



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE  
CENTRO DE TECNOLOGIA  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E  
DE COMPUTAÇÃO



# **ALGORITMOS GENÉTICOS: USO DE LÓGICA NEBULOSA E ANÁLISE DE CONVERGÊNCIA POR CADEIA DE MARKOV**

**Luiz Amorim Carlos**

Orientador: Prof. Dr. Aldayr Dantas de Araújo

Co-orientador: Prof. Dr. Daniel Aloise

**Tese de Doutorado** apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação da UFRN (área de concentração: Automação e Sistemas) como parte dos requisitos para obtenção do título de Doutor em Ciências.

Natal, RN, 05 de novembro de 2013

Divisão de Serviços Técnicos

Catálogo da publicação na fonte. UFRN / Biblioteca Central Zila Mamede

Carlos, Luiz Amorim.

Algoritmos genéticos: uso de lógica nebulosa e análise de convergência por cadeia de Markov / Luiz Amorim Carlos - Natal, RN, 2013  
49 f.: il.

Orientador: Aldayr Dantas de Araújo  
Co-orientador: Daniel Aloise

Tese (doutorado) - Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Programa de Pós-Graduação em Engenharia Elétrica e de Computação.

1. Algoritmos - Computação - Tese. 2.  $\text{\LaTeX}$ - Algoritmos genéticos - Modelagem - Tese. I. Araújo, Aldayr Dantas. II. Aloise, Daniel. III. Universidade Federal do Rio Grande do Norte. IV Título.

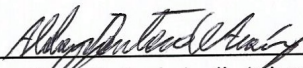
RN/UF/BCZM


CDU 519.254 (043.2)

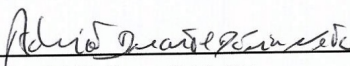
# ALGORITMOS GENÉTICOS: USO DE LÓGICA NEBULOSA E ANÁLISE DE CONVERGÊNCIA POR CADEIA DE MARKOV


**Luiz Amorim Carlos**

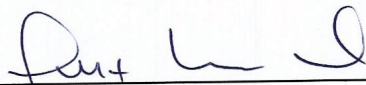
Tese de Doutorado aprovada em 05 de novembro de 2013 pela banca examinadora composta pelos seguintes membros:

  
Prof. Dr. Aldayr Dantas de Araújo (orientador) ..... DEE/UFRN

  
Prof. Ph.D. Daniel Aloise (co-orientador) ..... DCA/UFRN

  
Prof. Dr. Adrião Duarte Dória Neto ..... DCA/UFRN

  
Prof. Dr. Dário José Aloise ..... DI/UERN

  
Prof. Dr. Luiz Satoru Ochi ..... IC/UFF

*A minha esposa Aleca e minhas  
filhas Talita e Natalia, fontes  
renovadoras de minhas forças, pelo  
inestimável apoio e reconfortantes  
estímulos.*

---

# Agradecimentos

---

Ao meu orientador e amigo Professor Aldayr Dantas Araújo pela forma dedicada ao me oferecer sua experiência, seu apoio e incentivos, determinantes para a concretização deste trabalho.

Ao co-orientador Professor Daniel Aloise pela maneira segura e decidida ao assumir esta tarefa de forma tão extemporânea.

Ao Professor André Gustavo, do Depto. de Matemática, pelos recorrentes convites à retomada de trabalhos que havíamos feito e que serviram de base para este trabalho.

À Coordenação do Programa de Pós-Graduação em Engenharia Elétrica e aos serviços de secretaria, pelo apoio.

À UFRN que me recebeu como aluno de graduação e me acolheu no seu quadro de professores, o que tanto tem contribuído para minha formação e realização profissional.

Aos amigos e colegas que sempre depositaram confiança em mim.

Ao Professor Samaherni Moraes Dias, do Departamento de Engenharia Elétrica da UFRN, pela disponibilidade ao resolver algumas das minhas dificuldades de edição.

À minha família que tanto me deu forças manifestadas nos frequentes incentivos, durante esta jornada.

---

# Resumo

---

Neste trabalho, a cadeia de Markov será a ferramenta usada na modelagem e na análise de convergência do algoritmo genético, tanto para sua versão padrão quanto para as demais versões que o algoritmo genético permite. Além disso, pretende-se comparar o desempenho da versão padrão com a versão nebulosa, por acreditar que esta versão dá ao algoritmo genético uma grande capacidade para encontrar um ótimo global, próprio dos algoritmos de otimização global. A escolha deste algoritmo deve-se também ao fato do mesmo ter se tornado, nos últimos anos, uma das ferramentas mais usadas para achar uma solução do problema de otimização. Esta escolha deve-se à sua comprovada eficácia na busca de uma solução de boa qualidade para o problema, considerando que o conhecimento de uma solução de boa qualidade torna-se aceitável tendo em vista que pode não existir um outro algoritmo capaz de obter a solução ótima, para muitos desses problemas. Entretanto, esse algoritmo pode ser definido, levando em conta que o mesmo é dependente não apenas da forma como o problema é representado, mas também como são definidos alguns dos operadores, desde sua versão padrão, quando os parâmetros são mantidos fixos, até suas versões com parâmetros variáveis. Por isso, para se alcançar um bom desempenho com o aludido algoritmo é necessário que o mesmo tenha um adequado critério na escolha de seus parâmetros, principalmente da taxa de mutação e da taxa de cruzamento ou, até mesmo, do tamanho da população. É importante lembrar que as implementações em que parâmetros são mantidos fixos durante toda a execução, a modelagem do algoritmo por cadeia de Markov resulta numa cadeia homogênea, e quando permite a variação de parâmetros ao longo da execução, a cadeia de Markov que o modelo passa a ser do tipo não-homogênea. Portanto, na tentativa de melhorar o desempenho do algoritmo, alguns trabalhos têm procurado realizar o ajuste dos parâmetros através de estratégias que captem características intrínsecas ao problema. Essas características são extraídas do estado presente de execução, com o fim de identificar e preservar algum padrão relacionado a uma solução de boa qualidade e, ao mesmo tempo, descartando aquele padrão de baixa qualidade. As estratégias de extração das características tanto podem usar técnicas precisas quanto técnicas nebulosas, sendo neste último caso feita através de um controlador nebuloso. Com o fim de avaliar empiricamente o desempenho de um algoritmo não-homogêneo, apresenta-se testes onde se compara o algoritmo genético padrão com o algoritmo genético nebuloso, sendo a taxa de mutação ajustada por um controlador nebuloso. Para isso, escolhe-se problemas de otimização cujo número de soluções varia exponencialmente com o número de variáveis.

**Palavras-chave:** Otimização, Cadeia de Markov, Algoritmo Genético e Controlador Nebuloso.

---

# Abstract

---

In this work, the Markov chain will be the tool used in the modeling and analysis of convergence of the genetic algorithm, both the standard version as for the other versions that allows the genetic algorithm. In addition, we intend to compare the performance of the standard version with the fuzzy version, believing that this version gives the genetic algorithm a great ability to find a global optimum, own the global optimization algorithms. The choice of this algorithm is due to the fact that it has become, over the past thirty yares, one of the more importan tool used to find a solution of de optimization problem. This choice is due to its effectiveness in finding a good quality solution to the problem, considering that the knowledge of a good quality solution becomes acceptable given that there may not be another algorithm able to get the optimal solution for many of these problems. However, this algorithm can be set, taking into account, that it is not only dependent on how the problem is represented as but also some of the operators are defined, to the standard version of this, when the parameters are kept fixed, to their versions with variables parameters. Therefore to achieve good performance with the aforementioned algorithm is necessary that it has an adequate criterion in the choice of its parameters, especially the rate of mutation and crossover rate or even the size of the population. It is important to remember that those implementations in which parameters are kept fixed throughout the execution, the modeling algorithm by Markov chain results in a homogeneous chain and when it allows the variation of parameters during the execution, the Markov chain that models becomes be non - homogeneous. Therefore, in an attempt to improve the algorithm performance, few studies have tried to make the setting of the parameters through strategies that capture the intrinsic characteristics of the problem. These characteristics are extracted from the present state of execution, in order to identify and preserve a pattern related to a solution of good quality and at the same time that standard discarding of low quality. Strategies for feature extraction can either use precise techniques as fuzzy techniques, in the latter case being made through a fuzzy controller. A Markov chain is used for modeling and convergence analysis of the algorithm, both in its standard version as for the other. In order to evaluate the performance of a non-homogeneous algorithm tests will be applied to compare the standard fuzzy algorithm with the genetic algorithm, and the rate of change adjusted by a fuzzy controller. To do so, pick up optimization problems whose number of solutions varies exponentially with the number of variables.

**Keywords:** Optimization, Markov Chain, Genetic Algorithm and Fuzzy Controller.

---

# Sumário

---

<b>Sumário</b>	<b>i</b>
<b>Lista de Figuras</b>	<b>iii</b>
<b>Lista de Tabelas</b>	<b>v</b>
<b>Lista de Símbolos e Abreviaturas</b>	<b>vii</b>
<b>1 Introdução</b>	<b>1</b>
1.1 O algoritmo em análise, suas habilidades e limitações . . . . .	1
1.2 A ferramenta de modificação do AGP . . . . .	2
1.3 A ferramenta de análise e os requisitos exigidos . . . . .	2
1.4 Organização do texto . . . . .	3
<b>2 A otimização: uma visão cronológica</b>	<b>5</b>
2.1 A otimização . . . . .	5
2.2 O crescimento com o computador . . . . .	5
2.3 Os problemas de otimização e suas características . . . . .	6
2.4 Classificação pela dificuldade de resolução . . . . .	6
2.4.1 A classe $P$ . . . . .	7
2.4.2 A classe $NP$ e a inclusão $P \subseteq NP$ . . . . .	7
2.4.3 A classe $NP$ -Completo . . . . .	8
2.5 Algoritmos determinísticos versus algoritmos probabilísticos . . . . .	8
2.5.1 Alguns algoritmos determinísticos . . . . .	9
2.5.2 As metaheurísticas . . . . .	11
<b>3 Ferramentas de construção e análise</b>	<b>13</b>
3.1 A lógica nebulosa . . . . .	13
3.2 O conjunto nebuloso e suas funções de pertinência . . . . .	13
3.3 Operações com conjuntos nebulosos . . . . .	14
3.4 A inferência nebulosa . . . . .	16
3.5 Concepção de um controlador nebuloso . . . . .	20
3.5.1 O módulo da 'fuzzificação' . . . . .	20
3.5.2 O módulo de inferência . . . . .	20
3.5.3 O módulo de 'defuzzificação' . . . . .	20
3.6 Cadeia de Markov . . . . .	21



3.6.1	Definições e propriedades . . . . .	21
3.6.2	Ergodicidade da cadeia homogênea . . . . .	23
3.6.3	Ergodicidade da cadeia não-homogênea . . . . .	23
<b>4</b>	<b>A modelagem e análise de convergência</b>	<b>25</b>
4.1	O problema e o formato . . . . .	25
4.2	O AGH e o AGNH . . . . .	26
4.3	A modelagem . . . . .	28
4.4	Análise de convergência . . . . .	30
4.4.1	Quando a cadeia é do tipo AGH . . . . .	31
4.4.2	Quando a cadeia é do tipo AGNH . . . . .	31
<b>5</b>	<b>Um estudo de caso: AGH versus AGNHn</b>	<b>33</b>
5.1	O AGNHn . . . . .	33
5.2	O Controlador nebuloso usado no AGP . . . . .	34
5.2.1	O módulo de 'fuzzificação', para o AGNHn . . . . .	35
5.2.2	O módulo de inferência, para o AGNHn . . . . .	38
5.2.3	O módulo de 'defuzzificação', para o AGNHn . . . . .	38
5.3	Os problemas testes . . . . .	39
5.3.1	Os resultados obtidos . . . . .	39
<b>6</b>	<b>Conclusões e perspectivas</b>	<b>43</b>
6.1	Conclusões . . . . .	43
6.2	Perspectivas . . . . .	43
	<b>Referências bibliográficas</b>	<b>45</b>

---

# Lista de Figuras

---

3.1	Pertinências do antecedente $r(x)$ . . . . .	17
3.2	Pertinência do consequente: $sm(x)$ e $sp(x)$ . . . . .	17
3.3	Pertinência da extensão $p1(x,y)$ . . . . .	18
3.4	Pertinência da extensão $p2(x,y)$ . . . . .	19
3.5	Pertinências de $p^*(x,y)$ . . . . .	19
5.1	AGH + CONTROLADOR NEBULOSO = AGNHn . . . . .	34
5.2	Pertinência de ng . . . . .	35
5.3	Pertinência de dp . . . . .	36
5.4	Pertinência da taxa de mutação . . . . .	37
5.5	função 1 . . . . .	40
5.6	função 2 . . . . .	40
5.7	função 3 . . . . .	40
5.8	função 4 . . . . .	40
5.9	função 5 . . . . .	41
5.10	função 6 . . . . .	41



---

# Lista de Tabelas

---

5.1	A matriz de regras para a taxa de mutação . . . . .	38
-----	---	----



---

# Lista de Símbolos e Abreviaturas

---

$C$ :	O operador cruzamento ou sua matriz de transição
$G$ :	Conjunto discreto de pontos viáveis para o problema
$M$ :	O operador mutação ou sua matriz de transição
$S$ :	O operador seleção ou sua matriz de transição
$\Gamma$ :	Qualquer problema de otimização ou modelo
$\Omega$ :	Conjunto de estados
$\delta$ :	Coefficiente de ergodicidade de uma transição ou coeficiente de <i>Dobrushin</i>
$\gamma$ :	Uma instância do problema $\Gamma$
$\rho$ :	O período de um estado
$d_p$ :	Diversidade da população
$n_g$ :	Número de gerações
$p^{\rightarrow}$ :	Representa a operação <i>implica</i>
$p^*$ :	Representa a operação <i>relação</i>
$p_q$ :	Ponto de quebra usado no operador cruzamento
$t_p$ :	Tamanho da população
$tx_c$ :	Taxa de cruzamento
$tx_m$ :	Taxa de mutação
AGNHn:	Algoritmo Genético Não-Homogêneo com Controlador Nebuloso
AGH:	Algoritmo Genético Padrão modelado por cadeia de Markov
AGP:	Algoritmo Genético Padrão

---

# Capítulo 1

## Introdução

---

Neste capítulo apresenta-se o algoritmo em estudo destacando-se suas qualidades, limitações, ferramentas de melhoria e de análise de eficácia. Todas as ferramentas referidas terão seus detalhes apresentados em seção própria, conforme o caso exija.

### 1.1 O algoritmo em análise, suas habilidades e limitações

Este trabalho destina-se à análise do algoritmo genético, AGP, descrito em [Holland 1975], e o faz pelas razões, a seguir destacadas, que têm levado o mesmo a ser um dos mais usados algoritmos, com o fim a que ele se destina, a partir dos anos 90.

A primeira habilidade destacável do algoritmo é a capacidade que o mesmo tem demonstrado em obter uma solução de boa qualidade para o problema de otimização. Ressalte-se que, para esse tipo de problema, a satisfação por uma solução de boa qualidade, em lugar da solução ótima, deve-se à elevada complexidade de muitos problemas de otimização, muitas vezes imposta pelo problema, limitação essa que será abordada posteriormente em seção própria.

A segunda habilidade é a abrangência do campo de aplicação do algoritmo, decorrente do seu caráter adaptativo ao permitir seu uso em formas diferentes de representação do problema. Este caráter está associado aos conceitos de genótipo e fenótipo que o mesmo faz uso, possibilitando, com isso, sua aplicação em problemas dos mais variados matizes.

A terceira habilidade vem do fato do algoritmo genético padrão, tal como descrito em [Michalewicz 1992], aqui denominado por AGP, permitir múltiplas possibilidades de configuração para os seus operadores de seleção, de cruzamento e de mutação. Neste caso, o caráter adaptativo é reafirmado, permitindo, com isso, a busca por uma melhor configuração a tornar o algoritmo mais eficaz. Contudo esta habilidade traz enormes dificuldades para a busca desses valores dos parâmetros que confirmem o equilíbrio a se traduzir em um bom desempenho, sem contar, é claro, com a garantia de solução ótima, limitação de caráter teórico, associada à eficiência temporal.

Por isso, considera-se que o tema não é apenas estimulante mas também aberto a possibilidades de melhoria.

## 1.2 A ferramenta de modificação do AGP

Como já foi dito, o algoritmo genético pede adequadas regras na escolha de seus parâmetros e estas regras tanto podem produzir, ao longo da evolução, parâmetros constantes como podem produzir parâmetros variáveis. No caso de parâmetros fixos o que se tem feito é usar indicações da literatura, como em [Michalewicz 1992, Goldberg 1989]. Para o caso de parâmetros variáveis, quando exige a definição do valor a ser usado em cada iteração do algoritmo, algumas estratégias têm sido usadas, podendo estas serem classificadas por estratégias precisas, como em [Campos, Pereira, Carlos e de Assis 2012], ou estratégias nebulosas, como em [Cavalheiro 2003, Burdelis 2009]. A estratégia nebulosa é construída com elementos da lógica nebulosa, definida em [Zadeh 1965].

A lógica nebulosa, cujos detalhes serão apresentados em seção própria, é fundamentada em argumentos imprecisos, possibilitando a emulação de um operador, através de um conjunto de regras que representam a experiência do operador, na operação do instrumento a ser utilizado. Esse conjunto de regras quando usado juntamente com uma regra de inferência constituem o núcleo do que se chama controlador nebuloso e é com esse instrumento, descrito em seção própria, que se busca a melhoria do algoritmo do AGP, resultando no AGNHn.

É com o uso dessa estratégia, na definição do parâmetro de mutação que o algoritmo genético padrão, AGP, é modificado, dando lugar ao algoritmo genético nebuloso - AGNHn. Acredita-se que esta escolha, por impor um caráter endógeno, evita que fatores externos possam reduzir o desempenho do algoritmo, possibilitando um bom equilíbrio entre diversificação e intensificação, instrumentos basilares para um bom desempenho do algoritmo.

## 1.3 A ferramenta de análise e os requisitos exigidos

Como o algoritmo a ser modificado é o AGP e em virtude de sua modelagem ser feita, em seção própria, por cadeia de Markov homogênea, o mesmo pode ser também denominado por algoritmo genético homogêneo - AGH. Em virtude da modificação pretendida, seja pelo uso da ferramenta descrita em 1.2 ou qualquer outra, permitindo a variação da taxa de mutação, resultar em um algoritmo cuja cadeia de Markov que o modela é do tipo não-homogênea, sua denominação passa a ser algoritmo genético não-homogêneo - AGNH.

A cadeia de Markov, cujos detalhes técnicos serão apresentados em seção própria, será o instrumento de análise de convergência dos algoritmos. A escolha desta ferramenta se dá não apenas por ser ela adequada à análise de processos estocásticos que levam esse nome mas também em vista de sua abrangência, já que a mesma pode ser usada, para o fim declarado, tanto no algoritmo AGH, com seus parâmetros fixos, como em [Rudolph 1994] e em [Neto 2010], quanto no AGNH, com seus parâmetros variáveis, como em [Campos, Pereira, Carlos e de Assis 2012, Campos, Pereira e Rojas Cruz 2012].

Sendo a cadeia modeladora, do AGNH, do tipo não-homogênea, os requisitos de convergência para a mesma vão além dos exigidos para o caso da cadeia do tipo homogênea, o caso do AGH.



Essas condições precisam ser garantidas afim de garantir a convergência dos algoritmos, em especial para o caso não-homogêneo do tipo nebuloso, AGNHn, por ser este o objetivo da tese.

## 1.4 Organização do texto

No capítulo 2 procura-se contextualizar o tema dando elementos para a classificação de problemas e de algoritmos, com o fim de mostrar a importância e a necessidade da contribuição oferecida. No capítulo 3 apresenta-se as ferramentas usadas, tanto na construção da proposta quanto na análise. No capítulo 4 apresenta-se os algoritmos, padrão ou homogêneo - AGP/AGH, o não-homogêneo - AGNH, suas respectivas modelagens e, em seguida, faz-se as análises de convergências dos mesmos. No capítulo 5, apresenta-se um estudo de caso para comparar os desempenhos,  $AGH \times AGNHn$ , dos algoritmos analisados. Por fim, no capítulo 6 apresenta-se conclusões e perspectivas.



---

## Capítulo 2

# A otimização: uma visão cronológica

---

Neste capítulo pretende-se dar uma contribuição através de uma resumida visão geral da otimização, desde os primórdios até o seu estado atual, apresentando cronologicamente o seu desenvolvimento através de seus resultados mais expressivos.

### 2.1 A otimização

A otimização é uma área da matemática aplicada que cuida tanto da modelagem de problemas quanto da geração de algoritmos. Esses problemas constituem o desejo de otimização de sistemas para os quais se buscam modelos, enquanto os algoritmos são os instrumentos com os quais se pretende dar respostas a esses modelos, buscando em cada universo uma solução ótima para o mesmo. A otimalidade é obtida por algum critério chamado objetivo, definido e dependente das variáveis do modelo, que assumem valores no universo do mesmo.

### 2.2 O crescimento com o computador

A despeito de já existir o conhecimento de técnicas, desde o século 18, capazes de oferecer algoritmos que levem à solução do problema de otimização, estes algoritmos consideram apenas características associadas à existência e unicidade de solução sem levar em conta a dificuldade de resolução do problema associada à cardinalidade do conjunto de soluções possíveis que, a depender desta, pode levar a não aceitação do algoritmo, em vista do tempo que o mesmo gasta para oferecer a solução ótima. Ao tempo, sem o computador, só era possível realizar a busca de uma solução ótima à mão e, em consequência, somente seria possível para problemas com um baixo grau de dificuldade. Mas esses casos nunca despertaram interesse pois os problemas reais têm, quase sempre, um grande número de soluções possíveis a ponto de, como no caso, exponencial, demandar cálculos que só se tornaram realizáveis com o surgimento do computador. Por isso, o crescimento da otimização se deu casado com o surgimento do computador, e o aumento da capacidade de cálculo do computador fez reduzir em muito as limitações no desenvolvimento de novos algoritmos, apesar de persistirem outras dificuldades inerentes ao problema e invariantes com esse movimento, como nos chamados problemas não-polinomiais.

## 2.3 Os problemas de otimização e suas características

Esses problemas têm origem nas mais variadas áreas do conhecimento humano, como em tarefas de engenharia elétrica, de engenharia de produção, de computação, etc., como relatam [Rockafellar 1972, Salkin 1975, Gill et al. 1981, Chvátal 1983, Nemhauser e Wolsey 1988]. Em vista da presença ou não de características nas funções envolvidas do modelo, tais como linearidade, convexidade, diferenciabilidade, ou em relação ao domínio do modelo, podendo este ser irrestrito ou restrito e ainda contínuo ou discreto. Cada uma dessas características emprestam o nome aos problemas, entretanto a classificação seguinte é feita levando em conta a dificuldade de resolução do problema.

## 2.4 Classificação pela dificuldade de resolução

Nesta seção apresenta-se uma pequena amostra à classificação dos problemas de otimização, pela dificuldade na sua resolução e, com isso, identifica-se classes que dão o real sentido da busca por um algoritmo que tenha chance, ainda que incerto, de encontrar uma solução ótima ou próxima dela, o que inevitavelmente tem levado ao uso das metaheurísticas, dentre estas o algoritmo genético.

### O problema, a instância e seu tamanho

A modelagem matemática oferece o modelo ou, como se costuma chamá-lo, o problema de otimização. Esse modelo ou problema,  $\Gamma$ , é caracterizado por uma estrutura de dados, um conjunto de regras, que estabelece como esses dados se relacionam e como devem ser processados em busca da solução através de um critério ou objetivo que define a qualidade da solução encontrada. Cada atribuição à estrutura de  $\Gamma$ , por um particular conjunto de dados, é chamada de instância,  $\gamma$ , do problema  $\Gamma$ , ou, por facilidade, pode-se usar a representação  $\gamma \in \Gamma$ . Portanto, cada problema de otimização,  $\Gamma$ , tem a si associadas suas infinitas instâncias,  $\gamma$ 's, e cada uma dessas com tamanho de representação finito. Quando admitida a representação binária o tamanho da instância será o inteiro da forma  $\kappa(n) = \sum_{i=0}^n \delta_i 2^i$ , sendo  $\delta_i = 0$  ou  $1$ , para algum  $n$ .

### A classificação de problemas pela dificuldade de resolução

Para este fim, alguns critérios têm sido usados e dentre esses tem se destacado o critério do pior caso, a despeito do critério do caso médio, aquele critério baseado em um padrão médio de entradas de dados correspondentes à instância  $\gamma$ . A principal dificuldade encontrada está na caracterização desse padrão médio. O critério do pior caso, como o próprio nome declara, usa a estratégia de definir a dificuldade de resolução de  $\Gamma$  pela dificuldade de resolução da instância,  $\gamma \in \Gamma$ , mais desfavorável possível. Em vista disso, considere as definições seguintes.

### Algoritmo polinomial

Considere o polinômio de grau  $m(\gamma)$ ,  $p_{m(\gamma)}(x) = \sum_{i=0}^m \delta_i x^i$ , sendo  $\delta_i = 0$  ou  $1$ , e seja o inteiro,  $\kappa(m(\gamma)) = p_{m(\gamma)}(2)$ , descrito como antes, como sendo o tamanho da instância  $\gamma$ .

Um algoritmo  $A$  destinado a resolver um problema  $\Gamma$  é dito ter ordem polinomial, ou simplesmente se diz que  $A$  é  $O(\kappa(m(\gamma))^n)$ , quando  $A$  resolve qualquer instância  $\gamma \in \Gamma$  de tamanho  $\kappa(m(\gamma))$  no tempo  $p_n(\kappa(m(\gamma)))$ , quando é satisfeita a condição  $\kappa(m(\gamma)) \leq \alpha p_n(\kappa(m(\gamma)))$ , para todo  $n \geq m$ , algum  $\alpha > 0$  e todo  $\gamma \in \Gamma$ . Neste caso a função  $\kappa(m(\gamma))^n$  é chamada de complexidade do algoritmo  $A$  e dita polinomial. Em consequência, o problema,  $\Gamma$ , é também assim chamado.

#### 2.4.1 A classe $P$

Esta classe é constituída dos problemas  $\Gamma$  para os quais se conhece um algoritmo polinomial que resolve qualquer instância  $\gamma \in \Gamma$ . Dentre muitos estão os problemas: resolução de sistema linear, o problema da árvore geradora mínima, o problema de programação linear, etc.

Com a constatação da existência de problemas não pertencentes à classe  $P$ , ainda que possam vir a pertencerem, se faz necessário a definição de uma classe mais abrangente que a classe  $P$ , como a classe  $NP$  a seguir definida.

### O problema de decisão

Para se chegar à definição de  $NP$  define-se inicialmente o problema de decisão, como sendo aquele cuja solução é apenas o SIM ou o NÃO. Portanto, a cada problema  $\Gamma$  define-se um problema de decisão associado a  $\Gamma$  como sendo o problema dado pela inclusão  $\gamma \in \Gamma$ . Observe que em vista da infinidade de problemas de decisão,  $\gamma \in \Gamma$ , a resposta, em tempo polinomial, a esse problema seria dada com um hipotético conjunto de infinitas cópias de um algoritmo polinomial, rodando em paralelo. Este procedimento é o que passou a ser chamado algoritmo não-determinístico polinomial. Maiores detalhes sobre o tema podem ser encontrados em [Garey e Johnson 1979, Papadimitriou 1982, Nemhauser e Wolsey 1988, Ribeiro 1998]. Em vista disso, temos a definição seguinte.

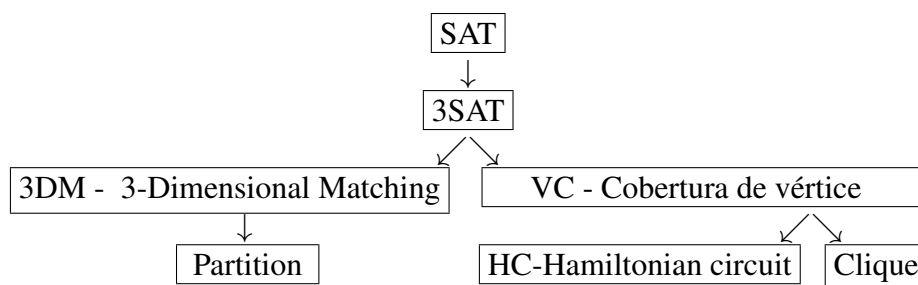
#### 2.4.2 A classe $NP$ e a inclusão $P \subseteq NP$

A classe  $NP$  é constituída dos problemas  $\Gamma$  para os quais qualquer problema de decisão associado a  $\Gamma$  seja resolvido por um algoritmo não-determinístico polinomial, ou seja, basta que o problema de decisão associado a  $\Gamma$  seja polinomial. Observe que esta exigência não é dirigida a  $\Gamma$  mas ao problema de decisão associado a  $\Gamma$ . Assim definida, fica claro que a relação entre  $P$  e  $NP$  fica dada por  $P \subseteq NP$ . Portanto pode-se dizer que  $NP$  é constituída dos problemas,  $\Gamma$ , para os quais não se conhece um algoritmo polinomial que o resolva.

Muitos problemas, como fizeram em [Cobham 1964, Edmonds 1965a, Edmonds 1965b, Edmonds e Johnson 1970, Edmonds e Karp 1972, Edmonds e Johnson 1973], foram introduzidos nessa classe, os quais deram motivos para que se chegasse ao resultado seguinte.

### 2.4.3 A classe *NP-Completo*

A classe dos *NP – Completo* pertence à classe *NP* pois seus respectivos problemas de decisão são resolvidos em tempo polinomial. No desejo de descrever o universo dos problemas, a partir da intratabilidade dos mesmos, torna-se inevitável partir do trabalho de Alan Turing com os seus problemas de mais alto grau de intratabilidade denominados de problemas indecidíveis, porque não há qualquer algoritmo que os resolva, como o seu legítimo representante, o chamado problema da parada. Entretanto, ao se restringir ao universo da otimização, tratando-se de problemas decidíveis, em 1971 Stephen Cook em seu destacado trabalho, [Cook 1971a], definiu a classe *NP – Completo* e introduziu o seu primeiro representante, o problema da satisfabilidade-SAT. Esta classe é, portanto, constituída dos problemas, de *NP*, que são tão intratáveis quanto o problema SAT. Posteriormente, outros problemas foram sendo introduzidos, como fez R. M. Karp em [Karp 1972] e outros, fazendo a classe crescer tanto que qualquer tentativa de descrição da mesma seria muito enfadonho. A seguir apresenta-se a árvore de redução, cujos detalhes podem ser encontradas em [Garey e Johnson 1979]



Todas essas introduções são feitas com a técnica de redução polinomial pela qual um primeiro problema é transformado em um segundo, por meio de uma transformação em tempo polinomial, determinando, com isso, que a intratabilidade do primeiro é dada pela intratabilidade do segundo, ou seja, se um problema da classe *NP – Completos* for introduzido na classe *P*, serão todos os outros.

A despeito da classificação dada acima, em função da dificuldade de resolução do problema, uma outra classificação dada, a seguir, é dirigida a classificar os algoritmos, respeitando o emparelhamento do algoritmo ao tipo de problema ao qual o algoritmo se destina.

## 2.5 Algoritmos determinísticos versus algoritmos probabilísticos

Dentre outras características que podem ser usadas para classificar os algoritmos de otimização, as duas classes seguintes merecem destaque. A classe dos algoritmos determinísticos, aqueles algoritmos que geram o mesmo traço, inclusive sua solução, em repetidas execuções, quando aplicados ao mesmo problema e a classe dos algoritmos probabilísticos, aqueles cujo traço de cada execução, inclusive sua solução, é aleatório.

### 2.5.1 Alguns algoritmos determinísticos

Muitos desses algoritmos foram gerados no período que vai do fim dos anos 60 ou início dos anos 70. Além de terem caráter predominantemente determinístico, quase sempre são monotônicos, quando a passagem da solução presente para a próxima é motivada pela melhoria no critério ou função objetivo. Ainda nessa classe, não se pode deixar de notar a explícita separação entre os métodos dirigidos a problemas irrestritos e a problemas com restrições, podendo ainda, estas restrições, serem de igualdade ou de desigualdades.

As características determinísticas desses algoritmos, sejam para problema irrestrito seja para problema com restrições, devem-se, em parte, ao uso de condições de caracterização de ótimos locais, a não ser em condições especiais como, por exemplo, quando se tem a convexidade ou informação sobre curvatura.

Quando o problema é irrestrito, as condições de otimalidade, com hipótese de diferenciabilidade, há muito foram estabelecidas pela matemática, com o conceito de ponto crítico, como em [Fletcher 1980]. Quando não se tem diferenciabilidade, em um conjunto discreto de pontos, presente nos problemas do tipo max-min, essas condições são dadas com o conceito de quase -diferenciabilidade, sendo um ótimo local caracterizado pela inclusão do subgradiente nulo no conjunto convexo chamado de subdiferencial. Os conceitos de subgradiente, obtido pela generalização do conceito de gradiente, e de subdiferencial podem ser vistos em [Rockafellar 1972].

Quando o problema é restrito, as restrições ainda podem ser de igualdade ou de desigualdade. Para o caso com restrições de igualdade, a caracterização de ótimo local, dadas em curso básico de cálculo, é definida pelos chamados multiplicadores de Lagrange e a relação desses com o gradiente. Para o problema com restrições de desigualdade as condições de caracterização foram obtidas, também com a hipótese de diferenciabilidade, nas conhecidas condições de Karush-Kuhn-Tucker, apresentadas em [Kuhn e Tucker 1951], com abordagem também em [Avriel 1976, Bazaraa e Shetty 1979, Fletcher 1981]. Estas condições são uma generalização das condições de Lagrange, já referidas e obtidas para restrições de igualdade.

Para todos esses problemas, sem contar, é claro, aqueles cuja classificação exige a diferenciabilidade, ainda há que se respeitar a natureza bivalente do domínio do problema, podendo ser este contínuo ou discreto e a repercussão dessa bivalência para os algoritmos, dirigidos a problemas contínuos ou a problemas discretos.

#### Quando o problema é contínuo

Talvez por oferecerem, na sua resolução, uma menor dificuldade que os problemas restritos, os problemas irrestritos, não lineares, foram os primeiros a terem algoritmos destinados à sua resolução, haja vista o método da máxima descida, 'steepest descent method', atribuído ao matemático Cauchy, que faz uso do gradiente da função para definir a direção de busca, método esse que só se tornou efetivo quando auxiliado por mecanismos de busca linear do tipo encontrado em [Goldstein 1965, Armijo 1966]. Ainda no caso de problema irrestrito se registra a existência de métodos que não usam a derivada, como em [Nelder e Mead 1965].

Com a introdução de restrições o problema tem o grau de dificuldade aumentado mas,

ao mesmo tempo, há de se destacar um problema dessa classe chamado de problema de programação linear, pois só assim garante-se solução para ele. O destaque deve-se ao método simplex, publicado por G. B. Dantzig no início dos anos 50, com detalhes em [Dantzig 1963, Chvátal 1983]. Durante muitos anos, a exemplo do desempenho demonstrado na prática, o simplex foi tido como um algoritmo polinomial. Entretanto, com a publicação do trabalho [Klee e Minty 1972] apresentando um exemplo, que pode ser visto em [Chvátal 1983], no qual o método não apresenta comportamento polinomial, chegando a realizar  $2^n - 1$  iterações onde  $n$  é o número de soluções básicas viáveis. Por isso e desde então, o problema de programação linear não podia ser incluído na classe dos problemas polinomiais. Contudo, no final da década de 70, foi definitivamente introduzido na classe dos problemas polinomiais com um trabalho de grande repercussão, devido a [Khachiyan 1979]. Posteriormente, com a versão oferecida por [Karmarkar 1984], baseado no trabalho de [Dikin 1967], deu lugar aos métodos de pontos interiores, cujos detalhes podem ser vistos em [Gonzaga 1989]. Esses métodos tornaram-se bastante promissores e o simplex finalmente passou a ter um concorrente no critério de eficiência. Ainda nos anos 80 esses métodos foram estendidos à classe dos problemas convexos, como em [Ye 1987] e [Gonzaga e Carlos 1990].

Talvez, pelo fato da introdução de restrições representar aumento na dificuldade de resolução do problema, no caso de restrições não-lineares, as primeiras tentativas em resolvê-lo tenham sido no sentido de eliminar o efeito das mesmas, tratando-as de forma implícita, com técnicas do tipo barreira e do tipo penalidade, cujos detalhes podem ser vistos em [Gill et al. 1981]. Ainda no caso de restrições não-lineares, uma outra estratégia bastante usada, pois deu resultado no caso do problema de programação linear, foi a estratégia de conjunto ativo, com o fim de definir uma direção de busca. Uma vez conhecido o conjunto ativo, define-se a direção de busca pela projeção do gradiente da função objetivo sobre o espaço tangente das restrições ativas. Essa classe de métodos chamados de métodos do gradiente projetado, apresentada por [Rosen 1960], certamente teve sua inspiração no algoritmo simplex, já conhecido à época. Entretanto, esses métodos oferecem uma dificuldade devido ao fato da projeção do gradiente, sobre o espaço tangente, mesmo sendo uma direção de busca que garanta a melhoria do critério, pode gerar inviabilidade. Por isso, os métodos do tipo gradiente projetado, quando usados em restrições não-lineares, precisam usar algum critério, chamado função mérito, que reflita o grau de curvatura das restrições, a fim de corrigir, tanto quanto possível, a inviabilidade gerada.

Por último, pode-se afirmar que o determinismo por uma solução que garanta uma melhoria no critério, quando comparado com o valor da solução presente, é o traço comum de todos esses métodos e esse mesmo caráter monotônico continua presente nos métodos dirigidos ao problema com quase -diferenciabilidade, onde a direção de busca, neste caso, é um sub -gradiente, pertencente ao sub -diferencial, ver detalhes em [Rockafellar 1972]. Esses métodos são os chamados métodos do tipo feixe 'bundle method', cujos detalhes podem ser vistos nos trabalhos de [Lemarechal 1980].

### **Quando o problema é discreto**

Essa é uma classe de problemas que tem ocupado grande parte da comunidade de pesquisadores. Este fato se dá porque muitos problemas, presentes nas mais variadas



áreas da atividade humana, podem ser modelados como tal. Dentre esses problemas, um representante que merece destaque é o conhecido problema do caixeiro viajante, não apenas pelo elevado grau de dificuldade na sua resolução, dado pelo elevado número de soluções viáveis, mas também por modelar um número elevado de problemas reais.

Os problemas discretos, podendo ser modelados como um problema de programação inteira, quando suas variáveis assumem apenas valores inteiros, podem também ser modelados, de forma equivalente, como problemas em que suas variáveis assumem apenas os valores 0 ou 1, por isso, de domínio representado por  $[0, 1]^n$ , para  $n$  variáveis. Para se chegar à solução desses problemas, dado o elevado número de soluções possíveis, os algoritmos usam a idéia de descarte de soluções, definido por algum critério de descarte de soluções, que não mais mereçam ser cogitadas em vista desse critério. Um desses métodos é o método de corte que se destina à resolução do problema linear inteiro e tem por base a relaxação da condição de inteiro. Com isso, o problema relaxado pode ser resolvido, por exemplo, pelo simplex, para gerar uma cota, conforme o caso, superior (ou inferior) para o problema de minimização (ou maximização). Esses métodos podem ser vistos em [Salkin 1975]. Um outro, talvez o mais importante método que usa o critério de descarte de solução, por ter em alguns casos um desempenho satisfatório, é o chamado método 'branch and bound'. cujos detalhes podem ser encontrados em [Nemhauser e Wolsey 1988]. Esses algoritmos definem um ramo ('branch') de problemas a ser explorado e faz descarte desse ramo com base no uso do limitante ('bound').

### 2.5.2 As metaheurísticas

Muitos destes algoritmos, por terem componentes não determinísticas associadas com componentes determinísticas ou heurísticas, passaram a ser chamados de metaheurísticas. Todas elas têm um traço comum ao perderem o caráter monotônico, quando a evolução já não é mais motivada pela melhoria no valor do critério mas evoluem com base em alguma medida de probabilidade de sucesso. Os primeiros algoritmos originariamente de otimização a usar essa estratégia, deixando de fora o método Monte Carlo, foram o algoritmo genético devido a [Holland 1975] e 'simulated annealing', devido a [Kirkpatrick et al. 1983]. Depois deste surgiram outros, dentre os quais o método de busca tabu, devido a [Glover 1986, Hansen 1986], cujos detalhes podem ser vistos em [Glover e Laguna 1993], GRASP em [Resende e Feo 1995], o colonia de formigas de [Dorigo et al. 1996], VNS em [Mladenovic 1995] dentre outros.

Esta estratégia, a despeito de permitir que se possa ir para uma solução de valor desfavorável, tem a finalidade de fugir de ótimos locais permitindo que se visite outras regiões promissoras. Com isto, constata-se que os algoritmos perdem o ímpeto por uma solução local, para evitar que venham a ficar presos em ótimos locais de baixa qualidade e, ao mesmo tempo, sem esquecer que todo ótimo global é também local, a busca por este precisa ser enfrentada. Por isso, o controle entre a necessidade pela busca de um ótimo local e a intensidade com que se realiza essa busca é o que se chama, respectivamente, de intensificação e de diversificação. Portanto, um bom algoritmo é aquele que encontra uma solução tão boa quanto possível, exigindo com isso um bom equilíbrio entre diversificação e intensificação.

Sendo as metaheurísticas, a menos da heurística usada, dirigidas a problemas genéricos de otimização, sua busca por regiões promissoras é feita sem o uso de informações particulares do problema. Por isso, existe a probabilidade do algoritmo escolher somente regiões que quando exploradas, na sua busca local, demonstrem não serem tão promissoras. Assim, quando nenhuma dessas regiões contém o ótimo global, tem-se como consequência inevitável a geração de uma solução diferente do ótimo global. Portanto, tornar o algoritmo mais dedicado, tanto quanto possível, com a introdução de características particulares do problema, na busca por tais regiões promissoras, tem sido o caminho trilhado pelos métodos de otimização global. Alguns desses algoritmos, como pode ser visto em [Hendrix e Tóth 2010], realiza esta tarefa com a hipótese de ser, a função objetivo, do tipo Lipschitz, ou seja,

$$|f(x) - f(y)| \leq L \|x - y\|, \forall x, y \in D$$

onde  $D$  é o domínio e  $L$  é a chamada a constante de Lipschitz. Esta condição, ao evitar que a função não seja tão íngreme, possibilita a descoberta de regiões certamente não promissoras, evitando que as mesmas sejam exploradas.

---

## Capítulo 3

# Ferramentas de construção e análise

---

Neste capítulo apresenta-se os elementos de lógica nebulosa e de controlador nebuloso para, com este, definir o algoritmo genético nebuloso, resultante da junção do AGP/AGH, tal como descrito da seção 4.2, com o controlador nebuloso. Por fim, apresenta-se a cadeia de Markov que será o instrumento de análise do algoritmo proposto.

### 3.1 A lógica nebulosa

A lógica nebulosa, introduzida por Zadeh, em [Zadeh 1965], está para a teoria de conjunto nebuloso da mesma forma que a lógica clássica está para a teoria clássica de conjuntos. Na teoria clássica, um conjunto  $X$  é uma coleção de elementos  $x$  de um dado universo  $U$ , onde se define a proposição clássica,  $x \in X$ , para a qual o valor lógico é *SIM* ou *NÃO* aos quais se associa, respectivamente, os valores 1 ou 0, também chamados de grau de pertinência do elemento  $x$  ao conjunto  $X$ . Portanto, a extensão sugerida por Zadeh permite que o grau de pertinência, agora diferente de 0 e 1, assumam valores no intervalo  $[0, 1]$  e, em consequência, permite-se que se defina para  $x$ , agora nebuloso, a proposição nebulosa  $x \in X$ , cuja avaliação agora é multi-valorada ou imprecisa. Para melhor esclarecer sobre a necessidade de tal extensão, sabe-se que a lógica clássica gera situações indesejáveis quando, por exemplo, se padroniza que uma pessoa é considerada baixa quando sua altura vai até  $1.50m$  e, portanto, quem tem altura  $1.51m$ , nesse padrão, já é considerado alto, mesmo julgamento dado a quem tem altura  $1.70m$ . A lógica nebulosa trata esse tipo de inadequação, respeitando a ordem natural de altura, através de grau valorado de pertinência, permitindo que a pessoa que tenha a altura  $1.51m$  seja também considerada baixa mesmo que tenha grau de pertinência menor do que a que tem altura  $1.50m$ , mas também permite que essa mesma pessoa, de altura  $1.51m$ , possa também ser considerada alta, com grau de pertinência mais alto do que aquela pessoa que tem altura  $1.50m$ .

### 3.2 O conjunto nebuloso e suas funções de pertinência

Os resultados pretendidos na seção 3.1 serão alcançados com a caracterização dos conjuntos nebulosos  $U_i \subseteq U$ , sendo  $\cup U_i = U$  e  $\cap U_i \neq \emptyset$ , através das respectivas funções

de pertinência  $\mu^i : U \rightarrow [0, 1]$  definida pela função

$$\mu^i(x) = \begin{cases} \mu^j(x) & \text{quando } \mu^j(x) \in (0, 1] \\ 0 & \text{caso contrário} \end{cases}$$

sendo  $U_i$  também chamado de conjunto de valor linguístico  $i$ , e as funções  $\mu^j : U_i \rightarrow [0, 1]$ , ambos definidos convenientemente, tanto na sua forma quanto na quantidade. Observe que a lógica clássica torna-se um caso particular da lógica nebulosa quando nesta existe apenas um valor linguístico, ou seja,  $\cup U_i = U$ , e  $\mu^i : U_i \rightarrow \{0, 1\}$ .

### 3.3 Operações com conjuntos nebulosos

Nesta seção apresenta-se as principais operações entre conjuntos nebulosos. Estas operações, a exemplo do que se faz na lógica clássica com a função característica, são feitas através de suas funções de pertinência. Portanto, nas definições seguintes considera-se que os conjuntos  $A$  e  $B$  serão caracterizados pelas respectivas funções de pertinência  $\mu_A$  e  $\mu_B$ . Sendo assim, seguem as definições dos operadores básicos de conjuntos e algumas de suas propriedades, cujos detalhes podem ser encontrados em [Zadeh 1965], onde foram lançadas.

#### A igualdade - $A = B$

Ocorre a igualdade entre os conjuntos  $A$  e  $B$  quando as funções de pertinência têm o mesmo domínio e  $\mu_A = \mu_B$ , em todo o domínio.

#### A inclusão - $A \subset (\subseteq) B$

A inclusão,  $A \subset (\subseteq) B$ , se dá quando as funções de pertinência satisfazem a  $\mu_A < (\leq) \mu_B$ . Neste caso o domínio de  $\mu_A$  está contido, podendo ser o próprio ou não, no domínio de  $\mu_B$ .

#### União de conjunto - $\cup$

Define-se  $A \cup B$  pela função de pertinência

$$\mu_{A \cup B} = \max\{\mu_A, \mu_B\}$$

sendo esta operação associativa,  $A \cup (B \cup C) = (A \cup B) \cup C$ .

#### Interseção de conjuntos - $\cap$

Define-se  $A \cap B$  pela função de pertinência

$$\mu_{A \cap B} = \min\{\mu_A, \mu_B\}$$

que também é associativa,  $A \cap (B \cap C) = (A \cap B) \cap C$ .

Com os operadores acima também vale a distributividade,

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

### O complementar de $A$ - $A^c$

Define-se  $A^c$  pela função de pertinência

$$\mu_{A^c} = 1 - \mu_A$$

Agora, com os operadores definidos acima, pode-se verificar a satisfação das leis de De Morgan,

$$(A \cup B)^c = A^c \cap B^c$$

$$(A \cap B)^c = A^c \cup B^c$$

### Diferença simétrica $A - B$

Este conjunto é caracterizado pela pertinência  $\mu_{(A-B)} = |\mu_A - \mu_B|$

### Generalizando os operadores

A operação  $(\cap)$ , realizada pelo operador descrito acima, pode também ser realizada pelos operadores:

1. *Produto* -  $\mu_{A \cap B} = \mu_A \cdot \mu_B$ ;
2. Por Lukasiewicz -  $\mu_{A \cap B} = \max\{0, \mu_A + \mu_B - 1\}$

dentre outros.

Com o operador,  $T_n$ , chamado t-norma,

$$T_n : \mu_A(U) \times \mu_B(U) \rightarrow [0, 1]$$

por  $T_n(\mu_A, \mu_B) = \mu_{A \cap B}$ , satisfazendo às seguintes condições:

1. Comutativa:  $T_n(a, b) = T_n(b, a)$
2. Associativa:  $T_n(a, T_n(b, c)) = T_n(T_n(a, b), c)$
3. Elemento neutro:  $T_n(1, a) = T_n(a, 1) = a$
4. Monotonicidade:  $T_n(a, b) \leq T_n(c, d)$  quando  $a \leq c$  e  $b \leq d$

faz cada um dos outros um caso particular deste.

De forma similar, quando for o caso da operação  $(\cup)$ , como a mesma também pode ser realizada pelos operadores:

1. Por soma-produto -  $\mu_{A \cup B} = \mu_A + \mu_B - \mu_A \cdot \mu_B$
2. Por dual de Lukasiewicz -  $\mu_{A \cup B} = \min\{1, \mu_A + \mu_B\}$

dentre outros.

Com o operador,  $T_c$ , chamado t-conorma,

$$T_c : \mu_A(U) \times \mu_B(U) \rightarrow [0, 1]$$

por  $T_c(\mu_A, \mu_B) = \mu_{A \cup B}$ , que satisfazendo às seguintes condições:

1. Comutativa:  $T_c(a, b) = T_c(b, a)$
2. Associativa:  $T_c(a, T_c(b, c)) = T_c(T_c(a, b), c)$
3. Elemento neutro:  $T_c(0, a) = T_c(a, 0) = a$
4. Monotonicidade:  $T_c(a, b) \leq T_c(c, d)$  quando  $a \leq c$  e  $b \leq d$

faz cada um dos outros um caso particular deste.

### 3.4 A inferência nebulosa

A chamada inferência nebulosa usa uma regra de inferência e uma base de regras, também chamada de matriz de regras. A regra de inferência nebulosa tem em seu antecedente um conjunto de regras da sua base. Estas regras são definidas pela sensibilização das mesmas a partir de algum valor preciso, as quais passam a ser chamadas de regras disparadas e com as quais se infere o resultado.

Na lógica clássica, para as proposições  $p$  e  $q$ , uma regra de inferência bastante usada é a chamada regra modus ponens, definida por

$$[p \wedge (p \rightarrow q)] \rightarrow q$$

onde  $(\wedge)$  é o operador conjunção e  $(\rightarrow)$  é o operador condicional. Para a lógica nebulosa, define-se, a partir desta, a regra modus ponens generalizada, dada por

$$[p' \wedge (p \rightarrow q)] \rightarrow q'$$

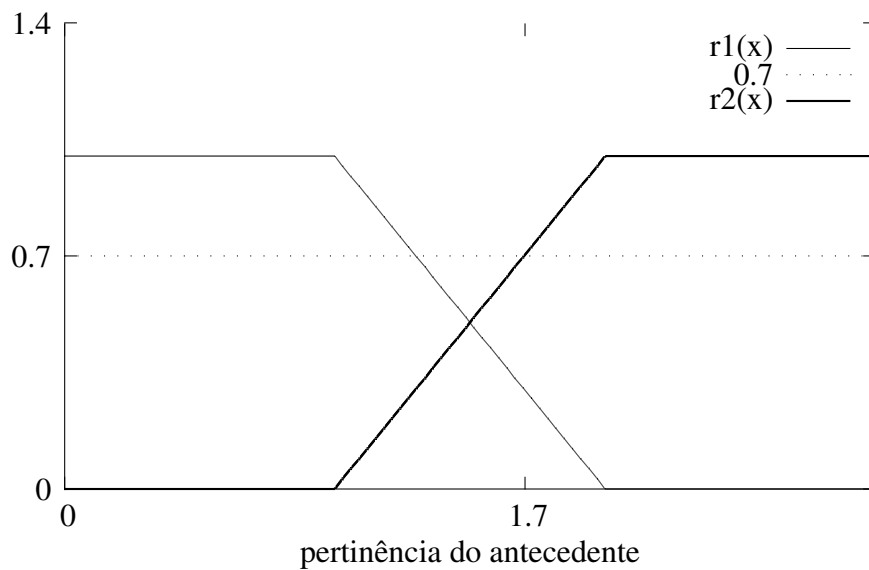
onde  $p$ ,  $p'$ ,  $q$  e  $q'$  são proposições nebulosas.

Nesta regra, a proposição  $p \rightarrow q$  é uma regra da matriz de regras enquanto que as proposições  $p'$  e  $q'$  serão construídas a seguir, a fim de garantir o valor verdadeiro da regra de inferência. Para isso, como sua realização depende da realização do operador de implicação e do operador relação, a descrição dos mesmos vem a seguir.

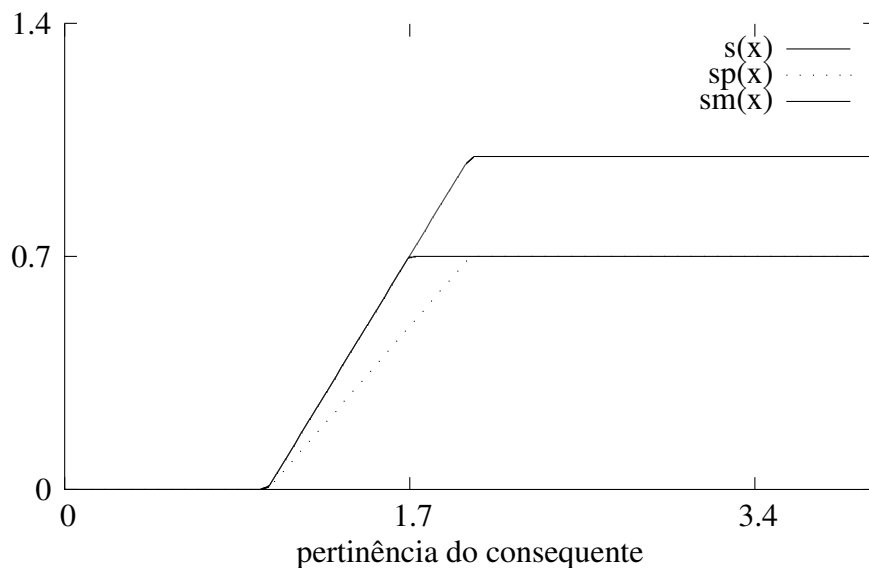
Para a definição do operador implicação, considere a sentença  $r \rightarrow s$ , onde  $r = r1 \vee r2$  e suas pertinências são dadas a seguir, sendo sensibilizadas com uma entrada. Agora, após a realização da operação  $\vee$ , aplica-se a regra do mínimo, como sugeriu [Mandani e Assilan 1999] ou a regra do produto, para se chegar ao consequente  $s$ . Estas operações ficam explicitadas nos gráficos seguintes:

#### Gráfico do antecedente versus gráfico do consequente

O gráfico do antecedente é obtido com a pertinência de  $r$  e a entrada precisa 1.7, ficando assim

Figura 3.1: Pertinências do antecedente  $r(x)$ 

Para o consequente  $s$ , a fim de garantir o valor verdade da implicação, a regra do min define a pertinência  $sm$  enquanto a regra do produto define  $sp$ , dadas por

Figura 3.2: Pertinência do consequente:  $sm(x)$  e  $sp(x)$ 

Portanto, sendo verdadeiro o antecedente  $r$  será também verdadeiro o consequente  $s$ , pois o corte, seja pelo mínimo ou pelo produto, apenas reduzem a pertinência deste, não havendo alteração no seu domínio.

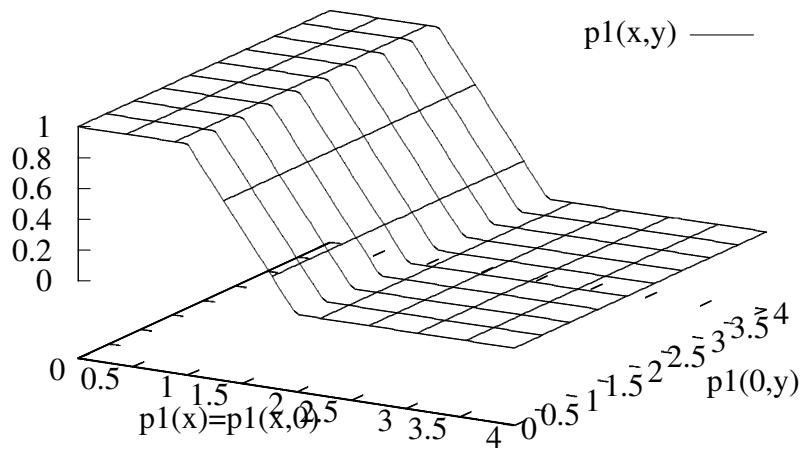
Quanto ao operador *relação*, aqui representada por  $*$ , o mesmo é definido para funções de pertinências com domínios diferentes, a fim de torná-los iguais. Para tal, considere as proposições  $p_1(x)$  e  $p_2(y)$  definidas nos universos  $X$  e  $Y$  e, por facilidade de descrição, admita que  $p \equiv p_1(x) \wedge p_2(y)$ . Sendo assim, a partir das funções de pertinência de  $p_1(x)$  e  $p_2(y)$  define-se as respectivas funções de pertinência de  $p_1(x,y)$  e  $p_2(x,y)$ , resultantes das respectivas extensões das mesmas e com estas, agora definidas no mesmo universo  $X \times Y$ , define-se a função

$$p^*(x,y) = p_1 * p_2 = * \{p_1(x,y), p_2(x,y)\}$$

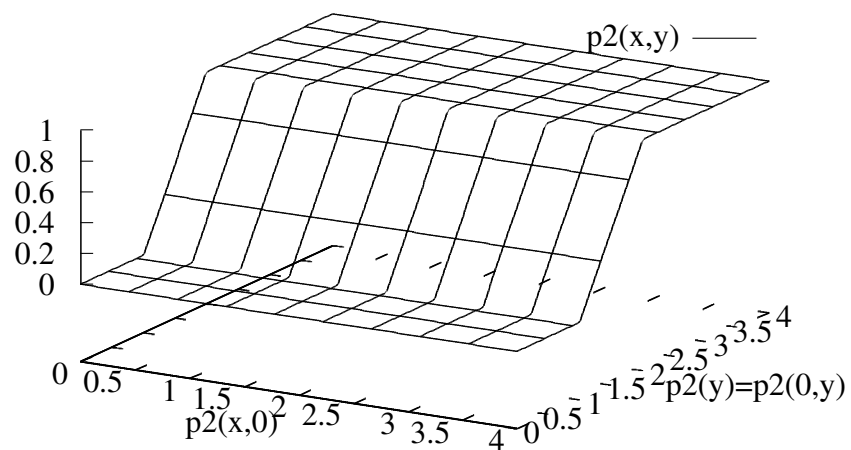
como sugerido em [Mandani e Assilan 1999, Takagi e Sugeno 1985], podendo  $*$  ser definida pelo min, pelo produto  $p_1(x,y)p_2(x,y)$  dentre outras. Agora, define-se a associação  $p' \equiv p_1 * p_2$  para se chegar ao antecedente da regra,  $[p' \wedge (p \rightarrow q)]$ . Esta operação fica explicitada na sequência dos seguinte gráficos

#### A extensão de $p_1(x)$

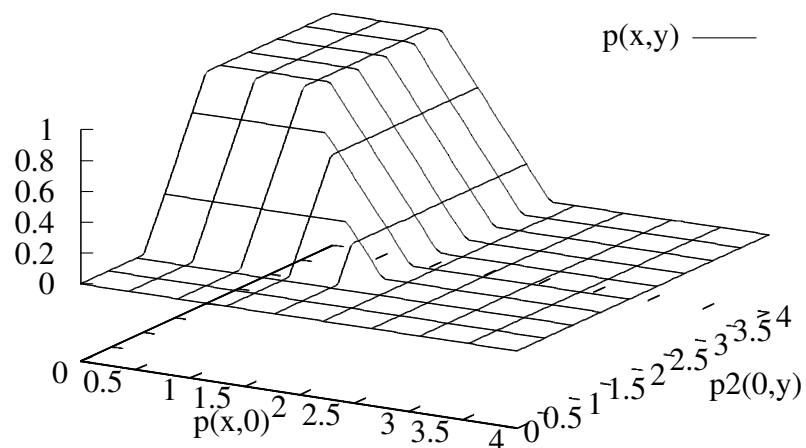
Figura 3.3: Pertinência da extensão  $p_1(x,y)$





**A extensão para  $p_2(x)$** Figura 3.4: Pertinência da extensão  $p_2(x,y)$ **A pertinência de  $p^*(x,y)$** 

Esta função é obtida com a sobreposição das duas superfícies anteriores e ficando com a que fica por baixo, pelo uso do operador min.

Figura 3.5: Pertinências de  $p^*(x,y)$ 

### 3.5 Concepção de um controlador nebuloso

Um controlador nebuloso é constituído de três módulos, o da 'fuzzificação', o da inferência e o da 'defuzzificação'. O primeiro módulo é responsável pela transformação da informação precisa em informação nebulosa enquanto que o módulo da inferência é responsável pela geração de novas informações nebulosas obtidas a partir das informações nebulosas do primeiro módulo e, por último, o terceiro módulo transforma as informações nebulosas do segundo módulo em informações precisas, do universo no qual teve início o processo.

#### 3.5.1 O módulo da 'fuzzificação'

Para definir o primeiro módulo identificam-se no problema, para o qual o controlador é dirigido, os domínios  $X$ 's e as variáveis  $x_i$ , que se tornarão nebulosas pela escolha da partição nebulosa de cada domínio  $X$ . Com essa escolha definem-se entre estas variáveis quais serão de entrada,  $x_{\text{entrada}}$  e quais serão de saída,  $x_{\text{saída}}$ , sendo estas últimas dependentes das primeiras e seus valores precisos é que serão usados no ajuste do que se pretende controlar. Para isso, associa-se a cada variável a sua correspondente variável linguística, de mesmo nome e assumindo valores linguísticos, tais como, baixa -B, média -M e alta -A, valores estes escolhidos com a conveniente partição nebulosa. Em cada universo linguístico associado a  $X$  definem-se as funções de pertinências  $\mu^j$ , sendo  $j$  um valor linguístico, por

$$\mu^j : X \rightarrow [0, 1]$$

que associa cada valor preciso de  $x \in X$  ao valor  $\mu^j(x)$ , chamado de grau de pertinência de  $x$  ao valor linguístico  $j$ , o que completa o módulo de 'fuzzificação'.

#### 3.5.2 O módulo de inferência

O principal módulo do controlador é o módulo de inferência que faz uso da operação de inferência, descrita na seção 3.4, a qual retira da sua matriz de regras cada uma das regras disparadas, pelos valores linguísticos de entrada, e dela infere os valores linguísticos de saída. Cada regra disparada é representada pela proposição  $p \rightarrow q$  da regra de inferência. É claro que quando a proposição  $p \rightarrow q$  é falsa o valor inferido é verdadeiro, pois  $p \rightarrow q$  representa a experiência de alguém, o operador, que tenha bastante conhecimento do processo para o qual se destina o controlador, já que este pretende substituir ao operador. Entretanto, não se pode excluir a possibilidade do operador tomar uma decisão errada.

#### 3.5.3 O módulo de 'defuzzificação'

Este módulo é responsável pela geração da informação precisa, extraída da informação nebulosa gerada no módulo de inferência, como resposta para que se realize o processo por completo.

## 3.6 Cadeia de Markov

Nesta seção procede-se a definição de cadeia de Markov e apresenta-se as propriedades que merecem ser destacadas a fim de facilitar a compreensão da ferramenta usada na análise de convergência do algoritmo em estudo.

A cadeia de Markov foi escolhida como ferramenta em virtude de sua abrangência tanto na modelagem como instrumento de análise, para os casos abordados.

### 3.6.1 Definições e propriedades

Um processo estocástico discreto  $\{X_t\}, t = 1, 2, \dots$ , com espaço de estados  $\Omega$ , é chamado de cadeia de Markov quando, para todo  $t$  e  $i_1, \dots, i_t \in \Omega$ , satisfaz à seguinte condição

$$P[X_t = i_t | X_{t-1} = i_{t-1}, X_{t-2} = i_{t-2}, \dots, X_1 = i_1] = P[X_t = i_t | X_{t-1} = i_{t-1}]$$

Sendo assim, define-se a chamada matriz de transição,  $P^{(t)}$ , ou matriz de um passo, de elementos

$$p_{ij}^{(t)} = P[X_t = i_t | X_{t-1} = i_{t-1}], \quad i, j \in \Omega$$

como sendo a matriz cujos elementos são as probabilidades de transições do estado  $i$  para o estado  $j$ , no instante  $t$ , quando a mesma satisfaz às seguintes condições

$$p_{ij}^{(t)} \geq 0 \text{ para todo } i, j \in \Omega \quad \text{e} \quad \sum_{j \in \Omega} p_{ij}^{(t)} = 1 \text{ para todo } i$$

### Cadeia homogênea versus não-homogênea

Uma cadeia é dita homogênea quando sua transição não depende de  $t$ , sendo representada por  $P$ . Quando a cadeia tiver transição dependente de  $t$  será a mesma dita não-homogênea, sendo representada por  $P^{(t)}$ .

Para uma distribuição de probabilidade  $\pi^0$ , sobre o conjunto de estados  $\Omega$ , a cadeia  $\{P^{(t)}\}_{t=1}^{\infty}$  fica completamente conhecida pela sequência  $\{\pi^{(t)}\}_{t=1}^{\infty}$ , de componentes  $\pi_j^{(t)}$ ,  $j \in \Omega$ , dada por

$$\pi^{(t)} = \pi^0 P^t \quad \text{quando a cadeia é homogênea}$$

pois sendo homogênea,  $P^{(t)} = P$ , e a matriz de transição de  $t$  passos é dada pelo produto  $P^t$ . Quando a cadeia é não-homogênea a mesma fica conhecida, pela equação

$$\pi^{(t)} = \pi^0 P^{(1)} \cdot P^{(2)} \cdot \dots \cdot P^{(t)}$$

ou ainda por

$$\pi^{(m,t)} = \pi^0 P^{(m+1)} \cdot P^{(m+2)} \cdot \dots \cdot P^{(t)}$$

Sendo markoviana, a cadeia, as equações apresentadas possibilitam a descrição adiantada da evolução da cadeia também chamada de cauda da cadeia, sendo esta de grande

utilidade na sua análise. Além disso, sendo a cadeia não-homogênea a equação possibilita o questionamento da dependência da convergência da cadeia à distribuição inicial, ou seja, quando se deseja saber se há convergência ou não de  $(\pi^{(m,t)})_{t=1}^{\infty}$ , podendo ainda ser independente ou não de  $\pi^0$ . Quando independente o comportamento é chamado de *perda de memória*.

Quando não houver convergência, mas há perda de memória, se diz que a cadeia é fracamente ergódica e quando houver convergência com perda de memória se diz que a cadeia é fortemente ergódica.

### Estados intercomunicantes e subcadeia

Um par de estados  $i, j$  é dito intercomunicantes quando existem instantes  $m$  e  $n$  tais que  $p_{ij}^{(m)} > 0$  e  $p_{ji}^{(n)} > 0$ .

Uma subcadeia de uma cadeia  $\{X_t\}, t = 1, 2, \dots$ , com espaço de estados  $\Omega = \{1, 2, \dots\}$ , é qualquer cadeia resultante da restrição de  $\{X_t\} \in \Omega'$ , onde  $\Omega' \subset \Omega$  é o espaço de estado da subcadeia. Sendo assim,  $\Omega'$  é chamado de conjunto fechado, no sentido da não intercomunicação de seus estados com qualquer outro estado de  $\Omega - \Omega'$ . Observe que  $\Omega'$  pode ter apenas um elemento e, neste caso, o seu estado é chamado de estado absorvente da cadeia.

### Irredutibilidade da cadeia

Uma cadeia é dita irredutível quando todos os seus estados são intercomunicantes.

### O período de uma cadeia

O período de um estado  $j$  é o inteiro,  $\rho$ , maior divisor comum do conjunto  $\{t | p_{jj}^{(t)} > 0\}$ , ou seja,  $\rho = \text{MDC}\{t | p_{jj}^{(t)} > 0\}$ . Quando  $\rho = 1$  diz-se que o estado é aperiódico.

Sendo a cadeia irredutível, pelo corolário II.2.1 de [Isaacson e Madsen 1976], quando afirma todos os seus estados têm o mesmo período.

Todos os estados de um subconjunto irredutível são do mesmo tipo

garante que todos os seus estados têm o mesmo período.

Uma cadeia é dita aperiódica quando é irredutível e todos os seus estados são aperiódicos ou quando redutível e os períodos de suas subcadeias não são múltiplos.

### A primeira visita a um estado

Seja,  $f_{ij}^t$ , a probabilidade da primeira visita ao estado  $j$  ocorrer no instante  $t$ , dado que  $X_k = i$ , ou seja

$$f_{ij}^t = P[X_{t+k} = j, X_{t+k-1} \neq j, \dots, X_{k+1} \neq j, X_k = i]$$

e toma-se,  $f_{ij}^0 = f_{ii}^0 = 0$ .

### Estado persistente versus estado transiente

Um estado  $j$  é dito persistente quando  $\sum_{t=1}^{\infty} f_{jj}^t = 1$  e transiente quando  $\sum_{t=1}^{\infty} f_{jj}^t < 1$ .

**Persistência positiva e nula**

Sendo  $\sum_{t=1}^{\infty} f_{jj}^t = 1$  e o valor esperado  $\sum_{t=1}^{\infty} t f_{jj}^t < \infty$  diz-se que o estado  $j$  é persistente positivo e quando  $\sum_{t=1}^{\infty} t f_{jj}^t = \infty$  diz-se que o estado  $j$  é persistente nulo.

**3.6.2 Ergodicidade da cadeia homogênea**

Esta característica da cadeia é identificada com a análise da mesma em tempos longos ou  $t$  grande, o suficiente para que se possa ter alguma previsibilidade sobre o comportamento futuro da mesma.

Quando, para todo  $i$ , são satisfeitas as condições

$$\lim_{t \rightarrow \infty} p_{ij}^t = \pi_j > 0 \quad (3.1)$$

$$\sum_{j \in \Omega} \pi_j = 1 \quad (3.2)$$

diz-se que a cadeia é ergódica

**3.6.3 Ergodicidade da cadeia não-homogênea**

Para o caso onde a cadeia é não-homogênea, sendo a transição dada por  $P^{(t)} = SC^{(t)}M^{(t)}$ , define-se

$$f^{(m,t)} = f^0 \cdot P^{(m+1)} \cdot P^{(m+2)} \cdot \dots \cdot P^{(t)}$$

e

$$g^{(m,t)} = g^0 \cdot P^{(m+1)} \cdot P^{(m+2)} \cdot \dots \cdot P^{(t)}$$

**A norma**

A norma usada, representada por  $\|\cdot\|$ , é definida por

$$\|\pi\| = \sum_{i \in \Omega} |\pi_i|, \quad \text{e} \quad \|P^{(t)}\| = \max_{i \in \Omega} \sum_{j \in \Omega} |p_{ij}^{(t)}|$$

com a qual define-se

**Ergodicidade fraca**

Uma cadeia  $P^{(t)}$  é dita fracamente ergódica quando, para todo  $m$ , vale

$$\limsup_{t \rightarrow \infty, f^0, g^0} \left\| f^{(m,t)} - g^{(m,t)} \right\| = 0$$

**Ergodicidade forte**

Uma cadeia  $P^{(t)}$  é dita fortemente ergódica quando, para todo  $m$ , existe um vetor  $g$

com  $\|g\| = 1$  e  $g_i \geq 0$  tal que

$$\limsup_{t \rightarrow \infty, g^0} \|f^{(m,t)} - g\| = 0$$

Uma outra grandeza, além da norma, também usada na análise de ergodicidade é o coeficiente de ergodicidade ou  $\delta$  de *Dobrushin*, o qual dá uma medida de distância entre as linhas da matriz de transição, definido a seguir.

**O coeficiente de ergodicidade ou  $\delta$  de *Dobrushin***

Sendo a matriz de transição da cadeia, com espaço de estado  $\Omega$  finito, dada por  $P^{(t)}$  define-se o coeficiente de ergodicidade de *Dobrushin*

$$\alpha(P^{(t)}) = 1 - \max_{i,j \in \Omega} \sum_{l \in \Omega} |p_{il}^{(t)} - p_{jl}^{(t)}|^+$$

onde  $|p_{il}^{(t)} - p_{jl}^{(t)}|^+ = \max\{0, p_{il}^{(t)} - p_{jl}^{(t)}\}$  e com este define-se

$$\delta(P^{(t)}) = 1 - \alpha(P^{(t)})$$

e, em decorrência de  $0 \leq \alpha(P^{(t)}) \leq 1$ , conclui-se que  $0 \leq \delta(P^{(t)}) \leq 1$ .

---

## Capítulo 4

# A modelagem e análise de convergência

---

Neste capítulo pretende-se obter a modelagem dos algoritmos. Para isso apresenta-se os algoritmos AGH e AGNH a serem modelados, para o problema no formato escolhido. Por último chega-se às condições que garantem a convergência, tanto para o AGH quanto para o AGNH.

### 4.1 O problema e o formato

Com o objetivo de definir a modelagem para os algoritmos acima, da forma mais abrangente possível, decide-se usar o problema de otimização no seguinte formato, tendo em vista que este modelo representa uma expressiva classe de problemas de otimização, cuja dificuldade de resolução está associada à cardinalidade do conjunto de soluções viáveis que cresce exponencialmente, como é o caso apresentado a seguir, onde esse crescimento se dá tanto com o crescimento de  $n$  quanto com crescimento de  $k$ , sem contar com o valor de  $t_p$  que é relativamente pequeno. Por isso, o problema é dado no seguinte formato

$$\max\{f(X^l) : X^l \in G\} \quad (4.1)$$

Admite-se que  $f$  é uma função positiva e limitada superiormente, sobre  $G$ , dado pelo produto cartesiano,  $G = X_1 \times X_2 \times \cdots \times X_n$ , obtidos da discretização dos contínuos,  $c(X_i)$ , domínios das variáveis  $x_i$ , para  $i = 1, \dots, n$ , com  $k$  escolhido a fim de garantir a precisão desejada.

Como o algoritmo depende da forma de representação do conjunto viável, a despeito da representação binária ter sido usada com maior frequência, talvez por facilidades na definição dos operadores, aqui decide-se pela representação em ponto-flutuante, já que esta não traz nenhum prejuízo para os fins desejados e possibilita uma abordagem mais genérica. Sendo assim,

$$X_i = \{x_{ij} \in X_i \mid i = 1, \dots, n\}$$

e o conjunto  $G$ , de cardinalidade  $2^{nk}$ , é constituídos dos pontos

$$X^l = (x_{1l}, \dots, x_{nl}), \quad x_{il} \in X_i, \quad l = 1, \dots, 2^{nk}$$

## 4.2 O AGH e o AGNH

Nesta seção apresenta-se os algoritmos AGP/AGH e AGNH a serem modelados, com o fim de permitirem as análises desejadas, e para tal usa-se uma versão elitista do algoritmo na qual o melhor elemento encontrado até o presente é mantido na população.

O algoritmo AGP/AGH, tal como descrito em [Goldberg 1989],[Holland 1975] e [Michalewicz 1992], parte de uma amostra de soluções viáveis  $Pop^{(0)}$ , de tamanho  $t_p$ , chamada de população, e evolui, no tempo  $t$ , realizando as operações de seleção, de cruzamento e de mutação, respectivamente, representadas por  $S$ ,  $C$  e  $M$ , sobre a população  $Pop^t$ , gerando, com isso, a sequência  $\{Pop^{(t)}\}_{t=1}^{\infty}$ .

Todos esses operadores são usados na busca do equilíbrio entre diversificação, a capacidade da população de se manter diversa a fim de evitar que o algoritmo fique preso a um ótimo local de baixa qualidade, e intensificação, a capacidade que o algoritmo tem de encontrar a melhor solução possível.

A dinâmica do algoritmo garante que a população resultante da aplicação destes operadores só dependa da população sobre a qual eles operam e não de toda a história do processo e, em consequência, como em [Rudolph 1994], garante que o algoritmo genético possa ser modelado por uma cadeia de Markov.

Em vista da multiplicidade de definições do algoritmo, referida na seção 1.1, a busca por um algoritmo com um bom desempenho, alternativo à versão padrão, AGP/AGH, quando se mantém os parâmetros fixos, tornou-se efetiva. Muitos trabalhos, nesse sentido, são encontrados na literatura e todos esses visam encontrar parâmetros que melhorem o desempenho do algoritmo. Nessas tentativas, alguns têm buscado respostas fazendo alterações na estrutura do algoritmo padrão, como em [Cerf 1996], quando faz o ajuste tendo como referência o tamanho da população ou como em [Greenwell et al. 1995] que busca a resposta na análise do operador mutação a fim de descobrir regiões nas quais há uma maior probabilidade do algoritmo encontrar o ótimo. Além destas outras tentativas têm sido feitas, como apontam os trabalhos de [Eiben e M. 1991, Nix e Vose 1992].

Quando a análise é feita a partir da modelagem do algoritmo por cadeia de Markov deve ser levado em conta que a cadeia que modela o algoritmo pode ser homogênea, como em [Rudolph 1994], mas também pode ser não-homogênea. Mas a não-homogeneidade, quando decorrente da mudança nas taxas de mutação e de cruzamento, pode ainda advir de técnicas precisas, como em [Campos, Pereira, Carlos e de Assis 2012, Campos, Pereira e Rojas Cruz 2012], mas também pode ser decorrente de técnicas imprecisas ou técnicas que fazem uso da lógica difusa, como em [Lee 1972, Cavalheiro 2003, Burdelis 2009]. O uso dessas técnicas, sejam precisas ou imprecisas, dão origem ao que aqui é chamado de AGNH mas, com o fim de destacar, tendo em vista o estudo de caso aqui apresentado, quando se tratar de técnica nebulosa o algoritmo será referido por AGNHn.

Por razões associadas à escolha da ferramenta de modelagem, esclarecidas ao tempo da definição da cadeia de Markov, somente o operador seleção não pode vir a depender do instante  $t$ , sendo o parâmetro  $t_p$  mantido fixo. Mas isto, a nosso ver, não deve trazer nenhum prejuízo ao desempenho do algoritmo pois a despeito de  $t_p$  não apenas refletir na cardinalidade da população mas também na sua diversidade, esta última pode ser melhor e adequadamente controlada pela mutação.



### Os operadores usados nos algoritmos

A seguir a descrição de cada um dos operadores usados no algoritmo AGH.

#### O operador seleção

O operador seleção opera sobre uma população de pontos

$$[X^{t_1}, X^{t_2}, \dots, X^{t_p}], \quad X^{t_i} \in G$$

e, tendo em vista que se busca o ponto ótimo, o faz com o uso de alguma medida de adequabilidade do ponto  $X^{t_i}$ , dada por  $p(X^{t_i})$ , que leve em conta o critério de otimalidade. Uma adequabilidade bastante usada é definida por

$$p(X^{t_i}) = \frac{f(X^{t_i})}{\sum_{j=1}^{t_p} f(X^{t_j})}, \quad i = 1, \dots, t_p$$

Tendo definidas as adequabilidades,  $p(X^{t_i})$ , um mecanismo de seleção, com reposição, é realizado com base numa roleta com setores de tamanhos  $p(X^{t_i})$ , que deve ser rodada  $t_p$  vezes, selecionando em cada rodada um elemento para a nova população.

Quando a escolha da adequabilidade levar a um super representante na população e, por isso, levar o algoritmo a uma indesejável convergência prematura, é o que passou-se a chamar de pressão seletiva. Entretanto, quando isso ocorrer, para tirar essa pseudo superioridade, tanto a técnica de escalamento da função objetivo, indicada em [Goldberg 1989], quanto outras técnicas, indicadas em [Michalewicz 1992], podem ser usadas com o fim de controlar essa pressão.

#### O operador cruzamento

A definição deste operador pede não somente a definição de  $tx_c$ , taxa de cruzamento, taxa essa dependente ou não de  $t$ , a determinar a quantidade de elementos que cruzarão, mas também dependente da escolha desses elementos na população, a definir quais elementos cruzarão, e, por último, depende do ponto de quebra,  $p_q$ , se for apenas um. Tendo sido escolhido o par  $X^i, X^j$  e o ponto de quebra, indicado por  $|$ , podendo este ser escolhido aleatoriamente, o cruzamento é feito em  $tx_c$  pares escolhidos na população, com a troca dos pontos

$$X^i = (x_{1_i}, \dots, x_{k_i}, |x_{(k+1)_i}, \dots, x_{n_i}) \text{ e } X^j = (x_{1_j}, \dots, x_{k_j}, |x_{(k+1)_j}, \dots, x_{n_j})$$

pelos pontos seguintes

$$X^{i'} = (x_{1_i}, \dots, x_{k_i}, x_{(k+1)_j}, \dots, x_{n_j}) \text{ e } X^{j'} = (x_{1_j}, \dots, x_{k_j}, x_{(k+1)_i}, \dots, x_{n_i})$$

Para não gerar indesejáveis tendências, a escolha dos pares pode ser feita considerando que os pontos da população são uniformemente distribuídos ou não, se assim desejar, como em [Goldberg 1989, Michalewicz 1992].

### O operador mutação

Este operador, cuja descrição pode ser vista em detalhes em [Michalewicz 1992], é o principal responsável pela diversidade da população e a taxa de mutação,  $tx_m$ , a determinação da quantidade de soluções sujeitas à mutação, pode depender ou não de  $t$ . Como no caso do cruzamento, este operador também depende da forma de representação da solução e quando esta representação é binária costuma-se fazê-la pela troca do bit escolhido.

#### Algoritmo AGH

Uma população inicial  $P$ , de cardinalidade  $t_p$ , é gerada e escolhe-se  $tx_c$ ,  $tx_m$  e  $p_q$ , sendo os mesmos mantidos fixos.

*Repita*

*Chame Seleção( $P$ ) retornando  $P_S$*   
*Chame Cruzamento( $P_S$ ) retornando  $P_{SC}$*   
*Chame Mutação( $P_{SC}$ ) retornando  $P_{SCM}$*   
 $P \leftarrow P_{SCM}$

*Até que* algum critério de parada seja satisfeito.

Uma versão não-padrão para o AG, com  $t_p$  fixo, e os demais parâmetros variando com  $t$ ,  $tx_c(t)$ ,  $tx_m(t)$  e  $p_q(t)$ , fica definida como segue.

#### Algoritmo AGNH

Uma população inicial  $P$ , de cardinalidade  $t_p$ , é gerada e define-se os valores iniciais para  $tx_c$ ,  $tx_m$  e  $p_q$ .

*Repita*

*Chame Seleção ( $P$ ) retornando  $P_S$*   
*Chame Cruzamento ( $P_S$ ), com  $tx_c$  e  $p_q$ , retornando  $P_{SC}$*   
*Chame Mutação ( $P_{SC}$ ), com  $tx_m$ , retornando  $P_{SCM}$*   
 $P \leftarrow P_{SCM}$   
*Chame a regra de geração de  $p_q$*   
*Chame a regra de ajuste para  $tx_c$*   
*Chame a regra de ajuste para  $tx_m$*

*Até que* algum critério de parada seja satisfeito.

## 4.3 A modelagem

Para se chegar à modelagem desejada observa-se inicialmente que os operadores  $S$ ,  $C$  e  $M$  do AGH podem ser modelados, cada um deles, por uma cadeia de Markov, como observou [Rudolph 1994], pois ficam bem definidos apenas com o conhecimento das populações sobre as quais eles operam. O conjunto de estados da cadeia é dado por  $\Omega = \{1, 2, \dots, 2^{nkt_p}\}$ , onde  $t_p$  é o tamanho das populações sobre as quais estes operadores atuam. Portanto, a transição da cadeia, que modela o AGH, se dá pela transição dada pelo produto  $\{(SCM)^{(t)}\}_{t=1}^{\infty}$ , sobre o conjunto de estados  $\Omega = \{1, 2, \dots, 2^{nkt_p}\}$ . Apenas os detalhes da modelagem de  $S$  são oferecidos porque esta é mandatária na análise desejada.

É importante observar que cada uma das matrizes  $S$ ,  $C$  e  $M$  pode ainda ser dependente de  $t$  ou não. Quando sim, diz-se que a cadeia é homogênea e quando não diz-se que a cadeia é não-homogênea. Sendo assim, quando a cadeia for homogênea será dada por  $P^{(t)} = \{(SCM)^{(t)}\}$ . Quando não-homogênea, a depender das características desejadas ao algoritmo a gerarem a não-homogeneidade, como aqui quando admite-se apenas que  $C$  e  $M$  dependem de  $t$ , para evitar variações na cardinalidade do conjunto de estados  $\Omega$ , levando as mesmas a  $C^{(t)}$  e  $M^{(t)}$ , resulta na cadeia

$$P^{(t)} = SC^{(t)}M^{(t)}$$

### A transição $S$

Para se chegar à transição  $S$  a partir da sequência de populações,  $\{Pop^t\}_{t=1}^\infty$ , definida por

$$Pop^t = [X^{t_1}, X^{t_2}, \dots, X^{t_p}]$$

sendo esta obtida de  $Pop^{t-1}$  pelo uso de uma roleta. A seguir, considere a sequência de variáveis aleatórias  $\{F^t\}_{t=1}^\infty$ , dada pelos arranjos

$$F^t = F^{t_1} F^{t_2} \dots F^{t_p}$$

onde cada variável aleatória  $F^{t_i}$ , assume valores em  $G$ , com  $i \in \Omega$  e, em consequência, a cardinalidade de  $\Omega$  é  $|\Omega| = 2^{nkt_p}$ . Sendo assim, define-se os conjuntos  $\Phi(F_j^t)$ , dado por

$$\Phi(F_j^t) = \{t_{j_1}, t_{j_2}, \dots, t_{j_{t_p}}\}$$

e a sequência  $\mathfrak{S}(F_j^t)$  dada por

$$\mathfrak{S}(F_j^t) = [t_{j_1}, t_{j_2}, \dots, t_{j_{t_p}}]$$

e com estes a matriz de transição,  $S = [s_{ij}]$   $i, j \in \Omega$ , dada por

$$s_{ij} = P[F^t = F_j^t | F^{t-1} = F_i^{t-1}]$$

ou ainda

$$s_{ij} = \begin{cases} 0 & \text{quando } \Phi(F_j^{t-1}) - \Phi(F_i^t) \neq \emptyset, \text{ para todo } j \\ \text{caso contrário} & \begin{cases} 1 & \text{quando } |\Phi(F_i^{t-1})| = 1 \\ \prod_{k \in \mathfrak{S}(F_j^t)} (1 - f^k)^{1-I(k)} (f^k)^{I(k)} & \text{caso contrário} \end{cases} \end{cases}$$

onde para cada arranjo  $F_j^t$  define-se o arranjo  $f^j = f^{j_1} f^{j_2} \dots f^{j_{t_p}}$  sendo

$$f^{j_i} = \frac{f(X^{j_i})}{\sum_{i=1}^{t_p} f(X^{j_i})} \quad j \in \mathfrak{S}(F_j^t)$$

com  $\emptyset$  representando o conjunto vazio,  $|*|$  representando a cardinalidade de  $*$  e  $\prod_{k \in I} f^k$  representando o produto de todos os  $f^{k_j}$  com  $j \in \mathfrak{S}(F_j^{t-1})$  para

$$I(k) = \begin{cases} 1 & \text{quando } k \in \mathfrak{S}(F_j^{t-1}) \\ 0 & \text{caso contrário} \end{cases}$$

#### A transição $C^{(t)}$

O cruzamento, cujos detalhes podem ser vistos em [Michalewicz 1992, Goldberg 1989], depende não somente de  $tx_c$ , a definir quantos pares cruzarão, mas também da escolha sobre a população, a definir quais pares cruzarão, e, por último, da forma como cruzarão, com a escolha do ponto de quebra,  $p_q$ , que define o ponto de troca de conteúdos entre os pares, podendo este ser gerado aleatoriamente e distintos, para cada par. Por isso, a transição  $C^t(Pop^t, tx_c^t, p_q)$ , sendo  $c_{ij}^t, i, j \in \Omega$ , será conhecida com a possibilidade da escolha

$$I(X^i, X^j) = \begin{cases} 1 & \text{quando o par } X^i, X^j, i, j \in \Phi(F_j^t) \text{ for escolhido para cruzar} \\ 0 & \text{caso contrário} \end{cases}$$

e, também, com a geração aleatória do inteiro  $0 \leq p_q \leq n$ , para se proceder o cruzamento, descrito acima. Observe que para se chegar a uma transição homogênea para  $C$ , ainda que  $tx_c$  seja fixo, será necessária a fixação do parâmetro  $p_q$ .

#### A transição $M^{(t)}$

A mutação, cujos detalhes podem ser vistos em [Michalewicz 1992, Goldberg 1989], depende não somente de  $tx_m$ , a dizer quantas componentes da população serão afetadas, mas também da escolha, na população, dessas componentes, a definir quais componentes mutarão. Por isso, sua transição  $M^t(Pop^t, tx_m^t)$ , sendo  $m_{ij}^t, i, j \in \Omega$ , será conhecida com as escolhas dadas por

$$I_{ij} = \begin{cases} 1 & \text{quando a componente escolhida a mutar for } i \in \mathfrak{S}(X^j), j \in \mathfrak{S}(F_j^t) \\ 0 & \text{caso contrário} \end{cases}$$

quando se procede a mutação da componente  $i$ .

## 4.4 Análise de convergência

As análises de AGH e AGNH são feitas separadas. Entretanto, o caso homogêneo pode ser alcançado a partir do caso não-homogêneo, como um caso particular.

### 4.4.1 Quando a cadeia é do tipo AGH

#### Sua aperiodicidade e sua irreduzibilidade

Tendo em vista que a população seguinte, definida pelos operadores, é construída por critérios aleatórios, não sendo permitida nenhuma regularidade nesta escolha, a conclusão sobre aperiodicidade é imediata. Quanto à irreduzibilidade, a mesma é garantida porque o operador seleção já garante intercomunicação entre qualquer população que tem pelo menos um dos pontos da população presente e, quando não tem, ainda assim, a intercomunicação fica garantida pela natureza aleatória dos operadores cruzamento e mutação.

#### Toda persistência de AGH é positiva

Sendo a cadeia persistente, em vista do elitismo usado, sendo também finita e irreduzível, o resultado fica garantido pelo lema III.2.1 de [Isaacson e Madsen 1976].

#### Sua ergodicidade

Os resultados acima garantem irreduzibilidade, aperiodicidade e persistência positiva, para a cadeia do AGH. Logo, tendo em vista o Teorema III.2.1 em [Isaacson e Madsen 1976] ao afirmar

Seja  $P = (p_{ij})$  a matriz de transição para uma cadeia de Markov que é irreduzível, persistente e aperiódica. Então, para todo  $i \in \Omega$  vale

$$(a) \begin{cases} \lim_{t \rightarrow \infty} p_{ij}^t = \frac{1}{\sum_{i=0}^{\infty} f_{ii}^t} & \text{se } \sum_{i=0}^{\infty} f_{ii}^t < \infty \\ 0 & \text{caso contrário} \end{cases}$$

$$(b) \begin{cases} \lim_{t \rightarrow \infty} p_{ij}^t = \frac{1}{\sum_{i=0}^{\infty} f_{ii}^t} & \forall j \in \Omega \text{ se } \sum_{i=0}^{\infty} f_{ii}^t < \infty \\ 0 & \text{caso contrário} \end{cases}$$

e também pelo Teorema III.2.2 de [Isaacson e Madsen 1976], quando afirma que

Seja  $P = (p_{ij})$  a matriz de transição de uma cadeia de Markov irreduzível, aperiódica e persistente positiva. Sendo  $\lim_{t \rightarrow \infty} p_{ij}^t$  conhecido e independente de  $i$ , toma-se

$$\pi_j = \lim_{t \rightarrow \infty} p_{ij}^t$$

sendo que  $\pi_j$ 's satisfazem às seguintes condições

$$\pi_j > 0, \sum_{j \in \Omega} \pi_j = 1 \quad \pi_j = \sum_{i \in \Omega} \pi_i p_{ij}$$

a ergodicidade da cadeia que modela o AGH, definida na seção 3.6.2, fica garantida.

### 4.4.2 Quando a cadeia é do tipo AGNH

Neste caso a ergodicidade da cadeia é definida através do coeficiente de ergodicidade da matriz  $P^{(t)} = SC^{(t)}M^{(t)}$ , tal como definido na seção 3.6.3.

**A finitude de  $\Omega$  em  $\delta(S)$  e a ergodicidade fraca de AGNH**

Sendo  $P^{(t)} = SC^{(t)}M^{(t)}$  e fazendo

$$P^{(m,t)} = P^{(m+1)} \cdot P^{(m+2)} \cdot \dots \cdot P^{(t)}$$

pelo Lema V.2.3. em [Isaacson e Madsen 1976] chega-se a

$$\delta(P^{(m,t)}) \leq \delta(P^{(m+1)})\delta(P^{(m+2)}) \dots \delta(P^{(t)}) \leq \delta(SC^{(t)}M^{(t)})$$

Em vista da finitude de  $\Omega$  existe um  $t_{max}$  a partir do qual os operadores de mutação e de cruzamento devem se tornar inertes, ou seja,  $P^{(t)} = S^{(t)} = S^t$  para  $t \geq t_{max}$ , logo

$$\lim_{t \rightarrow \infty} \delta(P^{(t)}) = \delta(\lim_{t \rightarrow \infty} S^t) = 0$$

Portanto, sendo  $\lim_{t \rightarrow \infty} \delta(P^{(m,t)}) = 0$ , pelo Teorema V.3.1 em [Isaacson e Madsen 1976], conclui-se que a cadeia é fracamente ergódica.

**Sua ergodicidade forte**

Sabe-se que a ergodicidade fraca é condição necessária para a ergodicidade forte e que, em virtude do espaço de estados,  $\Omega$ , ser finito, a cadeia tem pelo menos um *autovetor*  $\pi = \lim_{t \rightarrow \infty} \pi^{(t)}$ , com *autovalor* 1 e  $\|\pi\| = 1$ , para cada transição  $P^{(t)}$ . Sendo assim,

$$\sum_{j=1}^{\infty} \|\pi_j - \pi_{j+1}\| < \infty$$

e, pelo Teorema V.4.3 em [Isaacson e Madsen 1976], conclui-se que a cadeia de Markov que modela o AGNH é fortemente ergódica.

---

## Capítulo 5

# Um estudo de caso: AGH versus AGNHn

---

Neste capítulo apresenta-se a descrição do AGNHn, resultante do AGH pelo uso do controlador nebuloso, que ajustaria os parâmetros associados aos operadores de mutação,  $tx_m$ , e de cruzamento,  $tx_c$ . Este estudo de caso justifica-se não apenas pela constatação do resultado em si mas acredita-se que seja motivada pela defesa apresentada, a seguir.

O algoritmo genético é uma das metaheurísticas em que, a menos na busca local onde se pode usar uma heurística, a diversificação é realizada sem estabelecer nenhuma relação com o problema ao qual ela se dedica, pois apenas se faz a escolha aleatória de um elemento a mutar. Por isso, acredita-se que a introdução do controlador nebuloso, ao extrair informações do problema que vão definir o nível de diversificação a ser usado, em tempo de execução, pode dar ao algoritmo um caráter de algoritmo global e, em consequência, ter uma maior chance de encontrar um ótimo global.

### 5.1 O AGNHn

Esta versão parte da versão padrão, descrita na seção 4.2, e agrega o controlador nebuloso, definido para esse fim na seção 5.2, para ajustar apenas o parâmetro  $tx_m$ .

Para a realização de testes comparativos,  $AGH \times AGNHn$ , escolheu-se problemas que constam de testes realizados, tanto em [Michalewicz 1992] como em [Gonçalves 1997]. Estes problemas foram escolhidos por terem, alguns deles, muitos ótimos locais, para, nestes casos, testar a capacidade do algoritmo se livrar de ótimos locais de baixa qualidade. Ao mesmo tempo, por terem alguns dos problemas ótimos globais no interior e terem também, em outros casos, ótimos globais na fronteira, pretende-se testar a capacidade do algoritmo tanto para problemas irrestritos quanto para problemas restritos.

**Algoritmo AGNH**

Uma população inicial  $P$ , de cardinalidade  $t_p$ , é gerada e define-se os valores iniciais para  $tx_c$ ,  $tx_m$  e  $p_q$

Repita

Chame Seleção ( $P$ ) retornando  $P_S$

Chame Cruzamento ( $P_S$ ), com  $tx_c$  e  $p_q$ , retornando  $P_{SC}$

Chame Mutação ( $P_{SC}$ ), com  $tx_m$ , retornando  $P_{SCM}$

$P \leftarrow P_{SCM}$

Chame a regra de geração de  $p_q$

Chame a regra de ajuste para  $tx_c$

Chame a regra de ajuste para  $tx_m$

Até que algum critério de parada seja satisfeito.

## 5.2 O Controlador nebuloso usado no AGP

O controlador nebuloso seguinte destina-se a auxiliar o AGP, com o qual passa a ser chamado de AGNHn, com o fim de ajustar apenas o parâmetro taxa de mutação,  $tx_m$ . Portanto, sendo apenas uma variável a ser controlada, a  $tx_m$ , será esta a variável de saída do controlador e para esta estabelece-se sua dependência à diversidade da população,  $d_p$ , e também ao número de gerações,  $n_g$ , sendo as duas últimas as variáveis de entrada do controlador. Sua representação é a seguinte

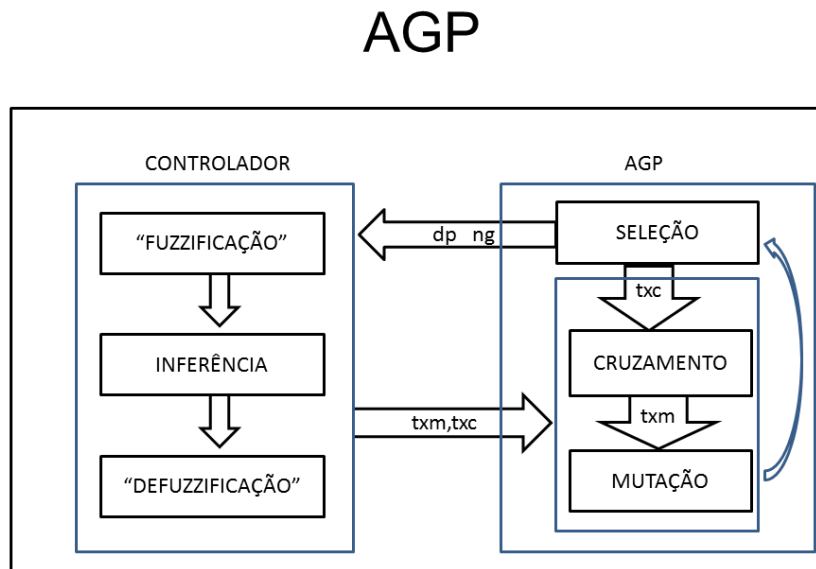


Figura 5.1: AGH + CONTROLADOR NEBULOSO = AGNHn



### 5.2.1 O módulo de 'fuzzificação', para o AGNHn

Aqui escolhe-se como variáveis de entrada a diversidade da população,  $d_p$  e o número de gerações,  $n_g$  e como variável de saída a taxa de mutação,  $tx_m$ . Para cada um dos universos  $d_p$ ,  $n_g$  e  $tx_m$  os valores linguísticos escolhidos foram baixo,  $B$ , médio,  $M$  e alto,  $A$ . É claro que a partição nebulosa de cada universo poderia ser outra, contendo um maior número de valores linguísticos. Sendo assim, define-se, com a partição nebulosa escolhida, as respectivas funções de pertinência, a seguir.

**Variáveis de entrada**

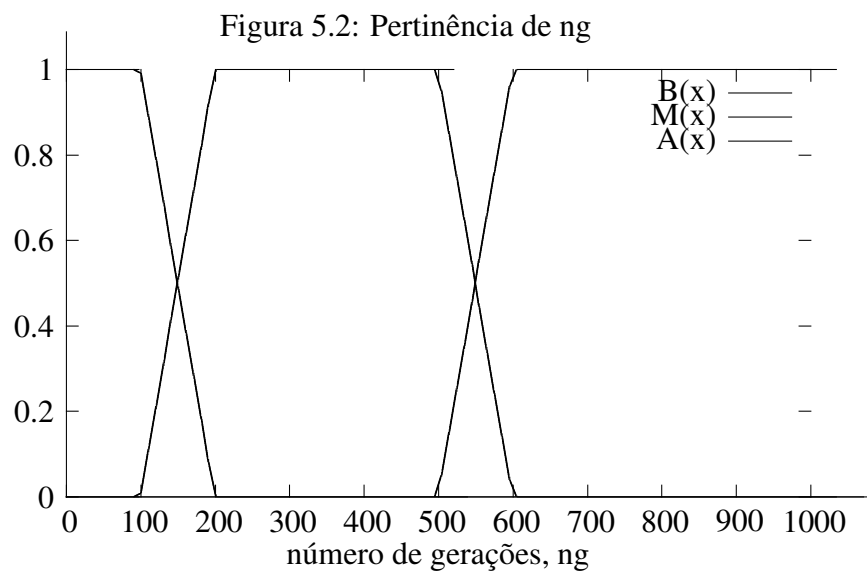
**As funções de pertinência da variável número de gerações,  $n_g$**

$$\mu^B(n_g) = \begin{cases} 1 & \text{para } 0 \leq n_g < 100 \\ -\frac{n_g-100}{100} + 1 & \text{para } 100 \leq n_g < 200 \\ 0 & \text{para } n_g \geq 200 \end{cases}$$

$$\mu^M(n_g) = \begin{cases} 0 & \text{para } 0 \leq n_g < 100 \\ \frac{n_g-100}{100} & \text{para } 100 \leq n_g < 200 \\ 1 & \text{para } 200 \leq n_g < 500 \\ -\frac{n_g-500}{100} + 1 & \text{para } 500 \leq n_g < 600 \\ 0 & \text{para } n_g \geq 600 \end{cases}$$

$$\mu^A(n_g) = \begin{cases} 0 & \text{para } 0 \leq n_g < 500 \\ \frac{n_g-500}{100} & \text{para } 500 \leq n_g < 600 \\ 1 & \text{para } n_g \geq 600 \end{cases}$$

cujo gráfico é o seguinte



### As funções de pertinência da diversidade da população, $d_p$

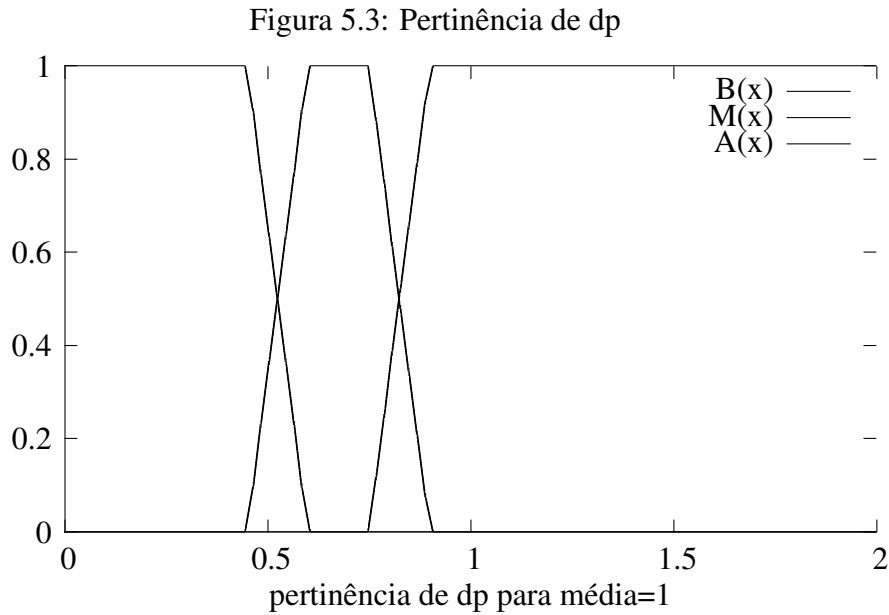
A diversidade da população,  $d_p$ , será a medida do estado da população presente a ser usada como medida da diversidade da população, e, portanto, a média

$$\bar{f} = \frac{\sum_{i=1}^{t_p} f(X_i)}{t_p}$$

será usada com este fim. Para isso, as seguintes funções foram definidas em domínios, com seus respectivos valores linguísticos, a fim de garantir uma medida central para 60% dos casos.

$$\begin{aligned} \mu_{\bar{f}}^B(d_p) &= \begin{cases} 1 & \text{para } 0 \leq d_p < 0.45\bar{f} \\ -\frac{d_p}{0.15\bar{f}} + 4 & \text{para } 0.45\bar{f} \leq d_p < 0.6\bar{f} \\ 0 & \text{para } d_p \geq 0.6\bar{f} \end{cases} \\ \mu_{\bar{f}}^M(d_p) &= \begin{cases} 0 & \text{para } 0 \leq d_p < 0.45\bar{f} \\ \frac{d_p}{0.15\bar{f}} - 3 & \text{para } 0.45\bar{f} \leq d_p < 0.6\bar{f} \\ 1 & \text{para } 0.6\bar{f} \leq d_p < 0.75\bar{f} \\ -\frac{d_p}{0.15\bar{f}} + 6 & \text{para } 0.75\bar{f} \leq d_p < 0.9\bar{f} \\ 0 & \text{para } d_p \geq 0.9\bar{f} \end{cases} \\ \mu_{\bar{f}}^A(d_p) &= \begin{cases} 0 & \text{para } 0 \leq d_p < 0.75\bar{f} \\ \frac{d_p}{0.15\bar{f}} - 5 & \text{para } 0.75\bar{f} \leq d_p < 0.9\bar{f} \\ 1 & \text{para } d_p \geq 0.9\bar{f} \end{cases} \end{aligned}$$

cujo gráfico é o seguinte



**Variável de saída****As funções de pertinências da taxa de mutação,  $tx_m$** 

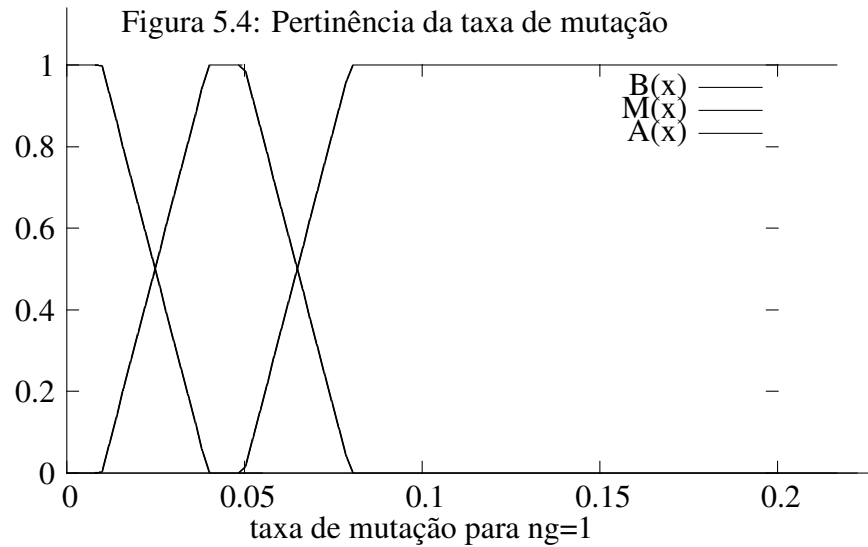
Observe que estas funções, de domínio  $[0, 1]$  são parametrizadas por  $n_g$  e esta parametrização foi assim obtida para reduzir possíveis trocas de valores linguísticos na resposta do controlador, principalmente com o crescimento de  $n_g$ .

$$\mu_{n_g}^B(tx_m) = \begin{cases} 1 & \text{para } 0 \leq tx_m < \frac{10^{-2}(2n_g-1)}{n_g} \\ -\frac{n_g}{3 \times 10^{-2}} tx_m + \frac{2}{3}(n_g+1) & \text{para } \frac{10^{-2}(2n_g-1)}{n_g} \leq tx_m < \frac{2 \times 10^{-2}(n_g+1)}{n_g} \\ 0 & \text{para } \frac{2 \times 10^{-2}(n_g+1)}{n_g} \leq tx_m \leq 1 \end{cases}$$

$$\mu_{n_g}^M(tx_m) = \begin{cases} 0 & \text{para } 0 \leq tx_m < \frac{10^{-2}(2n_g-1)}{n_g} \\ \frac{n_g tx_m}{3 \times 10^{-2}} + \frac{1}{3(1-2n_g)} & \text{para } \frac{10^{-2}(2n_g-1)}{n_g} \leq tx_m < \frac{2 \times 10^{-2}(n_g+1)}{n_g} \\ 1 & \text{para } \frac{2 \times 10^{-2}(n_g+1)}{n_g} \leq tx_m < \frac{10^{-2}(6n_g-1)}{n_g} \\ -\frac{n_g}{3 \times 10^{-2}} tx_m + 2n_g + 2/3 & \text{para } \frac{10^{-2}(6n_g-1)}{n_g} \leq tx_m < \frac{2 \times 10^{-2}(3n_g+1)}{n_g} \\ 0 & \text{para } \frac{2 \times 10^{-2}(3n_g+1)}{n_g} \leq tx_m \leq 1 \end{cases}$$

$$\mu_{n_g}^A(tx_m) = \begin{cases} 0 & \text{para } 0 \leq tx_m < \frac{10^{-2}(6n_g-1)}{n_g} \\ (\frac{n_g}{3 \times 10^{-2}}) tx_m - 2n_g + 1/3 & \text{para } \frac{10^{-2}(6n_g-1)}{n_g} \leq tx_m < \frac{2 \times 10^{-2}(3n_g+1)}{n_g} \\ 1 & \text{para } \frac{2 \times 10^{-2}(3n_g+1)}{n_g} \leq tx_m \leq 1 \end{cases}$$

cujo gráfico, para  $n_g = 1$ , vem a seguir.



### 5.2.2 O módulo de inferência, para o AGNHn

Neste caso, parte-se do seguinte conjunto de regras do tipo  $p \rightarrow q$  a constituírem a matriz de regras e representarem as decisões do especialista, tomadas em cada caso. Portanto, frente à proposição  $p$ , composta com os valores linguísticos  $d_p$  e  $n_g$ , a decisão a ser tomada será  $q$ , representando o valor linguístico inferido para  $tx_m$ , como descrito em 3.4. Estas decisões são definidas levando em conta que sempre que houver uma diminuição prematura de  $d_p$  a regra deve imprimir um aumento em  $tx_m$  e quando  $n_g$  for alto o valor de  $tx_m$  deve ser mantido ou até reduzido, mas nunca aumentado. A tabela seguinte foi sugerida a fim de contemplar estas condições.

Tabela 5.1: A matriz de regras para a taxa de mutação

$tx_m$		$n_g$				
		B	M	A	$B \wedge M$	$M \wedge A$
$d_p$	B	A	A	M	A	M
	M	A	M	B	M	B
	A	A	M	B	M	B
	$B \wedge M$	A	M	M	M	B
	$M \wedge A$	A	M	B	B	B

Em seguida, pela regra de inferência generalizada, dada por

$$[p' \wedge (p \rightarrow q)] \rightarrow q'$$

onde  $p$ ,  $p'$ ,  $q$  e  $q'$  são proposições nebulosas, chega-se à resposta deste módulo para  $tx_m$  correspondente ao valor linguístico, novamente inferido pela regra descrita em 3.4. Com base em  $tx_m$  chega-se à pertinência  $\mu^{out}$ , que será considerado no módulo de 'defuzzificação' seguinte.

### 5.2.3 O módulo de 'defuzzificação', para o AGNHn

Tendo recebido  $\mu^{out}$  do módulo de inferência, toma-se em seu domínio, por algum dos critérios descritos em [Takagi e Sugeno 1985, Pedrycz 1989, Mandani e Assilan 1999], um ponto para servir como resposta precisa do controlador, passando este ao algoritmo a fim do mesmo, na geração presente, realizar o operador de mutação. Neste caso, a regra usada foi a regra do centróide, dada por

$$\frac{\sum x_i \mu^{out}(x_i)}{\sum \mu^{out}(x_i)}$$

## 5.3 Os problemas testes

Os problemas seguintes são todos de maximização e foram escolhidos por terem graus de dificuldade diferentes, seja por terem muitos ótimos locais mas também por estarem esses ótimos no interior e também na fronteira, os mesmos constam de testes em [Michalewicz 1992] e [Gonçalves 1997], sendo os seguintes:

$$f_1(x_1, x_2) = 6 + x_1^2 - 3\cos(2\pi x_1) + x_2^2 - 3\cos(2\pi x_2), \text{ para } [-2, 1]^2$$

$$f_2(x_1, x_2) = 0.5 - \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{(10.001x_1^2 + x_2^2)^2}, \text{ para } [-1280/63, 1240/63]^2$$

$$f_3(x_1, x_2) = \frac{1}{0.3 + x_1^2 + x_2^2}, \text{ para } [-4, 1]^2$$

$$f_4(x_1, x_2) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_2^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times \\ \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)], \text{ para } [-2, 2]^2$$

$$f_5(x_1, x_2) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 - \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10, \text{ para } [5, 10] \times [0, 15]$$

$$f_6(x_1, x_2) = 3(1 - x_1)^2 \exp(-x_2^2 - (x_2 - 1)^2) + \left| 10(\frac{x_1}{5} - x_1^3 - x_2^5) \exp(x_1^2 - x_2^2) \right| + \\ + \frac{1}{3} \exp(-(x_1 - 1)^2 - x_2^2), \text{ para } [-2, 4]^2$$

### 5.3.1 Os resultados obtidos

Cada uma das duas versões do algoritmo genético, AGH e AGNHn, foi executada 100 vezes, para cada um dos problemas acima, e cada execução realiza 1000 iterações. Como se conhece, à priori, o ótimo do problema, a taxa de sucesso do algoritmo é a soma dos sucessos nas 100 execuções.

O código computacional, usado para geração dos seguintes gráficos, foi uma franquia do Professor André G. C. Pereira, do Departamento de Matemática - UFRN, e foi desenvolvido em R.

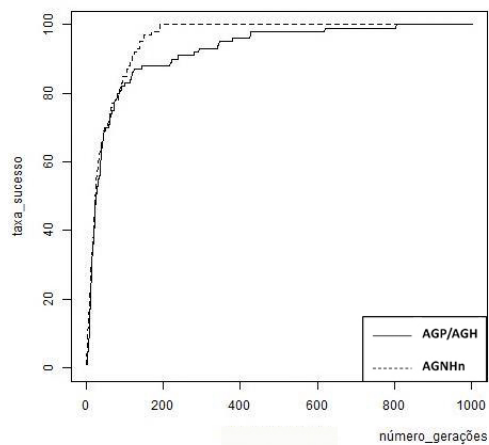


Figura 5.5: função 1

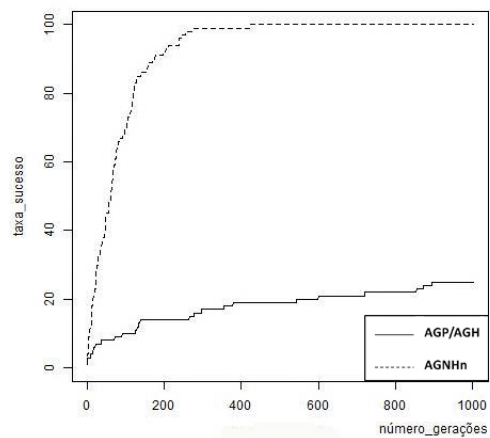


Figura 5.6: função 2

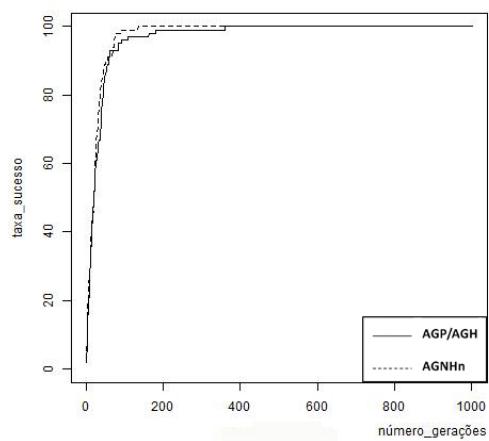


Figura 5.7: função 3

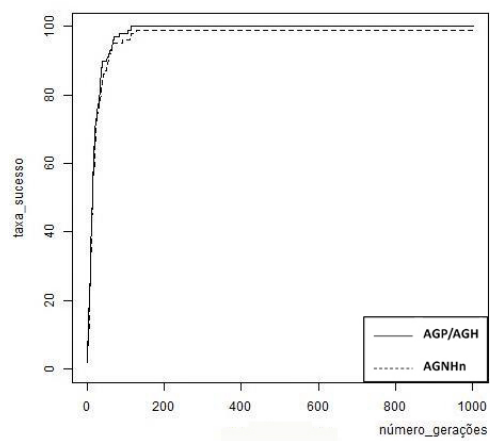


Figura 5.8: função 4

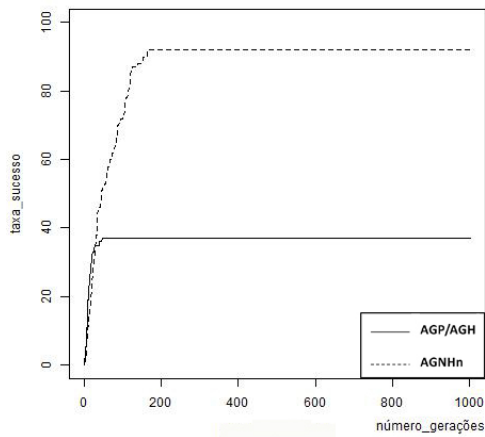


Figura 5.9: função 5

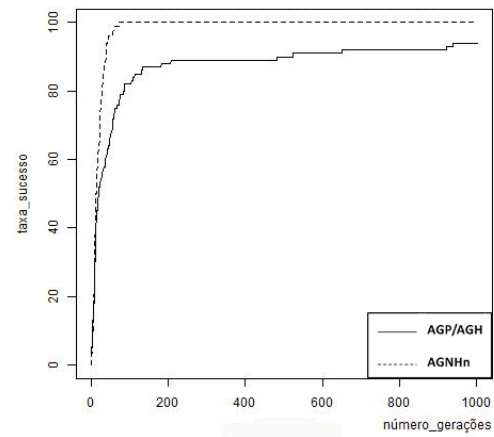


Figura 5.10: função 6

Nos resultados acima, observa-se que o AGP apesar de alcançar uma taxa de sucesso de 100% apenas nos problemas 1, 3 e 4, no problema 1 esta taxa só foi alcançada com  $n_g$  acima de 700, indicando, com isso, a dificuldade que o algoritmo teve em chegar ao ótimo global. Entretanto, nestes mesmos problemas o AGNHn não só alcançou a taxa de 100% mas o fez em um baixo  $n_g$ . Por outro lado, enquanto o AGNHn alcança uma alta taxa de sucesso, com um baixo  $n_g$ , chegando a 100% nos problemas 2 e 6 e próximo disso no problema 5, demonstrando, com isso, uma grande capacidade para chegar ao ótimo global, o AGP ficou, nestes mesmos problemas, preso a um ótimo local, tendo como resultado uma baixa taxa de sucesso.





---

## Capítulo 6

# Conclusões e perspectivas

---

### 6.1 Conclusões

A abordagem dada destaca a importância que a matriz  $S$  tem no alcance da ergodicidade da cadeia, levando as outras matrizes  $C(t)$  e  $M(t)$  a serem secundárias, nesse sentido. Além disso, não garante a convergência do algoritmo para um ótimo global, o que é compreensível, mas o faz transferindo aos operadores de mutação de cruzamento o ônus do insucesso, o que confirma a importância que o operador de mutação tem no alcance de um ótimo local de qualidade.

A análise assintótica não só esconde a importância prematura que os operadores, de cruzamento e de mutação, têm no processo mas também permite que se possa fazer  $tx_m \rightarrow 0$  e  $tx_c \rightarrow 0$ , com o aumento de  $n_g$ . Sendo assim, pode-se reafirmar que com o aumento do número de iterações cresce a esperança do algoritmo já ter gerado regiões verdadeiramente promissoras, não havendo mais necessidade, tampouco recomendação, para a diversificação e sim para intensificação. O mesmo pode-se afirmar sobre o operador cruzamento.

Nos testes computacionais realizados, confirma-se a grande capacidade que o algoritmo genético tem de encontrar ótimo global, a depender de um equilibrado ajuste de parâmetros, o que tem justificado o seu uso, com a frequente busca desse equilíbrio. Portanto, o uso do controlador nebuloso, no ajuste de seus parâmetros, pode ser uma estratégia usada a potencializar o algoritmo, na busca do ótimo global.

Para se ter a certeza de que o algoritmo genético, com a estratégia nebulosa, possa ter o comportamento próximo do comportamento de um legítimo algoritmo de otimização global, a exemplo do que sugere [Hendrix e Tóth 2010], serão necessários mais testes, além dos realizados neste trabalho.

### 6.2 Perspectivas

A tarefa de se ter um algoritmo que resolva globalmente qualquer problema de otimização parece inalcançável. Portanto, a busca por um algoritmo eficaz, ainda que não possa ser polinomial por imposição do problema a ser resolvido, passa necessariamente pela introdução, no algoritmo, de características associadas ao problema. Neste sentido, o AGNHn torna-se uma legítima tentativa de fazer o AGP um algoritmo dedicado.

Diante destas constatações e das possibilidades que a lógica nebulosa oferece, fica a depender da experiência a tarefa de encontrar uma adequada partição nebulosa, seus valores linguísticos, suas pertinências e uma adequada base de regras, para o problema, a darem ao AGNHn o desempenho mais próximo possível do que teria um legítimo algoritmo de otimização global.

---

## Referências Bibliográficas

---

- Armijo, L. (1966), ‘Minimization of functions having lipschitz continuous first partial derivatives.’, *Pacific J. Math.* (16), 1–3.
- Avriel, Mordecai (1976), *Nonlinear Programming Analysis and Methods*, Prentice-Hall, New Jersey.
- Bazaraa, Mokhtar S. e C. M. Shetty (1979), *Nonlinear Programming Theory and Algorithms*, John Wiley e Sons, New York.
- Burdelis, M. A. P. (2009), Ajuste de taxas de mutação e de cruzamento de algoritmos genéticos utilizando-se inferências nebulosas, Dissertação de mestrado, Universidade Politécnica de São Paulo, São Paulo, SP.
- Campos, V. S. M., A. G. C. Pereira e J. A. Rojas Cruz (2012), ‘Modelling the genetic algorithm by a non-homogeneous markov chain: Weak and strong ergodicity’, *Theory of Probability and its Applications* **57**(1), 185 – 192.
- Campos, V. S. M., A. G. C. Pereira, L. A. Carlos e I. A. S. de Assis (2012), ‘Algoritmo genético por cadeia de markov homogênea versus não-homogênea: Um estudo comparativo’, *Revista del Instituto Chileno de Investigación Operativa -ICHIO* **2**(1).
- Cavalheiro, Andreia Philipp (2003), Lógica difusa no controle de parâmetros do algoritmo genético para o problema do caixeiro viajante, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte, Natal, RN.
- Cerf, R. (1996), ‘A new genetic algorithm’, *Ann. Appl. Probability* **6**(3), 778–817.
- Chvátal, Vašek (1983), *Linear Programming*, W.H. Freeman and Company, New York.
- Cobham, A. (1964), The intrinsic computational difficulty of functions, *em* Y.Bar-Hillel, ed., ‘Proc. 1964 International Congress for Logic Methodology and Philosophy of Science’, North Holland, Amsterdam, pp. 24–30.
- Cook, S. A. (1971a), The complexity of theorem-proving procedures, *em* ‘Proc. 3rd Ann. ACM Symp. on Theory of Computing’, Association for Computing Machinery, New York, pp. 151–158.
- Dantzig, G. B. (1963), *Linear Programming and Extensions*, Princeton University Press, Princeton.

- Dikin, I. I. (1967), 'Iterative solution of problem of linear and quadratic programming', *Soviet Mathematic Doklady* (8), 674–675.
- Dorigo, M., V. Maniezzo e A. Colorni (1996), 'The ant system: Optimization by a colony of cooperating agents', *IEEE Transactions on Systems, Man, and Cybernetics Part* **26**(1), 29–41.
- Edmonds, J. (1965a), 'Paths, trees and flowers', *Canad. J. Math* (17), 449–467.
- Edmonds, J. (1965b), 'Minimum partition of a matroid into independent subsets', *J. Res. Nat. Bur. Standards Sect. B*(69), 67–72.
- Edmonds, J. e E. L. Johnson (1970), Matching: a well-solved class of integer linear programming, em Gordon e Breach, eds., 'Combinatorial Structures and their Applications', New York, pp. 89–92.
- Edmonds, J. e E. L. Johnson (1973), 'Matching, euler tours and chinese postman', *Math. Programming* (5), 88–124.
- Edmonds, J. e R. M. Karp (1972), 'Theoretical improvements in algorithmic efficiency for network flow problems', *J. Assoc. Comput. Mach.* (19), 248–264.
- Eiben, A. E., Aarts E. H. L. e Vann Hee K. M. (1991), 'Global convergence of genetic algorithms: A markov chain analysis', *Lecture Notes in Computer Science* **496**, 3–12.
- Fletcher, R. (1980), *Practical Methods of Optimization Unconstrained Optimization*, John Wiley e Sons, New York.
- Fletcher, R. (1981), *Practical Methods of Optimization Constrained Optimization*, John Wiley e Sons, New York.
- Garey, Michael R. e David S. Johnson (1979), *Computers and Intractability A guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York.
- Gill, Philip E., Walter Murray e Margaret H. Wright (1981), *Practical Optimization*, Academic Press, London.
- Glover, F (1986), 'Future paths for integer programming and links to artificial intelligence', *Computers and Ops. Res.* (5), 533–549.
- Glover, F. e M. Laguna (1993), Modern heuristic techniques for combinatorial problems, em C. R. Reeves, ed., 'Advanced Topics in Computer Science', Blackwell Scientific Publications, New York.
- Goldberg, David E. (1989), *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley Longman, Inc., Berkeley.
- Goldstein, A. A. (1965), 'On steepest descent', *SIAM J. Control* (3), 147–151.

- Gonçalves, C. R. (1997), Algoritmos de busca aleatória para otimização global: Estratégias de busca que preservam a distribuição assintótica, Tese de doutorado, Universidade de Brasília, Brasília-DF.
- Gonzaga, C. Clovis (1989), Algoritmos de pontos interiores para programação linear, 17 colóquio brasileiro de matemática, IMPA, Rio de Janeiro.
- Gonzaga, C. Clovis e L. A. Carlos (1990), A primal affine-scaling algorithm for linearly constrained convex programs, Relatório técnico, COPPE - SISTEMAS, Rio de Janeiro.
- Greenwell, R. N., Angus J. E. e M. Finck (1995), 'Optimal mutation probability for genetic algorithms', *Mathematical and Computer Modelling* **21**(8), 1–11.
- Hansen, P (1986), 'The steepest ascent mildest descent heuristic for combinatorial programming', *Congress on Numerical Methods in Combinatorial Optimization, Capri, Italy* (5), 533–549.
- Hendrix, Eligius M. T. e Boglárka G. Tóth (2010), *Introduction to Nonlinear and Global Optimization*, Springer, New York.
- Holland, J. H. (1975), 'Adaptation in natural and artificial systems', *University of Michigan Press, Ann Arbor* pp. 3–12.
- Isaacson, Dean L. e Richard W. Madsen (1976), *Markov Chains Theory and Applications*, John Wiley & Sons, New York.
- Karmarkar, N (1984), 'A new polynomial time algorithm for linear programming', *Combinatorica* (4), 373–395.
- Karp, R. M. (1972), Reductibility among combinatorial problems, *em* R. E. Miller e J. W. Thatcher, eds., 'Complexity of Computer Computations', New York, pp. 85–103.
- Khachiyan, L. G. (1979), 'A polynomial algorithm for linear programming', *Soviet Math. Doklady* (20), 191–194.
- Kirkpatrick, S., Gelatt C. D. e Vecchi M. P. (1983), 'Optimization by simulated annealing', *Science* (220), 671–680.
- Klee, V. e G. J. Minty (1972), How good is the simplex algorithm?, *em* O. Shisha, ed., 'Inequalities-III', New York, pp. 159–175.
- Kuhn, H. W. e A. W. Tucker (1951), Nonlinear programming, *em* J. Neyman, ed., 'Proceedings of the Second Berkley Symposium on Mathematical Statistics and Probability', Berkley, pp. 481–492.
- Lee, Michael (1972), Dynamic control, *em* O. Shisha, ed., 'Inequalities-III', New York, pp. 159–175.

- Lemarechal, C. (1980), *Extensions diverses des methods de gradient et applications*, Tese de doutorado, Paris IX.
- Mandani, E. H. e S. Assilan (1999), ‘An experiment in linguistic synthesis with a fuzzy logic controller’, *International Journal of Human-Computer Studies* **51**(2), 135–147.
- Michalewicz, Z. (1992), *Genetic Algorithms + Data + Structures = Evolution Programs*, Springer-Verlag, New York.
- Mladenovic, N. (1995), A variable neighborhood algorithm - a new metaheuristic for optimization combinatorial, *em* ‘Abstract of papers presented at Optimization Days’, 12, Montreal.
- Nelder, J. A. e R. Mead (1965), ‘A simplex method for function minimization’, *Computer Journal* (7), 308–313.
- Nemhauser, George L. e Laurence A. Wolsey (1988), *Integer and Combinatorial Optimization*, John Wiley and Sons, New York.
- Neto, J. C. R. (2010), *Modelagem dos algoritmos genético simples e simulated annealing por cadeia de markov*, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte, Natal, RN.
- Nix, A. E. e M. D. Vose (1992), ‘Modeling genetic algorithms with markov chains’, *Ann. of Mathematics and Artificial Intelligence* (5), 79–88.
- Papadimitriou, Christos H. (1982), *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall Inc., New Jersey.
- Pedrycz, W. (1989), *Fuzzy Control and Fuzzy Systems*, John Wiley, New York.
- Resende, M. G. C. e T. A. Feo (1995), ‘Greedy randomized adaptive search procedures’, *Journal of Global Optimization* **6**, 109 – 133.
- Ribeiro, Celso C. (1998), *Metaheurísticas*, Escola brasileira de computação, PUC-RIO, Rio de Janeiro.
- Rockafellar, R. T. (1972), *Convex Analysis*, Princeton University Press, Princeton.
- Rosen, J. B. (1960), ‘The gradient projection method for nonlinear programming, parte i - linear constraints’, *SIAM J. Appl. Math.* (8), 181–217.
- Rudolph, G. (1994), ‘Convergence analysis of canonical genetic algorithms’, *IEEE Transactions on Neural Networks* **5**(1), 96–101.
- Salkin, Harvey M (1975), *Integer Programming*, Addison-Wesley Publishing Company, Inc., Cleveland, Ohio.

- Takagi, T. e M. Sugeno (1985), 'Fuzzy identification of systems and its applications to modeling and control', *IEEE Trans. Systems Man Cybernet* (15), 116–132.
- Ye, Y. (1987), Interior Algorithms for Linear, Quadratic and Linearly Constrained Convex Programming, Tese de doutorado, Stanford University, Stanford, Ca.
- Zadeh, L. (1965), 'Fuzzy sets', *Information and Control* **8**, 338 – 353.