Metaheurísticas Populacionais

Baseado no livro *METAHEURISTICS - From Design to Implementation* El-Ghazali Talbi

Gustavo Peixoto Silva

23 de Junho de 2014

Conteúdo

Capítulo 1

Metaheurísticas Singulares

1.1 Busca em Vizinhança Variável - *Variable Neighborhood Search- VNS*

VNS é um algoritmo estocático no qual um conjunto de estruturas de vizinhanças N_k $(k = 1, ...n)$ são definidas a priori. Então, cada iteração do algoritmo é composta de três passo: perturbação, busca local e movimento. A cada iteração, uma solução inicial é perturbada na vizinhança corrente N_k. Por exemplo, uma solução x' é gerada randomicamente na vizinhança corrente $N_k(x)$. Um procedimento de busca local é aplicada à solução x' atingindo a solução x''. A solução corrente é trocada pelo novo otimo local x'' se e somente se uma solução melhor foi encontrada, isto é, se $f(x'') < f(x)$. O mesmo procedimento de busca é então reiniciado a partir da solução x'' na primeira vizinhança N_1. Caso contrário, ou seja, se $f(x'') \geq f(x)$, o algoritmo move para a próxima vizinhança, e tenta melhorar a solução corrente. O Algoritmo 1 apresenta um padrão básico do VNS.

Uma implementaçao muito empregada na prática utiliza como busca local a método de descida VND. Esta implementação é apresentada no Algoritmo 2.

Algorithm 1 *Algoritmo VNS Básico para Problema de Minimização*

1: **function** VNS(N_k, k_{max}, x_0)
2: N_k, para $k = 1, ..., k_{max}$ conjunto de vizinhanças
3: $x \leftarrow x_0$; ▷ solução inicial
4: **repeat**
5: $k \leftarrow 1$;
6: **repeat**
7: escolha randomicamente $x' \in N_k(x)$ ▷ perturbação
8: $x'' \leftarrow busca\_local(x')$;
9: **if** $f(x'') < f(x)$ **then**
10: $x \leftarrow x''$;
11: $k \leftarrow 1$;
12: **else**
13: $k \leftarrow k + 1$;
14: **end if**
15: **until** $k = k_{max}$
16: **until** critério de parada seja atingido
17: **return** x
18: **end function**

Algorithm 2 *Algoritmo VNS com busca local VND*

```
1: function VNS(N_k, k_max, N_l, l_max, x_0)
2:     N_k, k = 1, ..., k_max vizinhanças para perturbação
3:     N_l, l = 1, ..., l_max vizinhanças para busca local          ▷ opcional
4:     x ← x_0;                                                     ▷ solução inicial
5:     repeat
6:         k ← 1;
7:         repeat
8:             escolha randomicamente x' ∈ N_k(x);                  ▷ perturbação
9:             l ← 1;                                               ▷ busca local com VND
10:            l_max ← k;                                           ▷ opcional
11:            repeat
12:                encontre o melhor vizinho x'' ∈ N_l(x')
13:                if f(x'') < f(x') then
14:                    x' ← x'';
15:                    l ← 1;
16:                else
17:                    l ← l + 1;
18:                end if
19:            until l ≤ l_max
20:            if f(x'') < f(x) then          ▷ move ou não para outra solução
21:                x ← x"';
22:                k ← 1;
23:            else
24:                k ← k + 1;
25:            end if
26:        until k ≤ k_max
27:    until critério de parada seja atingido
28:    return x;
29: end function
```

Capítulo 2

Metaheurísticas Populacionais

2.1 Introdução

Uma metaheurística populacional inicia o processo de otimização com um conjunto de soluções, denominado população inicial, sendo que cada indivíduo representa uma solução viável para o problema. Iterativamente ela gera novos indivíduos e troca a população corrente por uma nova população de soluções.

Na fase de geração, uma nova população de soluções é gerada a partir da população anterior. Na fase de troca, a seleção acontece considerando a população corrente e a nova população de soluções. As melhores soluções vão compor a nova população. Este processo continua iterativamente até que algum critério de parada seja satisfeito.

Memória de Busca A memória de busca representa o conjunto de informações extraídas e armazenadas durante a busca. Na maioria das metaheurísticas populacionais, a memória de busca se limita à população de soluções. Na Colônia de Formigas temos a matriz de feromônio, enquanto na EDA (*Estimation of Distribution Algorithms*), temos um modelo de probabilidade de aprendizagem que compõe a memória de busca.

Geração Neste passo uma nova população é gerada. Nos algoritmos evolucionários as soluções que compõem a população são selecionadas e reprodu-

zidas usando operadores variacionais, como por exemplo mutação e a recombinação de parte das soluções, ou *genes* de um indivíduo. Estes operadores atuam diretamente nos indivíduos da população. Uma nova solução é construída a partir dos diferentes atributos das soluções pertencentes à população corrente.

Seleção O último passo consiste em selecionar as novas soluções a partir da união da população corrente e a população gerada. A estratégia tradicional elege a população que abada de ser gerada como a nova população. Outra estratégia usa algum tipo *elitismo* nesta fase, sendo selecionadas as melhores soluções dentre as duas populações.

2.2 População Inicial

A definição da população inicial é um fator crucial para a eficiência e eficácia do algoritmo. Na geração da população inicial, o principal critério a se considerar é a diversificação. Se uma população inicial não é bem diversificada, pode ocorrer uma convergência prematura. Por exemplo, isto pode ocorrer se uma população inicial é gerada usando uma heurística gulosa para cada solução da população.

As estratégias utilizadas para inicializar uma população podem ser classificadas em: geração randômica; diversificação sequencial; diversificação paralela e inicialização heurística.

2.2.1 Geração Randômica

Normalmente, a população incial é gerada randomicamente. Em um problema de otimização contínua, por exemplo, o valor real de cada variável pode ser gerado randomicamente dentro do seu intervalo de viabilidade.

Em problemas de otimização discreta, a idéia de inicialização randômica pode ser aplicada aos vetores binários, vetores discretos e permutações, dependendo do problema.

2.2.2 Diversificação Sequencial

A população incial também pode ser gerada de forma uniforme no espaço de viabilidade. Alguns procedimentos não requerem avaliar a função objetivo ou as restrições associadas ao problema. Em uma diversificação sequencial, as soluções são geradas em sequencia de tal forma que a diversificação é otimizada, ou seja, maximizada.

2.2.3 Diversificação Paralela

Na estratégia de diversificação paralela, as soluções de uma população são geradas paralelamente de forma independente.

2.2.4 Initialização Heurística

Qualquer heurística (p.e. busca local) pode ser usada para inicializar uma população, como por exemplo, uma heurística gulosa para o problema. Dependendo da topologia do espaço das soluções do problema, esta estratégia pode ser mais eficiente do que a inicialização randômica. Se esta estratégia for usada, deve haver obviamente uma randomização do processo gulosa para obter diferentes soluções iniciais. A principal desvantagem desta abordagem é que a população inicial pode perder em diversidade, o que irá levar a uma convergência prematura e uma estagnação da população.

2.3 Critério de Parada

Varios critérios de parada são utillizados em algoritmos populacionais. Algumas são semelhantes àqueles utilizados por algoritmos que trabalham com uma única solução. Os critérios de parada são classificados em:

- Procedimentos estáticos: Nestes procedimentos sabe-se a priori o número de iterações do algoritmo. Por exemplo, pode-se usar um número fíxo de iterações (gerações), um tempo máximo de processamento, ou um número máximo de avaliações da função objetivo.

- Procedimentos adaptativos: Neste caso, o final da busca não pode ser conhecido *a priori*. Pode-se utilizar um número fixo de iterações sem melhora ou quando um determinado valor satisfatório para a função objetivo é alcançado

2.4 Algoritmos Evolucionários

Os Algoritmos Evolucionários se baseiam na noção de competição. Eles representam uma classe de algoritmos de otimização iterativos que simulam a evolução das espécies. Eles se baseiam na evolução de uma população de indivíduos. Inicialmente, esta população é gerada por algum método de inicialização (2.2). Cada indivíduo da população é uma versão codificada de uma solução candidata. Uma função objetivo associa um valor de aptidão (*fitness*) a cada indivíduo indicando sua adequabilidade ao problema. A cada passo, indivíduos são selecionados para formar os pais dos próximos indivíduos, seguindo o paradígma da seleção no qual os indivíduos com os melhores valores de *fitness* são selecionados com maior probabilidade. Posteriormente, os indivíduos selecionados se reproduzem usando operadores variacionais (cruzamente, mutação) para gerar novos descendentes. Finalmente, é aplicado um esquema de troca para determinar quais indivíduos da população irão sobreviver dentre os descendentes e os pais. Esta iteração representa uma nova geração de indivíduos. Este processo se repete iterativamente até que o critério de parada seja atingido.

Nos algoritmos evolucionários, o genótipo representa a codificação enquanto o fenótipo representa a solução propriamente dita. Assim, o genótipo deve ser decodificado para gerar o fenótipo. Os operadores variacionais atuam no nível do genótipo enquanto a função de aptidão (função objetivo) irá usar o fenótipo associado aos indivíduos. A aptidão de um indivíduo mede a capacidade dos indivíduos de sobreviver no seu ambiente. No caso em que a codificação é usada diretamente, o genótipo é similar ao fenótipo. Caso contrário, o genótipo e o fenótipo são estruturas diferentes.

2.4.1 Algoritmos Genéticos

Algortimos Genéticos são uma classe muito popular de Algoritmos Evolucionários. Tradicionalmente, AGs são associados com o uso de representação binária, mas atualmente podem ser encontradas aplicações com outros tipos de representação. Um AG normalmente aplica principalmente um operador de cruzamente a duas soluções, além do operador de mutação randomicamente o conteúdo do indivíduo para promover a diversidade. AGs usam uma probabilidade para selecionar que é originalmente uma seleção proporcional. A troca, ou seleção dos sobreviventes, é a geração, isto é, os pais são substituídos sistematicamente pelos descendentes. A operação de cruzamente é baseada em n-pontos ou em cruzamento uniforme enquanto a mutação é a troca do valor de um *bit*. Normalmente, as probabilidades fixas p_m e p_c são aplicada nos operadores de mutação e de cruzamente repectivamente.

2.5 Conceitos de Algoritmos Evolucionários

Os principais conceitos dos algoritmos evolucionparios são os seguintes:

- **Representação:** Nos algoritmos evolucionários a solução codificada se refere aos *cromossomos*, enquanto as variáveis de decisão dentro da solução são os *genes*. Os possíveis valores das variáveis (genes) são os *alelos* e a posição de um elemento (gene) dentro de um cromossomo é denominda *locus*.

- **População inicial:** Componente comum de busca em todas metaheurísticas populacionais

- **Função objetivo:** Comum a todas as metaheuristicas. Em AEs (Algoritmos Evolucionários), o termo *fitness* se refere à função objetivo.

- **Estratégia de seleção:** A estratégia de seleção diz respeito às seguintes questões: "Quais os progenitores devem ser escolhidos para gerar a próxima geração de tal forma que melhore o *fitness*?"

- **Estratégia reprodutiva:** A estratégia consiste em definir operadores de mutação e de cruzamento adequados para gerar novos indivíduos (descendentes).

- **Estratégia de troca:** Os novos descendentes competem por seu espaço com os indivíduos antigos na próxima geração. Este processo de sobrevivência envolve o valor da fo.

- **Critério de parada:** Comum a todas as metaheuristicas. Existem critérios de parada específicos de metaheurísticas populacionais.

2.5.1 Métodos de Seleção

O mecanismo de seleção é uma das principais componenetes de busca dos AEs. O princípio do método de seleção é "quanto melhor for o indivíduo, maiores as chances dele se tornar um reprodutor". Esta estratégia de seleção levar a população a soluções melhores. Entretanto, indivíduos piores não devem ser descartados e eles devem ter uma chance de ser selecionados. Isto pode levar a um material genético útil para o processo. A estratégia de seleção determina quais os indivíduos que irão se reproduzir e quantos filhos cada indivíduo selecionado irá produzir. Em AEs, pode haver duas maneiras distintas de determinar o *fitness* de cada indivíduo.

- *Atribuição de fitness proporcional* no qual o valor absoluto da FO é associada a cada indivíduo.

- *Atribuição de fitness baseada na classificação* no qual um valor relativo da FO é associado a cada indivíduo. Por exemplo, uma classificação da população está associado a cada indivíduo de acordo com a sua classificação em uma ordenação decrescente dos indivíduos. Então, os pais são selecionados de acordo com o *fitness* por meio de uma das seguintes estratégias: seleção da roleta, sorteio estocástico universal (SUS), seleção por torneio, e seleção baseada em classificação.

2.5.1.1 Seleção por Roleta

Este é a estratégia de seleção mais comum na literatura. Ela irá designar a cada indivíduo uma probabilidade de seleção proporcional ao seu *fitness* relativo. Seja f_i o *fitness* do indivíduo p_i na população P. Sua probabilidade de ser selecionado é $p_i = f_i/(\sum_{j=1}^{n} f_j)$. Suponha um gráfico de pizza onde a cada indivíduo é atribuída uma área proporcional ao seu *fitness* e uma roleta é colocada ao redor desta pizza. A seleção de μ indivíduos é realizada por meio de μ rodadas independentes da pizza. Cada rodada seleciona um indivíduo. Assim, os indivíduos melhores tem maiores chances de serem escolhidos. Na seleção por roleta, indivíduos fora da média introduzirão um viés no início da pesquisa, que pode causar uma convergência prematura e uma perda de diversidade.

2.5.1.2 Seleção Estocástica Universal

Para reduzir o viés da estratégia de seleção por roleta, a seleção estocástica universal pode ser usada. Neste caso μ indicadores igualmente espaçados são distribuídos ao redor da pizza. E com uma única rodada são selecionados todos os pais.

2.5.1.3 Seleção por Torneio

A seleção por torneio consiste em selecionar randomicamente k indivíduos; o parâmetro k é dito o tamanho do grupo do torneio. Um torneio é então aplicado aos k membros do grupo para selecionar o melhor. Para selecionar μ indivíduos, o procedimento deve ser repetido μ vezes.

2.5.1.4 Seleção Baseada em Classificação

Ao invés de usar o valor *fitness* individual, a classificação dos indivíduos é usada. A função privilegia os indivíduos com classificação alta (i.e. bom *fitness*). A classificação pode ser linearmente escalada usando a seguinte fórmula:

$$P(i) = \frac{2-s}{\mu} + \frac{2.r(i)(s-1)}{\mu(\mu-1)}$$

onde s é pressão de seleção ($1.0 < s \leq 2.0$), μ é o tamanho da população, e $r(i)$ é a classificação associada com o indivíduo i. Quanto maior for a pressão de seleção s, mais importância será dada aos melhores indivíduos.

2.5.2 Reprodução

Uma vez que foi realizada a seleção do indivíduos para formar os pais, a regra da fase de reprodução consiste na aplicação dos operadores variacionais tais como mutação (operador unário) e cruzamente (operador binário).

2.5.2.1 Mutação

Operadores de mutação são unários atuando em um único indivíduo. Mutações representam pequenas modificações de indivíduos selecionados da população. O parâmetro p_m define a probabilidade de modificar cada elemento (gene) da representação. Ela pode afetar apenas um gene. Em geral, pequenos valores são recomendados para esta probabilidade ($p_m \in [0.001, 0.01]$). Algumas estratégias inicializam a probabilidade de mutação em $1/k$, onde k é o número de variáveis de decisão. Assim, em média apenas uma variável sofre mutação.

Alguns pontos importantes que devem ser levados em conta no desenvolvimento e uso de um operador de mutação são:

- **Abrangência:** O operador mutação deve permitir que toda solução do espaço de busca seja alcançada.
- **Validade:** O operador deve produzir soluções viáveis. Isto nem sempre é possível para problemas muito restritos.
- **Localidade:** Uma mutação deve produzir mudanças mínimas no indivíduo. A intensidade da mutação deve ser controlável. Assim como uma vizinhança nas metaheurísticas singulares (MSs), a principal propriedade que deve caracterizar um operador de mutação é localidade.

Localidade pode ser visto como o efeito na solução (fenótipo) quanto é realizado um movimento (perturbação) na sua representação (genótipo). Quando pequenas modificação são feitas no genótipo, o fenótipo deve revelar pequenas mudanças. Neste caso, a mutação é dita ter uma forte localidade. Portanto, um AE (Algoritmo Evolucionário) irá realizar uma pesquisa significativa no espaço de soluções do problema. Uma localidade fraca é caracterizada por um grande efeito no fenótipo quando pequenas modificações são feitas no genótipo. Em casos extremos, a busca irá convergir para uma busca aleatória no espaço de busca.

A mutação nos AEs está relacionada aos operadores de vizinhança das MSs. Portanto, a definição das estruturas de vizinhança podem ser utilzadas como operadores de mutação. Ou seja, os seguintes tipos de movimentos:

- **Mutação na representação binária:** A mutação mais comumente usada da é o operador de troca (*flip*).
- **Mutação em representação discreta:** Consiste em troca o valor associado com um elemento por outro valor do "alfabeto".
- **Mutação em permutações:** Mutação em representações baseadas em ordenações são geralmente baseadas em operadores de troca, inversão e inserção de genes.

2.5.2.2 Recombinação ou Cruzamento

Diferentemente de operadores unários tais como mutação, o operador cruzamento é binário e às vezes n-ário. A regra dos operadores cruzamento é herdar algumas características dos dois pais para gerar dos descendentes. Assim como o operador mutação, a definição de um operador cruzamento depende principalmente da representação utilizada. Diferentemente das MSs, onde os operadores de busca são sempre unários, estes operadores foram desenvolvidos especificamente para os AEs.

Alguns pontos importantes devem ser levados em consideração ao desenvolver e usar operadores de cruzamento:

- **Hereditariedade:** A principal característica de um operador de cruzamento é a hereditariedade. O operador cruzamento deve herdar um material genétcio de ambos os pais. Um operador é uma recombinação pura (fortemente hereditário) se dois indivíduos idênticos gerem filhos idênticos. Portanto, os operadores mutação e cruzamento são complementares. Neste caso, a mutação traz alguma diversidade nos indivíduos introduzindo alguns valores perdidos nos indivíduos da população corrente.

  Um operador cruzamento Ox é *respeitoso* se as decisões comuns em ambos os pais forem preservadas, a *não classificador* se a distância entre os pais(p_1, p_2) e o descendente o é menor do que a distância entre os pais:

  $$d(p_1, o) \leq d(p_1, p_2) \text{ and } d(p_2, o) \leq d(p_1, p_2), \forall o \in O(p_1, p_2, Ox)$$

  onde $O(p_1, p_2, Ox)$ é o conjunto de possíveis descendentes gerados pelo operador cruzamento Ox

- **Validade:** The crossover operator should produce valid solutions. This is not always possible for constrained optimization problems.

A taxa de cruzamento p_c ($p_c \in [0, 1]$) representa a proporção dos pais aos quais o operador cruzamento será aplicado. O melhor valor para o parâmetro p_c está relacionado com outros parâmetros como o tamanho da população, a probabilidade de mutação e o processo de seleção. O intervalo mais comumente usado é [0.45, 0.95]. Técnicas adaptativas para o cruzamento também podem ser utilizadas.

O operador básico de cruzamento é o 1-ponto e sua generalização é o n-pontos cruzamento. Estes operadores foram inicialmente propostos para representações binárias. No operador cruzamento 1-ponto, um ponto de cruzamento k é escolhido randômicamente, e dois filhos são criados trocando os dois segmentos do pais. Em geral, uma distribuição randômica uniforme é usada para selecionar o ponto k de corte.

Exemplo: Considerando apenas um ponto de cruzamento teremos: pai1 = (10011|1101), pai2 = (01101|0001) → filho1 = (10011|0001), filho2 = (01101|1101). Para dois pontos de corte teremos: pai1 = (10|0111|101), pai2 = (01|1010|001) → filho1 =(10|1010|101), filho2 =(01|0111|001).

Utilizando o cruzamento uniforme, 2 indivíduos podem ser recombinados sem levar em conta o tamanho dos segmentos. Cada elemento dos descendentes é selecionado randômicamente de cada pai e cada pai irá contribuir igualmente para gerar os filhos.

Para representações em valores reais, além dos cruzamento anteriores, são também utilizado os seguintes operadores: recombinação centrada na média e recombinação centrada nos pais. Na recombinação centrada na média, os descendentes são gerados próximos aos centróide de seus pais. Na recombinação centrada nos pais, os filhos são gerados próximos aos pais. A cada pais é atribuída uma probabilidade igual de criar um descendente em sua vizinhança.

Aplicando os operadores clássicos de cruzamento a permutações podem ser geradas soluções inviáveis. Portanto vários operadores de cruzamento de permutações foram desenvolvido, como por exemplo:

- **Order crossover (OX):** Primeiro, dois pontos de cruzamento são selecionados randômicamente. Do pai 1, será copiado no filho, na mesma posição, a parte entre os dois pontos. No pai 2, será iniciada uma busca a partir do segundo ponto de cruzamento e serão tomados os elementos que ainda não foram selecionados do pai 1. Estes elementos entrarão no descendente, a partir do segundo ponto de cruzamento. O operador de cruzamento OX é um operador puramente de recombinação. Se o preenchimento for iniciado no primeiro ponto de cruzamento, o operador não será puro. Do pai 1, a order relativa, a adjacência e as posições absolutas são preservadas. Do pai 2, apenas a ordem relativa é preservada.

  Exemplo: Considere os seguintes descendentes com os pontos de corte já definidos: pai 1 (AB|CDEF|GHI), pai 2 (hd|aeic|fbg). Pai 1 → $\ldots CDEF \ldots$, pai 2 → $bghai$. Devemos completar o descendente a

partir da carga herdada do pa1 1, preenchendo as posições que faltam com o material do pai 2, iniciando pela posição imediatamente após o segundo ponto de corte. Assim temos: (ai|CDEF|bgh)

- **Partially mapped crossover (PMX):** Assim como no cruzamento anterior, dois pontos são selecionados randomicamente. Do pai 1 será copiado no descendente, na mesma posição, a parte entre os dois pontos. Percorrer o pai 2, iniciando após o primeiro ponto de cruzamento e preenchendo as primeiras posições intercalando entre o lado direito e o lado esquedo do descendente.

  Exemplo: Considere os seguintes descendentes com os pontos de corte já definidos: pai 1 (AB|CDE|FG), pai 2 (cf|eba|dg). Pai 1 $\rightarrow \ldots CDE \ldots$, pai 2 $\rightarrow bagf$. Devemos completar o descendente a partir da carga herdada do pa1 1, preenchendo as posições que faltam com o material do pai 2, iniciando pela posição imediatamente após o segundo ponto de corte (..|CDE|b.). O próximo elemento da pai 2 será colocado na primeira posição vazia à esquerda (a.|CDE|b.). O próximo elemento do pai 1 irá para a próxima posição vazia à direita (a.|CDE|bg) e finalmente temos o descendente completo (af|CDE|bg).

- **Two-point crossover:** Neste operador, dois pontos são selecionados randomicamente. Os elementos fora dos dois pontos selecionados são herdados de um pai do descendente e os outros elementos são colocados na ordem em que aparecem no outro pai.

  Exemplo: Considere os seguintes descendentes com os pontos de corte já definidos: pai 1 (AB|CDE|FG), pai 2 (cf|eba|dg). Então o descendente fica da seguinte forma: filho=(AB|ced|FG) .

- **Outros cruzamento:** Outros operador cruzamento podem ser usados como o cruzamento de preservação máxima, cruzamento cíclico (CX) que preserva a posição absoluta dos elementos, cruzamento da inclusão, e o cruzamento baseado na posição dos elementos (POS).

2.5.3 Estratégias de Troca

A fase de troca consiste na seleção dos sobreviventes das duas populações, os pais e os seus descendentes. Considerando que o tamanho da população é constante, é necessário retirar indivíduos de acordo com uma certa estratégia de seleção. As estratégias mais extremas são:

- **Troca generalizada:** A troca se refere a toda a população, ou seja, todos os μ indivíduos. A população dos descendentes vai substituir sistematicamente a população dos pais. Esta é estratégia do AG canônico.

- **Troca em estado estacionário:**A cada geração de um AG, apenas um descendente é gerado. Ele pode, por exemplo, substituir o pior indivíduo da população dos pais.

Entre estas duas estratégias extremas, varios esquemas diferentes que consistem em substituir um dado número λ indivíduos da população pode ser aplicado ($1 < \lambda < \mu$). O *elitismo* consiste em selecionar os melhores indivíduos entre os pai e os filhos. Esta abordagem leva a uma rápida convergência, podendo ser prematura. Às vezes, é necessário selecionar indivíduos ruins para evitar o problema de erro de amostragem. Estas trocas podem ser estocásticas ou determinísticas.

Resumindo alguns parâmetros dos AEs, temos:

- **Probabilidade de Mutação p_m:** Um probabilidade de mutação muito elevada irá levar a uma interrupção entre as gerações e a busca fica mais randômica. Normalmente, valores baixos são recomendados para a probabilidade de mutação ($p_m \in [0.001, 0.01)$. Usualmente a probabilidade é inicializada como $1/k$ onde k é o número de variáveis de decisão. Portanto, em média, apenas uma variável sofre mutação.

- **Probabilidade de Cruzamento p_c:** A probabilidade de cruzamento normalmente recebe um valor entre médio e alto, entre [0.3, 0.9].

- **Tamanho da População:** Quanto maior for a população, melhor será a convergência através de boas soluções. Erros de amostragem são mais

sigificativos em populações menores. Entretanto, o tempo de processamento dos AGs cresce linearmente com o tamanho da população. Um equilíbrio deve ser encontrado entre a qualidade da solução obtida e o tempo de processamento do algoritmo. Na prática, o tamanho de uma população varia entre 20 e 100 indivíduos. Alguns resultados teóricos indicam que o tamanho da população deve crescer exponencialmente com o tamanho dos indivíduos, o que leva a uma população muito grande para ser tratada.

http://www.cs.colostate.edu/ genitor/MiscPubs/tutorial.pdf