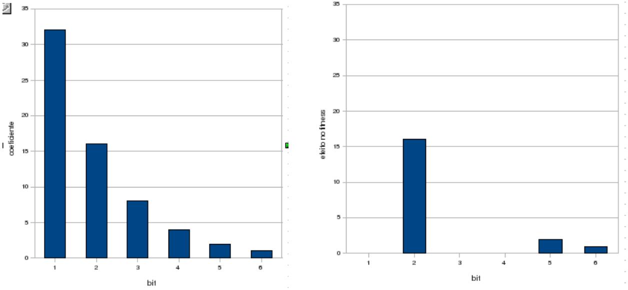
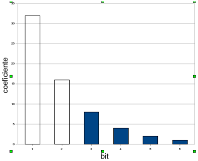


Figura: Coeficientes para $\ell = 6$ e contribuição dos *bits* de 010011

Tempo para convergir do BinInt

1. Os BBs mais salientes convergiram primeiro [7]
2. Inicialmente a população é aleatória, RRRRRR, λ=0
3. Para 1 RRRRR, $\lambda = 1$
4. Para 1 1 1 1 1 1, $\lambda = \ell = 6$
5. Dado λ, $f(x_j) = \sum_{j=1}^{\lambda} 2^{\ell-j} + \sum_{j=\lambda+1}^{\ell} 2^{\ell-j} x_j = \mathbb{C}_\lambda + \sum_{j=\lambda+1}^{\ell} 2^{\ell-j} x_j$
    - O coeficiente $c_{\lambda+1}$ é o que FAZ A DIFERENÇA
6. $\lambda = 2$



- 111010, $f(x) = 48 + 2^3 0 + 2^2 0 + 2^1 1 + 2^0 0 = 50$
- 111010, $f(x) = 48 + 2^3 0 + 2^2 0 + 2^1 0 + 2^0 1 = 51$
- 111101, $f(x) = 48 + 2^3 1 + 2^2 0 + 2^1 0 + 2^0 0 = 64$

[7] Na competição entre blocos o fator crítico é a escala de cada coeficiente e não o tamanho do bloco. Assim é suficiente estudar blocos de 1 *bit*

Tempo para convergir do BinInt

1. Para a média
   - Os *loci* que convergiram estão em 1
   - Os que não convergiram tem valor esperado $\frac{1}{2}$
2. $\overline{f}(\lambda) = 1\sum_{i=1}^{\lambda} 2^{\ell-i} + \frac{1}{2}\sum_{\lambda+1}^{\ell} 2^{\ell-i}$

$$= 1\sum_{j=\ell-1}^{\ell-\lambda} 2^j + \frac{1}{2}\sum_{j=\ell-\lambda-1}^{0} 2^j$$

$$= 1\sum_{j=\ell-\lambda}^{\ell-1} 2^j + \frac{1}{2}\sum_{j=0}^{\ell-\lambda-1} 2^j$$

$$= 1\left(\sum_{j=0}^{\ell-1} 2^j - \sum_{j=0}^{\ell-\lambda-1} 2^j\right) + \frac{1}{2}\sum_{j=0}^{\ell-\lambda-1} 2^j$$

Tempo para convergir do BinInt

1. Portanto

$$\overline{f}(\lambda) = 1\left(2^{\ell-1+1} - 2^{\ell-\lambda-1+1}\right) + 2^{-1}2^{\ell-\lambda-1+1}$$

$$\overline{f}(\lambda) = 2^{\ell-1}\left(2 - 2^{-\lambda+1} - 2^{-\ell}\right)$$

2. $\sigma^2 = \overline{f^2(\lambda)} - \overline{f(\lambda)}^2$
3. A variância dos *bits* de $j = 0$ a $j = \lambda$ é zero
4. A variãncia total é dos *bits* de $j = \lambda + 1$ a $j = \ell$
5. $f(\lambda) = \mathbb{C}_\lambda + \sum_{j=0}^{\ell} 2^{\ell-\lambda-1} x_j$
6. $f'(\lambda) = f(\lambda) - \mathbb{C}_\lambda = \sum_{j=0}^{\ell} 2^{\ell-\lambda-1} x_j$

Tempo para convergir do BinInt

1. $\overline{f'(\lambda)}$ é o valor médio da soma de todos os números inteiros representáveis por $\ell - \lambda - 1$ *bits*

$$\frac{\sum_{i=0}^{2^{\ell-\lambda}-1} i}{2^{\ell-\lambda}}$$

2. Basta então usar $\sum_{j=0}^{n} i = \frac{n(n+1)}{2}$ para obter $\overline{f'(\lambda)}^2$
3. De forma similar obtém-se $\overline{f'(\lambda)^2}$ usando $\sum_{j=0}^{n} j^2 = \frac{n(n+1)(2n+1)}{6}$
4. Isso resulta em

$$\sigma(\lambda) = \frac{2^{\ell-\lambda-1}}{\sqrt{3}}$$

Tempo para convergir do BinInt

1. Usando o Teorema de Fisher para truncamento

$$\overline{f}_{t+1} - \overline{f}_t = \mathcal{I}\sigma_t$$

$$2^{\ell-1}\left(2 - 2^{-\lambda_{t+1}} - 2^{-\ell}\right) - 2^{\ell-1}\left(2 - 2^{-\lambda_t} - 2^{-\ell}\right) = \mathcal{I}\frac{2^{\ell-\lambda_t-1}}{\sqrt{3}}$$

$$2^{\ell-1}\left(2^{\lambda_t} - 2^{-\lambda_{t+1}}\right) = 2^{\ell-1}\frac{\mathcal{I}}{\sqrt{3}}2^{-\lambda_t}$$

$$\left(2^{\lambda_t} - 2^{-\lambda_{t+1}}\right) = \frac{\mathcal{I}}{\sqrt{3}}2^{-\lambda_t}$$

Tempo para convergir do BinInt

1. Fazendo $\mu = 2^{-\lambda}$, obtém-se uma equação de diferenças

$$\mu_{t+1} - \mu_t = -c\mu_t$$

onde

$$c = \frac{\mathcal{I}}{\sqrt{3}}$$

2. Resolvendo para μ_t, obtém-se uma equação da diferença

$$\mu_{t+1} - \mu_t = -c\mu_t$$

3. Resolvendo para μ_t, obtém-se $\mu_t = \mu_0 \left(n - c\right)^t \; \mu_0 = 2^0 = 1$
4. Assim

$$2^{-\lambda_t} = (1 - c)^t \rightarrow \ln\left(2^{-\lambda_t}\right) = t \ln\left(1 - c\right)$$

$$t_g = \frac{-\lambda_t \ln 2}{\ln(1 - c)} = \frac{-\ell \ln 2}{\ell \ln(1 - c)}$$

Tempo para convergir do BinInt

1. Usando a aproximação $ln\,(1-x) \approx -x$, para x pequeno

$$t_g = \frac{-\ell \ln 2}{-c}$$

2. Como $c = \frac{\mathcal{I}}{\sqrt{3}}$, obtém-se

$$t_g = \frac{\sqrt{3}\ln 2}{\mathcal{I}}\ell$$

Respostas já obtidas

1. Recordando as questões sobre convergência de múltiplos blocos
   - **1. Como evitar que a instância ótima local ganhe ?**

$$n \approx -\ln \alpha 2^{k-1} \frac{\sigma_{BB}\sqrt{2m}}{d}$$

   - **2. Em quantas gerações os blocos convergem ?**
     - 1. Problema UmMax

$$t_g = \frac{\sqrt{\ell}}{\mathcal{I}} \frac{\pi}{2}$$

     - 2. Problema BinInt

$$t_g = \frac{\sqrt{3}\ln 2}{\mathcal{I}} \ell$$

2. Poderia-se obter a **complexidade de tempo** fazendo

$$n.t_g \quad !$$

Complexidade de n

1. Em $n \approx -\ln \alpha 2^{k-1} \frac{\sigma_{BB}\sqrt{2m}}{d}$
   - $m = \frac{\ell}{k}$
   - Para problemas com k e σ_{BB} limitados por uma constante

$$n \leq -c_n \ln \alpha \sqrt{\ell},$$

   $c_n > 0$ é uma constante
2. α depende da precisão estipulada
   - **1.1** Se α for constante

$$n = O(\sqrt{\ell})$$

   - **1.2** Pelo critério de convergência do AGsr, $\alpha = c\frac{1}{\ell}$

$$n = O(\sqrt{\ell} \ln \ell)$$

Complexidade de t_g

1. Em

$$t_g = \frac{\sqrt{\ell}}{\mathcal{I}} \frac{\pi}{2}$$

   e

$$t_g = \frac{\sqrt{3}\ln 2}{\mathcal{I}} \ell$$

   - $\mathcal{I} = \mathcal{I}(s)$ conforme exemplificado
   - Maiores estudos podem ser realizados
   - Assumindo que s independe da tamanho ℓ do problema, $\mathcal{I}$ **é uma constante**

2. **2.1** Com base no Problema UmMax

$$t_g = O(\sqrt{\ell})$$

3. **2.2** Com base no Problema BinInt

$$t_g = O(\ell)$$

Complexidade de tempo do AGsr

1. Combinando os limitantes por meio do produto $n.t_g$
   - menores limitantes **1.1** e **2.1**

$$O(\sqrt{\ell}\sqrt{\ell}) = O(\ell)$$

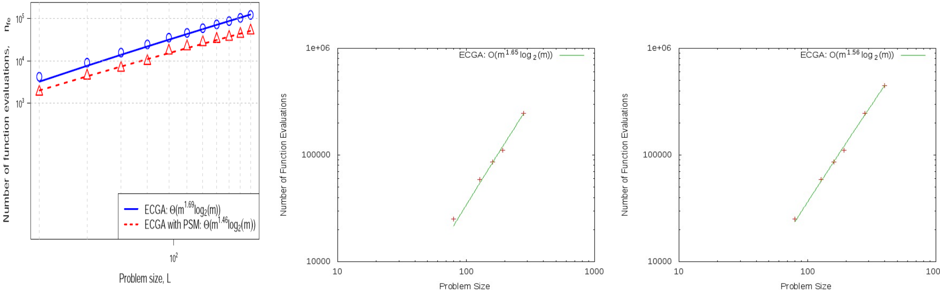
   - maiores limitantes **1.2** e **2.2**

$$t_g = O(\ell\sqrt{\ell}\ln\ell) = O(\ell^{1,5}\ln\ell)$$

2. Portanto **a complexidade do AGsr deve ser subquadrática !**

Complexidade de tempo do AGsr

1. Comparação do modelo com resultados experimentais



Com o aumento da escala dos problemas testados, o expoente aproxima-se do valor teórico: 1,5

1,69 1,65 1,56

Tratando as pendências

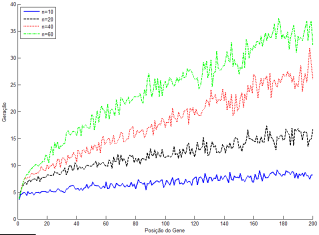
1. Basta crossover sobre "metavariáveis" ? O que falta para evitar convergência para soluções subótimas ?
   - Amostragem suficiente - **OK**
   - Saber se t_g é menor o tempo para convergir ao acaso - à deriva
     - O tempo de deriva [8] é linear com n

$$t_d = c_d n$$

     - MAS t_d é menor que t_g

$$t_g = \ell^{1,5} \ln \ell$$

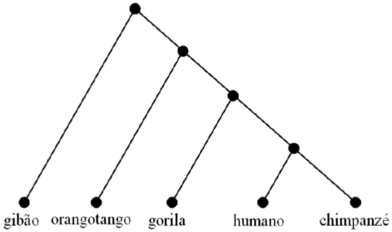
   - Então, SIM pode ocorrer convergência permatura - BinInt

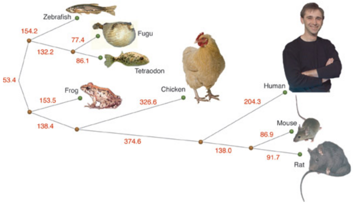


[8] Thierens, D., Goldberg, D.E, Pereira, A.G., Domino convergence, drift, and the

Tratando as pendências

1. **Solução**: convergir em menos gerações
   - Otimização por Divisão - ODiv [9]
     - Tipo o Algoritmo Filogenético - 1 geração [10]





[9] Ver poster do Marcio Kassouf Crocomo na 2a. ELBCE
[10] Ver poster do Danilo Vasconcellos Vargas na 2a. ELBCE

Tratando as pendências

1. Como determinar os subproblemas ?
    - Algoritmos de estimação de distribuição [11] atuais
    - Consegue-se em $m^{1,6} \log_2 m$ avaliações
    - MAS o tempo de execução é relativamente alto devido ao custo de contrução dos modelos probabilísticos
    - **Solução**: modelos probabilísticos mais "leves"
        - Tipo uma Filogenia do Algoritmo Filogenético
    - SOMENTE para linearmente separáveis!
    - **Solução**: Usar modelo probabilístico que EXPLICITE relações hierárquicas entre blocos
        - Tipo a Árvore Filogenética do Algoritmo Filogenético

[11] Livros do Larrañaga e do Pelikan

Tratando as pendências

1. Como lidar com blocos grandes ?
    - Resultados experimentais recentes mostram que o Algoritmo Filogenético lida com $k = 8$ ao invés dos típicos $k = 3, \ 4, \ 5$
2. Mas uma busca exaustiva dentro de blocos grandes pode ser impraticável
    - Cada bloco de *bits* pode ser visto como uma metavariável
    - Um algoritmo de otimização contínua pode ser aplicado então
    - Executando eficientemente a busca dentro de cada bloco [12]

[12] Ver poster do Vinicius Veloso de Melo na 2a. ELBCE

Agradecimentos

1. CNPq
2. FAPESP
3. Thyago Sellmann Pinto Cesar Duque em conjunto com Kumara Sastry e David Goldberg
4. Vinicius Veloso de Melo 
5. Danilo Vasconcellos Vargas 
6. Marcio Kassouf Crocomo 
7. Antonio Helson Mineiro Soares 
8. Marcilyanne Moreira Gois