Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Computação Científica
Departamento de Matemática Aplicada

DISSERTAÇÃO DE MESTRADO

UM NOVO ALGORITMO GENÉTICO PARA A OTIMIZAÇÃO DE CARTEIRAS DE INVESTIMENTO COM RESTRIÇÕES DE CARDINALIDADE

Carlos Henrique Dias
Prof. Dr. Francisco de Assis Magalhães Gomes Neto - orientador

Campinas, SP
Março/2008

UM NOVO ALGORITMO GENÉTICO PARA A OTIMIZAÇÃO DE CARTEIRAS DE INVESTIMENTO COM RESTRIÇÕES DE CARDINALIDADE

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por **Carlos Henrique Dias** e aprovada pela comissão julgadora.

Campinas, 26 de Março de 2008.

Prof. Dr. Francisco de A. M. Gomes Neto
Orientador.

Banca Examinadora:

Prof. Dr. Francisco de Assis Magalhães Gomes Neto (IMECC-UNICAMP)
Prof.ª Dr.ª Márcia Aparecida Gomes Ruggiero (IMECC-UNICAMP)
Prof.ª Dr.ª Vitória Maria Miranda Pureza (DEP-UFSCar)

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica, UNICAMP, como requisito parcial para obtenção do Título de MESTRE em Matemática Aplicada.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Crisllene Queiroz Custódio – CRB8a 162/2005

Dias, Carlos Henrique
D543n Um novo algoritmo genético para a otimização de carteiras de investimento com restrições de cardinalidade / Carlos Henrique Dias -- Campinas, [S.P. : s.n.], 2008.

Orientador : Francisco de Assis Magalhães Gomes Neto
Dissertação (Mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Otimização de Carteiras de Investimento. 2. Algoritmos genéticos. 3. Restrições de cardinalidade. I. Gomes Neto, Francisco de Assis Magalhães. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

(cqc/imecc)

Título em inglês: A new genetic algorithm for portfolio optimization with cardinality constraints

Palavras-chave em inglês (Keywords): 1. Portfolio optimization . 2. Genetic algorithms. 3. Cardinality constraints

Área de concentração: Otimização

Titulação: Mestre em Matemática Aplicada

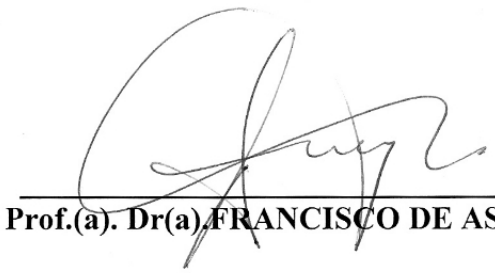
Banca examinadora: Prof. Dr. Francisco de Assis Magalhães Gomes Neto (IMECC/UNICAMP)
Profa. Dra. Vitória Maria Miranda Pureza (DEP/UFSCar)
Profa. Dra. Márcia Aparecida Gomes Ruggiero (IMECC/UNICAMP)

Data da defesa: 26/03/2008.

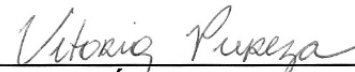
Programa de Pós-Graduação: Mestrado em Matemática Aplicada

Dissertação de Mestrado defendida em 26 de março de 2008 e aprovada

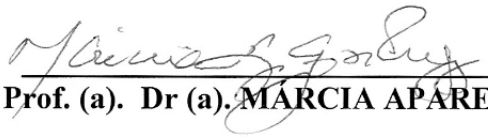
Pela Banca Examinadora composta pelos Profs. Drs.



Prof.(a). Dr(a).FRANCISCO DE ASSIS MAGALHÃES GOMES NETO



Prof. (a). Dr (a). VITÓRIA MARIA MIRANDA PUREZA



Prof. (a). Dr (a). MÁRCIA APARECIDA GOMES RUGGIERO

AGRADECIMENTOS

Agradeço primeiramente a DEUS por sempre atender minhas preces e permitir a realização desta grande conquista.

Em segundo, quero agradecer à minha noiva Vanessa que sempre me apoiou e compreendeu o significado da matemática para mim. Peço desculpas pelos inúmeros finais de semana que deixei de estar ao seu lado para estudar. Sou muito grato a sua família que sempre me tratou muito bem.

Aos meus país e irmã não sei nem como agradecer, agradeço a DEUS por ter uma família tão maravilhosa que sempre me apoiou em meus estudos, mesmo não entendendo muito bem do que se tratava.

Agradeço ao meu orientador e amigo Chico, que sempre acreditou em meu trabalho, mesmo não sendo um de seus melhores alunos. Sou muito grato pela sua grande paciência e boa vontade em solucionar minhas dúvidas.

Sou grato aos meus amigos Pedro e Rodrigo por sempre se disporem a ajudar-me quando precisei. Agradeço novamente ao Pedro por fornecer-me os resultados obtidos em seu trabalho de dissertação mestrado.

Finalmente, agradeço a todos os meus amigos e parentes que sempre torceram pelo meu sucesso.

RESUMO

Este trabalho tem por finalidade a determinação da fronteira eficiente de investimento através da otimização do modelo de média-variância com restrições de cardinalidade e limite inferior de investimento. Por tratar-se de um problema inteiro e não linear, cuja solução exata é de difícil obtenção, optamos por empregar um algoritmo genético, na linha desenvolvida por Chang *et al.* [3], que até hoje serve como referência para a determinação da fronteira eficiente de Pareto para problemas de otimização de investimentos. Entretanto, verificamos que o algoritmo proposto por Chang *et al.* apresenta uma distribuição não uniforme na geração de soluções aleatórias. Para contornar esse problema, introduzimos um novo esquema de geração de cromossomos, baseado na discretização do espaço, que permite a geração de soluções que satisfazem diretamente a restrição de montante total aplicado. Com essa nova abordagem, foi possível definir operadores de seleção, *crossover* e mutação bastante eficientes. Os resultados obtidos mostram que o novo algoritmo é mais robusto que aquele proposto por Chang *et al.*

Palavras-chave: Otimização de Carteiras de Investimento. Algoritmo Genético. Restrição de Cardinalidade.

ABSTRACT

In this work we consider the problem of determining of the efficient frontier of a portfolio using the mean-variance model subject to a cardinality constrain and to lower bounds on the amount invested in the selected assets. As this nonlinear integer programming problem is hard to solve exactly, we use a genetic algorithm, following the lines described by Chang *et al*. [3], still considered as a reference in the field. However, as the feasible solutions generated by the algorithm of Chang *et al*. are not uniformly distributed over the solution set, we introduce a new scheme for defining the chromosomes, based on the discretization of the feasible region, so that the amount invested always sum up to one for every solution obtained by the algorithm. This new approach allows us to define very efficient selection, crossover and mutation procedures. The numerical results obtained so far show that the new method is more robust than the one proposed by Chang *et al*.

Keywords: Portfolio Optimization. Genetic Algorithms. Cardinality Constraints.

SUMÁRIO

INTRODUÇÃO

O problema de otimização de carteiras de investimento consiste na seleção de um conjunto de ativos que dê ao investidor um retorno esperado, minimizando o risco. Deste modo, a carteira de investimento (*portfolio*) pode ser composta por vários ativos do mercado financeiro, em diferentes quantidades.

Para alguns investidores, a montagem de uma carteira de investimento tem como objetivo principal a diminuição do risco associado à variação dos preços dos ativos. Já outros investidores, que preferem um ganho um pouco maior, montam sua carteira de forma a manter um nível de retorno, evitando grandes riscos. Obviamente, existem também os investidores arrojados, aqueles que montam sua carteira apenas visando os ganhos, ou seja, sem dar importância ao risco. Em suma, o problema de otimização de carteiras de investimento possui uma variável que somente o investidor pode fornecer, que é a disposição que cada um tem para o risco. Desta forma, os modelos de otimização de carteiras de investimento devem ser capazes de fornecer não uma, mas uma família de soluções ótimas, ou seja, uma fronteira eficiente de investimento.

A fronteira eficiente de investimento é uma curva que mostra ao investidor quais são os possíveis ganhos para vários níveis de risco. Assim, a solução de um modelo de otimização de carteiras de ações será útil para qualquer tipo de investidor: arrojado, conservador, etc. Fica claro, então, que uma das grandes utilidades da fronteira eficiente de investimento é fornecer ao

investidor as melhores soluções de forma que este possa escolher o melhor investimento conforme sua aversão ao risco.

O desenvolvimento de modelos de otimização de *portfolios* teve origem na área econômico-financeira. Uma das principais contribuições nesta área foi o modelo de Média-Variância proposto por Markowitz [12], que teve, e ainda tem, uma boa aceitação por parte dos investidores, pois utiliza princípios bem simples de análise do passado para determinar o futuro, além de mostrar que o retorno esperado e a volatilidade (ou risco) dos prováveis retornos são aspectos cruciais na definição da carteira ótima.

O modelo média-variância de Markowitz passou por diversas modificações, em várias linhas. Por exemplo, Konno [9] propôs um modelo de otimização de carteiras de investimento usando uma função linear por partes que mede o desvio absoluto da média de retorno de cada um dos ativos, ou seja, sem utilizar a covariância. Este modelo tem a vantagem de exigir apenas a resolução de um problema de programação linear, ao invés de trabalhar com um problema quadrático, como na formulação de Markowitz. Entretanto, Mitra *et al.* [15] julgam que este benefício não supera a perda de um medidor de risco como a covariância.

Tentando melhorar a mensuração do risco, Konno & Suzuki [10] e Markowitz *et al.* [13] trabalharam com funções objetivo mais complexas, baseadas nas noções de *skewness*[1] e semi-variância[2], respectivamente.

A necessidade de tornar os modelos mais próximos da realidade do mercado de ações fez com que restrições mais complexas fossem inseridas no modelo de média-variância. Conseqüentemente, gerou-se um aumento na dificuldade da resolução destes modelos por métodos exatos. Com a crescente complexidade dos modelos, alguns dos quais classificados como *NP-Difíceis*, diversos autores procuraram utilizar os algoritmos meta-heurísticos para resolução dos problemas de otimização. É nessa linha que se destaca o artigo de Chang *et al.* [3], um dos mais citados na determinação da fronteira eficiente de investimento, que propõe três

---

[1] *Skewness* é uma medida de simetria de dados, usada porque, segundo Konno *et al.*, as taxas de retorno dos investimentos não são simetricamente distribuídas.

[2] Semi-variância é uma medida assimétrica de risco que leva em conta a variância somente para retornos abaixo de um limite pré-estabelecido.

meta-heurísticas (algoritmos genéticos, busca tabu e têmpera simulada) para a resolução de seu modelo de otimização, composto por variáveis inteiras e reais.

Outras propostas envolvendo modelos de programação inteira foram feitas por Rolland [17] e Schaerf [18], que utilizam a busca tabu, por Kellerer & Maringer [8], que utilizam uma estratégia híbrida que combina princípios da têmpera simulada com os algoritmos evolucionários, e por di Gaspero *et al.* [6] e Moral-Escudero *et al.* [16] que também utilizam uma estratégia híbrida, porém combinando algoritmos meta-heurísticos com um algoritmo exato de programação quadrática.

Neste trabalho, concentramo-nos na formulação de um novo algoritmo genético para a determinação da fronteira eficiente de investimento. No próximo capítulo, fazemos uma introdução ao modelo de Markowitz e à fronteira eficiente. O capítulo 2 é devotado ao algoritmo de Chang *et alii* [3]. Uma breve discussão sobre o espaço de soluções e as desvantagens do método de Chang *et alii* é apresentada no capítulo 3. No capítulo 4, mostramos como discretizar o espaço de soluções, técnica que serve de base para um novo algoritmo genético, proposto no capítulo seguinte. No capítulo 6, descrevemos uma reformulação da estratégia de representação das soluções factíveis, com o propósito de acelerar o algoritmo. Finalmente, o capítulo 7 apresenta os resultados obtidos com a aplicação do novo algoritmo a problemas reais, além de uma análise comparativa que mostra que este novo algoritmo é bem melhor que aquele proposto por Chang *et alii*.

CAPÍTULO 1

O MODELO MÉDIA-VARIÂNCIA DE MARKOWITZ

1.1 INTRODUÇÃO

O modelo de Markowitz [12] é um modelo de otimização multi-objetivo usado para equilibrar o retorno esperado e o risco de um *portfolio*, ou carteira de investimento, que é o conjunto de ativos financeiros nos quais se investe. Para um investidor, o retorno e a instabilidade de ativos são aspectos cruciais na escolha de um *portfolio*. Markowitz usa medidores estatísticos de esperança e variância de retorno para descrever, respectivamente, os benefícios e os riscos associados a um investimento. Deste modo, o objetivo pode ser minimizar o risco para um dado nível de retorno ou maximizar o retorno para um dado nível de risco.

1.2 MODELO DE MARKOWITZ

Em geral, os ativos que compõem o mercado financeiro possuem um histórico de retorno. Markowitz utiliza este histórico para construir seu modelo de média-variância. Este modelo é baseado no emprego de medidores estatísticos de média de retorno e de variância, tanto para os ativos e quanto para os *portfolios*. Estes medidores são descritos a seguir. Para mais detalhes, vide Luenberger [11].

1.2.1 Média de retorno dos ativos.

A média de retorno dos ativos, em termos estatísticos, é representada pela esperança de retorno que cada um dos ativos possui, ou seja,

$$E(r_i) = \frac{1}{T}\sum_{t=1}^{T} r_i^t,$$

onde r_i^t é o retorno do ativo i no período t e T é o número de períodos considerados. Por simplicidade, adotaremos o termo μ_i para representar a média de retorno de um ativo i.

1.2.2 Variância da média de retorno dos ativos.

Infelizmente, a média de retorno não é suficiente para a escolha de um investimento, uma vez que alguns ativos, embora tenham uma boa média de retorno, podem possuir uma

variação muito acentuada de preços. Um bom medidor de variação de dados é o desvio padrão, que é dado por

$$\sigma_i = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(r_i^t - \mu_i\right)^2}\ .$$

Para simplificar o modelo de otimização, o medidor de variação de dados usualmente utilizado é a variância, que nada mais é que o desvio padrão ao quadrado, ou seja,

$$\sigma_i^2 = \frac{1}{T}\sum_{t=1}^{T}\left(r_i^t - \mu_i\right)^2.$$

1.2.3 Covariância entre os ativos.

Como quaisquer dois ativos que compõem o mercado financeiro podem possuir uma correlação, torna-se necessária a utilização de um medidor de relação mútua entre duas variáveis. O medidor mais comumente empregado é a covariância. A covariância entre um ativo *i* e um ativo *j* é dada por

$$\sigma_{ij} = \frac{1}{T}\sum_{t=1}^{T}\left(r_i^t - \mu_i\right)\left(r_j^t - \mu_j\right).$$

Obviamente, a covariância entre um ativo *i* e ele próprio é a variância, ou seja, $\sigma_{ii} = \sigma_i^2$.

A covariância também pode ser determinada pelo coeficiente de correlação ρ_{ij}, que também é um medidor de relação mútua entre duas variáveis, com a diferença que $\rho_{ij} \in [-1, 1]$. Para descrever a covariância entre um ativo *i* e um ativo *j* em função do coeficiente de correlação, usamos

$$\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j\ . \tag{1.1}$$

1.2.4 Média de retorno de um *portfolio*.

Embora tenhamos tratado, até agora, da média de retorno de um ativo, também é possível determinar a média de retorno de uma carteira de investimento, ou seja, a média de retorno do investimento feito em um conjunto de ativos.

O retorno para cada um dos T períodos de um dado investimento é

$$\begin{aligned} r^1 &= w_1 r_1^1 + w_2 r_2^1 + \dots w_N r_N^1 \\ r^2 &= w_1 r_1^2 + w_2 r_2^2 + \dots w_N r_N^2 \\ \vdots \quad &\quad \vdots \quad \vdots \quad \vdots \\ r^T &= w_1 r_1^T + w_2 r_2^T + \dots w_N r_N^T \ , \end{aligned}$$

onde w_i é a proporção de investimento no ativo i. Logo, a média de retorno do *portfolio* nos T períodos é

$$\bar{r} = \frac{1}{T} \sum_{t=1}^{T} r^t .$$

Como, para o cálculo da média de retorno do *portfolio*, as proporções de investimento w_i são as mesmas em todos os períodos, temos

$$\begin{aligned} \bar{r} &= \sum_{i=1}^{N} w_i \left(\frac{1}{T} \sum_{t=1}^{T} r_i^t \right) \\ &= \sum_{i=1}^{N} w_i \mu_i . \end{aligned} \tag{1.2}$$

Portanto, como a média de retorno de cada ativo, μ_i, independe da proporção de investimento, w_i, é possível determinar as proporções escolhendo apenas um nível de retorno

para a carteira de investimento. As proporções de investimento no modelo de Markowitz são determinadas, desta mesma forma, a partir de um nível de retorno.

1.2.5 Variância da média de retorno do *portfolio*.

Além de determinar a média de retorno de um *portfolio*, é importante calcular a variância deste retorno, que é dada por

$$\sigma^2 = \frac{1}{T}\sum_{t=1}^{T}\left(r^t - \bar{r}\right)^2.$$

Usando a definição de r^t e a equação (1.2), obtemos

$$\begin{aligned}
\sigma^2 &= \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{i=1}^{N} w_i r_i^t - \sum_{i=1}^{N} w_i \mu_i\right)^2 \\
&= \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{i=1}^{N} w_i \left(r_i^t - \mu_i\right)\right)^2 \\
&= \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{i=1}^{N} w_i \left(r_i^t - \mu_i\right)\right)\left(\sum_{j=1}^{N} w_j \left(r_j^t - \mu_j\right)\right) \\
&= \frac{1}{T}\sum_{t=1}^{T}\sum_{i,j=1}^{N} w_i w_j \left(r_i^t - \mu_i\right)\left(r_j^t - \mu_j\right) \\
&= \sum_{i,j=1}^{N} w_i w_j \frac{1}{T}\sum_{t=1}^{T}\left(r_i^t - \mu_i\right)\left(r_j^t - \mu_j\right) \\
&= \sum_{i,j=1}^{N} w_i w_j \sigma_{ij}\,. \qquad (1.3)
\end{aligned}$$

Este resultado mostra como a variância da média de retorno do *portfolio* pode ser calculada a partir da covariância dos retornos dos ativos. A esta medida de variação de retorno, representada por σ^2, damos o nome de risco de investimento do *portfolio*.

1.2.6 O modelo média-variância de Markowitz.

O modelo média-variância de Markowitz tem como objetivo a minimização do risco, dado por (1.3), mantendo igual a R a média de retorno do *portfolio*, dada em (1.2).

Assim, supondo que o mercado financeiro possua N ativos disponíveis para investimento e que um investidor deseje obter um retorno R, o modelo clássico de média-variância proposto por Markowitz para este tipo de problema será

$$\begin{aligned} \min_{w} \ & \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij} \\ s.a \quad & \sum_{i=1}^{N} w_i \mu_i = R \\ & \sum_{i=1}^{N} w_i = 1 \\ & 0 \le w_i \le 1, \quad i = 1, \ldots, N, \end{aligned} \tag{1.4}$$

onde w_i é a proporção de investimento no ativo i, μ_i é o retorno esperado no ativo i, e σ_{ij} é a covariância entre os ativos i e j.

Com a resolução do modelo (1.4) para vários níveis de retorno R, é possível montar uma fronteira eficiente de investimentos, ou "linha crítica de investimento" (vide Mitra *et al.* [15]), conforme mostrado na Figura 1.

1.3 A FRONTEIRA EFICIENTE DE INVESTIMENTOS

A fronteira eficiente de investimento (F.E.I.) ou fronteira eficiente de Pareto é uma curva não decrescente que contém as melhores possibilidades de conciliação entre risco e retorno.

Na Figura 1 apresentamos uma fronteira eficiente de investimento traçada a partir de ativos da bolsa de valores do Reino Unido.

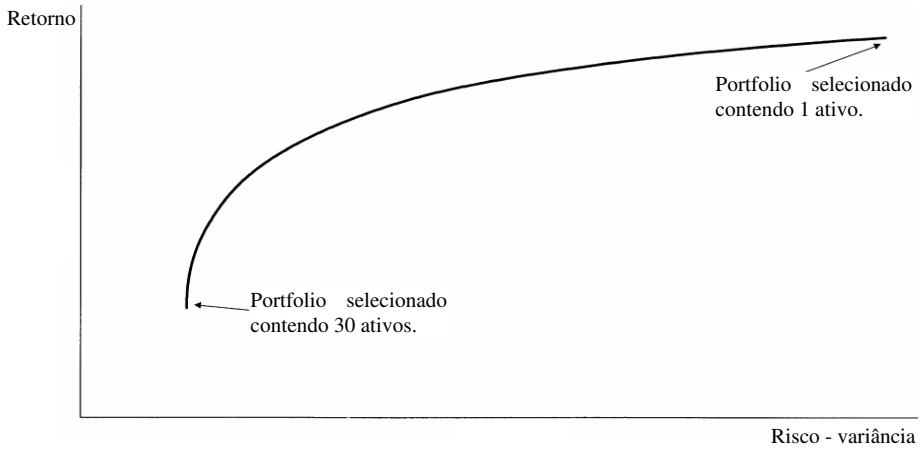


Figura 1 – F.E.I. dos ativos da bolsa de valores do Reino Unido (UK FTSE).

Fonte: T.–J. Chang, N. Meade, J.E. Beasley, Y.M. Sharaiha; **Heuristics for cardinality constrained portfolio optimisation.** Computers & Operations Research 27: 2000, p. 1281.

Na Figura 1, o eixo das ordenadas representa a média de retorno do *portfolio* e o eixo das abscissas a variância da média de retorno (ou risco) do *portfolio*.

A curva da Figura 1 mostra as diversas possibilidades de investimento em termos do retorno esperado e da aversão ao risco. Em geral, se o investidor tem aversão ao risco, deverá procurar uma carteira que possua uma grande diversificação de ativos. Por outro lado, se o investidor é mais audaz, normalmente optará por uma carteira pouco diversificada. Observa-se, assim, a utilidade da fronteira eficiente de investimento na análise prática das condições do mercado financeiro.

Outro modelo muito utilizado na prática para a determinação da fronteira eficiente de investimento é o modelo parametrizado, definido por

$$\begin{aligned} \min_{w} \quad & \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1-\lambda) \sum_{i=1}^{N} w_i \mu_i \\ s.a \quad & \sum_{i=1}^{N} w_i = 1 \\ & 0 \le w_i \le 1, \quad i = 1, \ldots, N, \end{aligned} \tag{1.5}$$

onde $\lambda \in [0, 1]$ é o parâmetro que determina o nível de aceitação de risco.

Para valores de λ próximos de zero, o risco é considerado pouco importante, de modo que o investidor está mais preocupado, neste caso, com o retorno. Já valores de λ próximos de um evidenciam um investidor mais preocupado com risco que com o retorno. Assim, a escolha do λ deve ser feita de forma a considerar a preocupação que cada investidor tem com o risco e com o retorno.

No modelo (1.5), a fronteira eficiente de investimento é construída a partir da variação de λ. Na prática, os valores de λ são determinados a partir da partição do intervalo $[0, 1]$. Considerando, por exemplo, o número de partições igual a *E*, temos $\lambda \in \left\{0, \frac{1}{E-1}, \frac{2}{E-1}, \cdots, \frac{E-1}{E-1}\right\}$.

Como foi observado por Chang *et al.* [3], a fronteira eficiente obtida pelo modelo clássico (1.4) é exatamente igual àquele obtida a partir do modelo parametrizado (1.5).

1.4 CONSIDERAÇÕES PRÁTICAS SOBRE O MODELO DE MÉDIA-VARIÂNCIA

O modelo de média-variância descrito anteriormente fornece uma formulação matemática adequada para o problema de decisão de investimento em sua forma mais geral. Porém, em situações reais, é necessário considerar alguns aspectos práticos como o desejo do investidor em montar uma carteira com um número pré-estabelecido de ativos (restrição de cardinalidade), ou o estabelecimento, pelo mercado financeiro, de uma proporção mínima de investimento em determinados ativos, ou ainda a venda de ações somente em lotes.

Embora muitas outras restrições estejam presentes no mercado real, vamos considerar, neste trabalho, apenas as restrições de cardinalidade e de limite mínimo de proporção de investimento, dadas, respectivamente, por

$$\sum_{i=1}^{N} z_i = K \tag{1.6}$$

e

$$w_i \geq \varepsilon_i z_i, \quad i = 1, \ldots, N, \tag{1.7}$$

onde z_i é uma variável binária que vale zero se o ativo *i* não pertence à carteira e vale 1 se o ativo pertence à carteira, *K* é o número de ativos que a carteira deverá possuir e ε_i é a fração mínima do montante disponível que deve ser investida no ativo *i*, caso este faça parte da carteira.

Adicionando as restrições (1.6) e (1.7) ao modelo parametrizado (1.5), obtém-se

$$\begin{aligned} \min_{w} \quad & \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1-\lambda) \sum_{i=1}^{N} w_i \mu_i \\ s.a \quad & \sum_{i=1}^{N} w_i = 1 \\ & \sum_{i=1}^{N} z_i = K \\ & w_i \geq \varepsilon_i z_i, \qquad i = 1, \ldots, N. \\ & z_i \in \{0, 1\}, \quad i = 1, \ldots, N. \end{aligned} \tag{1.8}$$

1.5 FRONTEIRA EFICIENTE DE INVESTIMENTOS – MODELO RESTRITO

Da mesma forma como foi feito para o modelo sem restrição de cardinalidade, podemos traçar a fronteira eficiente para o modelo restrito a partir da variação do valor de λ. Porém, apesar desta curva continuar sendo não decrescente, como a do modelo (1.4), ela não é mais contínua.

Para ilustrar esta curva, consideremos o seguinte exemplo. Suponha que existam quatro tipos de ativos disponíveis para investimento, com média de retorno, desvio padrão e coeficiente de correlação conforme a Tabela 1.

Tomemos a restrição de cardinalidade $K = 2$ (a carteira poderá ser composta somente por dois ativos) e desconsideremos a existência de limite mínimo de proporção de investimento para cada um dos ativos. Considerando todos os possíveis valores dos coeficientes w_i, construímos todas as possíveis combinações de dois ativos.

Tabela 1 – Média de retorno, desvio padrão e correlação para o exemplo de quatro ativos.

Ativo	Retorno	Desvio	Matriz de Correlação			
	(Média)	Padrão	1	2	3	4
1	0.004798	0.046351	1	0.118368	0.143822	0.252213
2	0.000659	0.030586		1	0.164589	0.099763
3	0.003174	0.030474			1	0.083122
4	0.001377	0.035770				1

A Figura 2 representa o universo de possíveis *portfolios* (carteiras) compostos por dois ativos. Certos *portfolios* são dominados, ou seja, são tais que existe um outro *portfolio* com menor risco e maior retorno. Por exemplo, todos os *portfolios* da curva que começa no ativo 4 e termina no ativo 1 são dominados. Se retirarmos, da Figura 2, os *portfolios* dominados, teremos a fronteira eficiente representada na Figura 3.

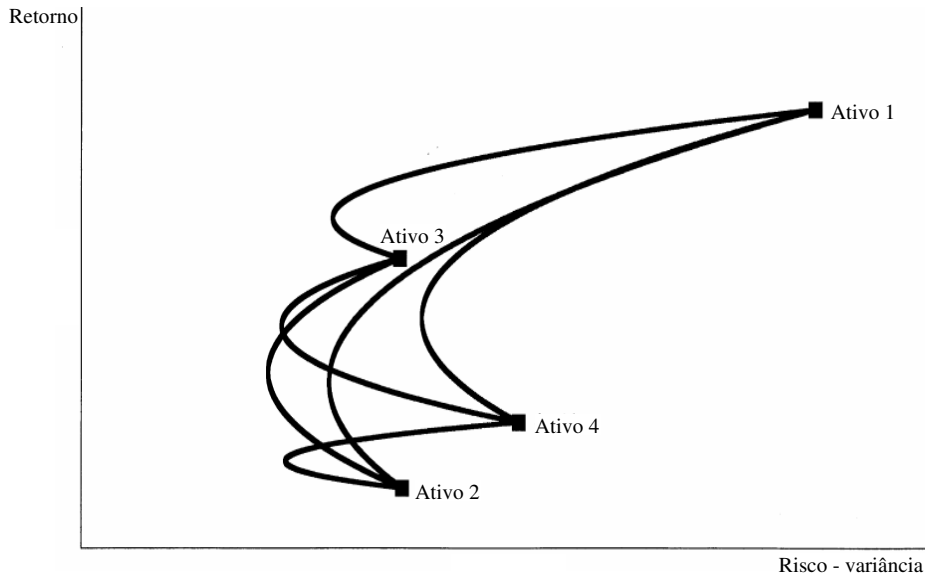


Figura 2 – Exemplo com combinação factível de dois ativos.

A fronteira eficiente representada na Figura 3 é uma curva descontínua, pois os *portfolios* que faltam à curva para torná-la contínua são dominados.

Figura 3 – Exemplo de fronteira eficiente para o modelo restrito.

1.6 OS PONTOS INVISÍVEIS

A resolução do modelo parametrizado restrito (1.8) para vários valores de λ não descreve completamente a fronteira eficiente de investimento. A justificativa para este fato pode ser obtida a partir de uma análise detalhada da função objetivo deste modelo.

Consideremos duas soluções sobre a fronteira eficiente de investimento de tal modo que as duas retas tangentes que passam por estas soluções tenham o mesmo coeficiente angular. A existência destes dois tipos de soluções está relacionada com a presença de dois segmentos, provenientes de curvas diferentes, com a mesma taxa de crescimento nos pontos em questão (vide Figura 4).

A ocorrência de pontos com as mesmas taxas de crescimento decorre do fato da curva ser descontínua, embora não decrescente. A descontinuidade ocorre porque o número de ativos pertencentes à carteira é menor que o número de ativos disponíveis para investimento, um fato muito comum em situações práticas, já que o mercado é composto por muitos ativos e o investidor tem interesse em formar carteiras que englobam uma pequena parte do universo de ativos.

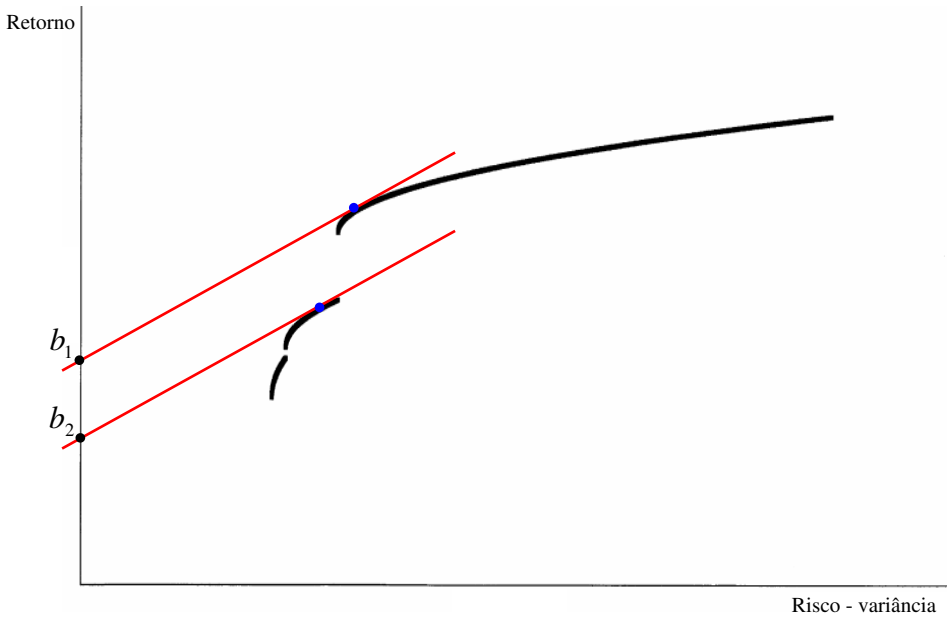


Figura 4 – Fronteira eficiente para o modelo restrito com duas soluções selecionadas.

Sejam Γ_1 e Γ_2 os valores da função objetivo do modelo (1.8) para estas duas soluções selecionadas, conforme a Figura 4. Esses valores podem ser escritos como combinação do risco e do retorno usando-se

$$\Gamma_1 = \lambda_1 [RIS_1] - (1 - \lambda_1)[RET_1]$$

e

$$\Gamma_2 = \lambda_2 [RIS_2] - (1 - \lambda_2)[RET_2].$$

Na fronteira eficiente, o eixo das ordenadas representa o retorno e o das abscissas o risco. Podemos reescrever as equações de modo que Γ_1 e Γ_2 fiquem fixos e os valores do retorno variem em função do risco. Assim, obtemos

$$[RET_1] = \frac{\lambda_1}{1-\lambda_1}[RIS_1] - \frac{1}{1-\lambda_1}\Gamma_1 \tag{1.9}$$

e

$$[RET_2] = \frac{\lambda_2}{1-\lambda_2}[RIS_2] - \frac{1}{1-\lambda_2}\Gamma_2 \, . \tag{1.10}$$

Observa-se que $\frac{\lambda_1}{1-\lambda_1}$ e $-\frac{1}{1-\lambda_1}\Gamma_1$ são, respectivamente, o coeficiente angular e o coeficiente linear da reta dada na equação (1.9), enquanto $\frac{\lambda_2}{1-\lambda_2}$ e $-\frac{1}{1-\lambda_2}\Gamma_2$ são, respectivamente, o coeficiente angular e o coeficiente linear da reta definida pela equação (1.10). Veja a Figura 5 abaixo.

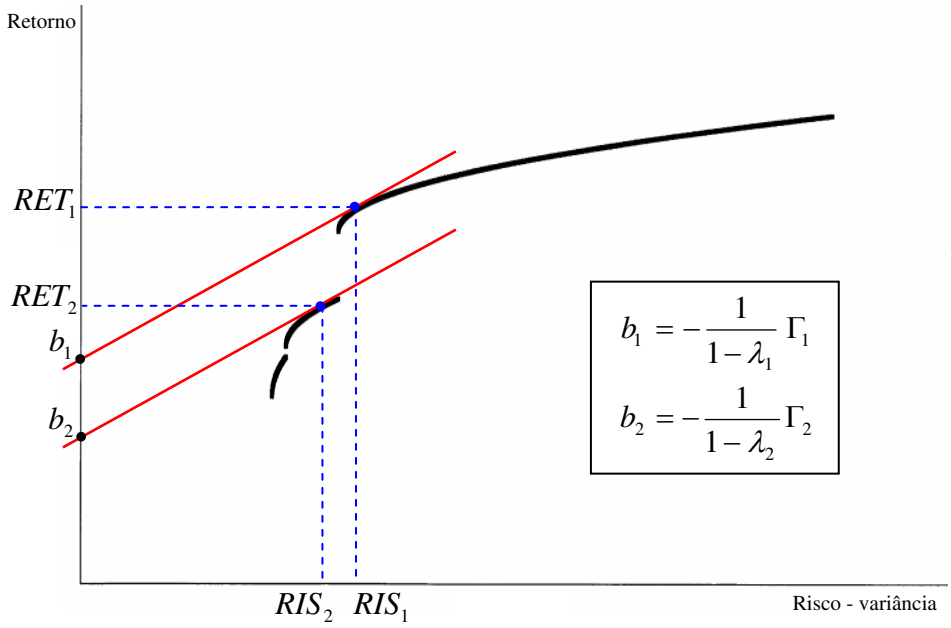


Figura 5 – Retas que tangenciam duas soluções da fronteira eficiente do modelo restrito.

Como as retas são paralelas, $\frac{\lambda_1}{1-\lambda_1} = \frac{\lambda_2}{1-\lambda_2}$, de modo que $\lambda_1 = \lambda_2$. Além disso, sabendo que $-\frac{1}{1-\lambda_1}\Gamma_1 > -\frac{1}{1-\lambda_2}\Gamma_2$, a reta da equação (1.9) estará sempre acima da reta associada à equação (1.10). Logo, $\Gamma_1 < \Gamma_2$, ou seja, a solução gerada pela equação (1.9) tem função objetivo de menor valor.

Portanto, como Γ_1 e Γ_2 são geradas pelo mesmo parâmetro λ, a segunda solução torna-se invisível. Ou seja, o algoritmo de otimização não irá considerar Γ_2 como solução ótima. Assim, a solução do modelo parametrizado geraria uma fronteira eficiente incompleta, como aquela mostrada na Figura 6.

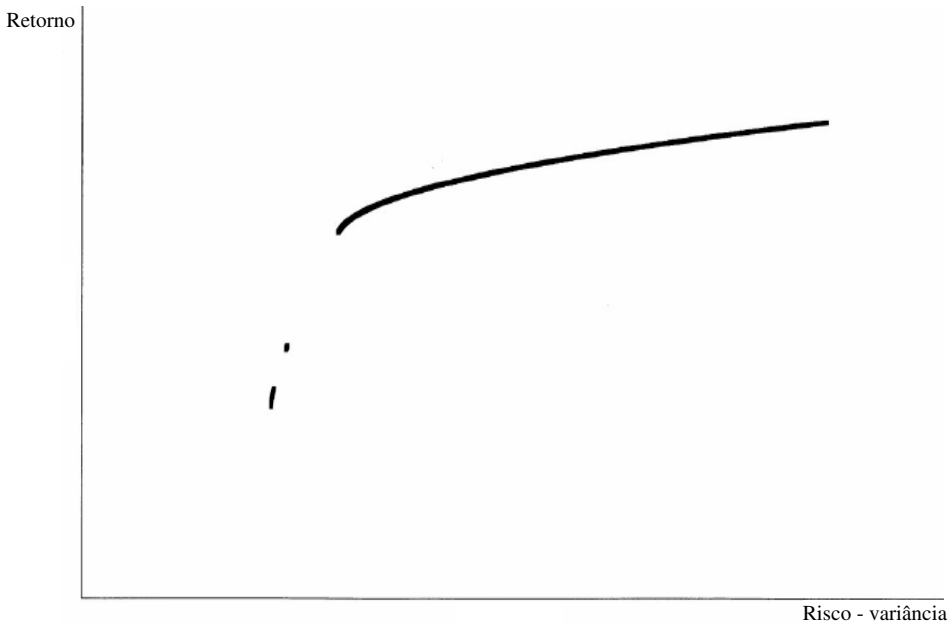


Figura 6 – Fronteira eficiente somente com os pontos visíveis.

A Figura 7 mostra os pontos que não poderiam ser encontrados pela simples otimização do modelo parametrizado para vários valores de λ.

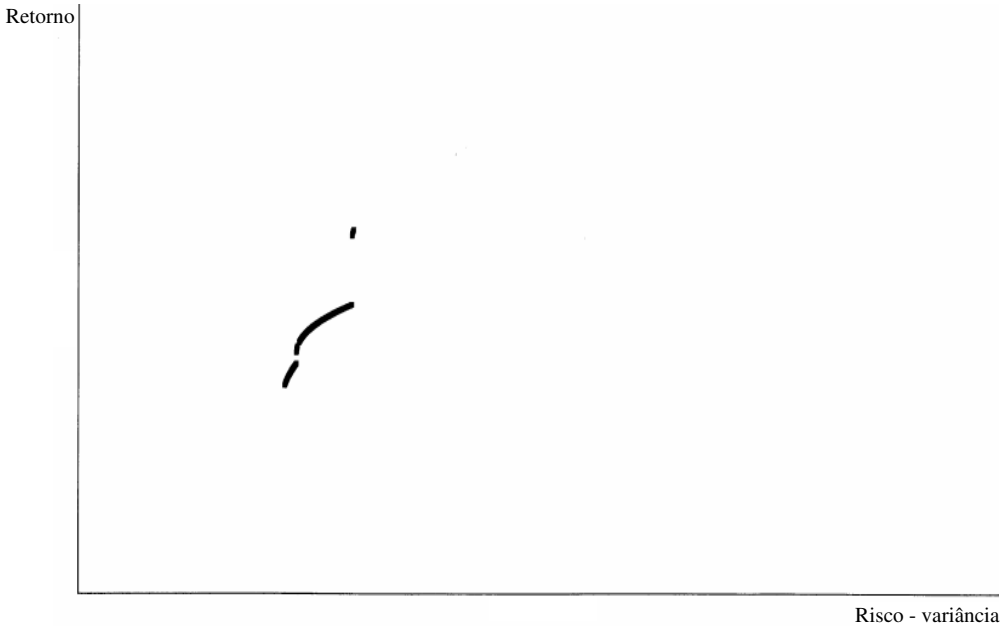


Figura 7 – Fronteira eficiente somente com os pontos invisíveis.

Como se observa, apesar do valor da função objetivo ser pior nos pontos invisíveis, estes não são pontos dominados, ou seja, não existe nenhum outro investimento que tenha, ao mesmo tempo, menor risco e maior retorno. Assim, a resolução do modelo parametrizado a partir de métodos exatos pode gerar uma fronteira eficiente incompleta.

Uma solução para evitar o problema da geração da fronteira eficiente incompleta é a utilização de algoritmos meta-heurísticos na resolução do modelo (1.8). Nos capítulos 2 e 5 serão apresentadas propostas de algoritmos para a resolução deste problema.

CAPÍTULO 2

PROPOSTA DE CHANG PARA A DETERMINAÇÃO DA F.E.I.

2.1 INTRODUÇÃO

Para a construção da fronteira eficiente restrita, o artigo de Chang *et al.* [3] propõe a utilização de três métodos heurísticos: algoritmos genéticos, busca tabu e têmpera simulada. O artigo sugere a utilização das heurísticas argumentando que o modelo restrito é um problema de programação quadrática composto por variáveis reais e binárias, o que torna sua resolução difícil por métodos exatos. Mitra *et al.* [15] classificam este problema como NP-Difícil, o que confirma a grande dificuldade em resolver esta classe de problemas com métodos exatos.

Chang *et al.* emprega a formulação parametrizada do modelo (1.8) para a determinação da F.E.I. A justificativa é que, nesse modelo, o retorno deixa de constituir uma restrição para fazer parte da função objetivo, tornando mais fácil a resolução do problema por uma heurística. Apesar do modelo parametrizado gerar pontos invisíveis, a utilização de um

algoritmo heurístico implica na investigação de muitas soluções, favorecendo, conseqüentemente, a localização de vários desses pontos invisíveis.

Neste trabalho, concentramos-nos nos algoritmos genéticos, pois os resultados obtidos por Chang *et al.* com esta heurística foram superiores.

2.2 O ALGORITMO GENÉTICO

O algoritmo genético (AG) é um método de busca estocástico baseado na teoria da evolução natural de Darwin [4], segundo a qual os organismos de uma população que adaptam-se melhor ao meio em que vivem têm mais chances de sobreviver e reproduzir, enquanto os indivíduos menos adaptados geralmente são eliminados. Esta combinação de boas características de indivíduos produz descendentes cada vez mais adaptados ao meio.

Os fundamentos teóricos do AG foram originalmente desenvolvidos por Holland [7]. Um AG simula o processo de evolução natural a partir de uma população de soluções iniciais, através da aplicação de operadores genéticos de reprodução e mutação nesses indivíduos. Em termos de otimização, cada indivíduo da população é uma solução codificada, composta por um conjunto de genes, a que chamamos de cromossomo. Assim, cada cromossomo representa uma possível solução para um dado problema.

A aptidão (ou *fitness*) do indivíduo é calculada com base em uma dada função objetivo. Indivíduos com boa aptidão terão mais chance de serem selecionados para reprodução (geralmente chamada de *crossover*, ou cruzamento). Nesse processo de reprodução, selecionam-se dois cromossomos (dois pais) e faz-se a combinação de seus genes, obtendo-se cromossomos filhos[3] que possuem uma fração cromossômica de cada um dos pais. Outra alteração nos cromossomos entre gerações é a mutação, em que se alteram alguns bits (ou genes) de um determinado cromossomo aleatoriamente.

[3] Em algoritmos genéticos clássicos o crossover gera 2 filhos.

Os passos básicos de um AG são descritos abaixo. Uma descrição mais detalhada pode ser encontrada em [1] e [14].

Algoritmo 1 (Algoritmo Genético padrão).

1. *Gerar uma população inicial.*

2. *Calcular a aptidão dos indivíduos da população inicial*.

3. *Repetir:*

3.1. *Selecionar pais da população para Crossover.*

3.2. *Realizar o Crossover.*

3.3. *Realizar Mutação na população.*

3.4. *Calcular a aptidão dos filhos e dos indivíduos que passaram pela mutação.*

3.5. *Selecionar os membros da população que formarão a próxima geração.*

4. até *atingir o número máximo de iterações.*

2.3 O ALGORITMO GENÉTICO PROPOSTO POR CHANG *ET ALII*

Nesta seção, apresentaremos o algoritmo genético proposto por Chang *et al.* para a determinação da F.E.I. do modelo restrito. Como veremos, nesta formulação, alguns procedimentos clássicos dos AG foram definidos de forma bastante peculiar. Assim, por exemplo, há apenas um *crossover* e uma mutação por iteração. Outras modificações foram empregadas para adequar o método ao modelo de otimização, já que trabalhamos com a representação de soluções em duas partes, uma inteira e outra real.

2.3.1 Representação da solução.

No algoritmo genético utilizado por Chang *et. al.*, a representação da solução tem duas partes distintas: o conjunto Q, composto pelos K ativos selecionados, e o conjunto S dos K números reais $s_i \left(0 \le s_i \le 1,\, i \in Q\right)$ que fornecem os valores de investimento. A Figura 8 fornece uma representação pictórica do cromossomo.

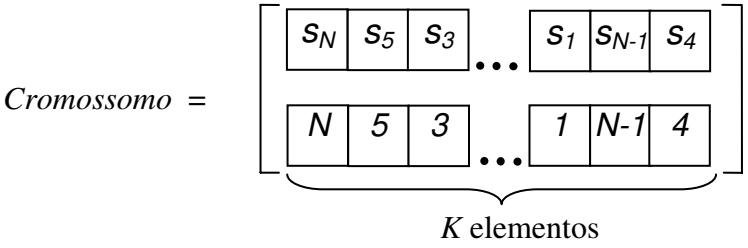


Figura 8 – Representação do cromossomo.

O valor de investimento s_i determina, através de uma mudança de variável, a proporção de investimento w_i no ativo i. Naturalmente, esta mudança somente é realizada nos ativos escolhidos para investimento ($i \in Q$). Os valores w_i são obtidos da seguinte forma:

$$w_i = \varepsilon_i + s_i \frac{F}{L}\,, \tag{2.1}$$

onde $F = 1 - \sum_{i \in Q} \varepsilon_i$ é a fração livre para investimento, e $L = \sum_{i \in Q} s_i$.

Para tornar clara esta representação, tomemos o exemplo em que N = 10, K = 2 e $\varepsilon_i = 0{,}1\,,\ \forall i$. Suponhamos que a solução (ou o cromossomo) seja dada por $Q = \{3\,,\,7\}$ e por $S = \{s_3 = 0{,}9\,,\ s_7 = 0{,}5\}$. Assim, $F = 1 - \sum_{i \in \{3,7\}} \varepsilon_i = 0{,}8$ e $L = \sum_{i \in \{3,7\}} s_i = 1{,}4$. Então, a proporção de investimento dos ativos na carteira é

$$w_3 = 0{,}1 + 0{,}9\left(0{,}8/1{,}4\right) = 0{,}6143$$

e

$$w_7 = 0{,}1 + 0{,}5\left(0{,}8/1{,}4\right) = 0{,}3857\,.$$

Podemos notar que esses valores satisfazem à restrição de limite inferior e de soma igual a 1.

O algoritmo de Chang *et al.* permite também a definição de limites superiores para as frações de investimento. Ou seja, ele contempla restrições do tipo

$$w_i \le \delta_i z_i, \quad i = 1, \ldots, N,$$

onde δ_i é a máxima proporção que pode ser investida no ativo *i*. Caso um valor w_i viole esta restrição, o algoritmo utiliza um procedimento iterativo para tornar factível a solução. A idéia básica deste procedimento é a seguinte: se existir $j \in Q$ tal que $w_j > \delta_j$, então w_j receberá o valor δ_j e as outras frações de investimentos serão recalculadas utilizando-se $F = 1 - (\sum_{i \in Q - \{j\}} \varepsilon_i + \delta_j)$ e $L = \sum_{i \in Q - \{j\}} s_i$. Este procedimento é repetido até que $w_i \le \delta_i, \forall i \in Q$.

2.3.2 Geração da população inicial.

Para gerar um cromossomo da população inicial, constrói-se dois vetores. O primeiro, denominado *s*, é composto por números aleatórios pertencentes a $(0\,,1]$. O segundo vetor, chamado de Q, é composto por números aleatórios inteiros não repetidos entre 1 e *N*. Os dois vetores têm comprimento igual a *K* (tamanho da carteira). Assim, o primeiro vetor, após a mudança de variável, corresponderá às proporções de investimento e o segundo vetor corresponderá aos ativos nos quais se irá investir.

2.3.3 *Crossover*.

Chang *et al.* utilizam o *crossover* do tipo uniforme para geração de novos indivíduos. Neste tipo de *crossover*, dois pais geram apenas um filho. Assim, se um ativo *i* está presente em ambos os pais, o filho herdará este ativo (o valor de investimento será escolhido aleatoriamente de um dos pais). Já para os ativos que estão presentes em somente um dos cromossomos, faz se um sorteio no qual cada ativo tem 50% de probabilidade de ser inserido no cromossomo filho. Um exemplo é dado na Figura 9.

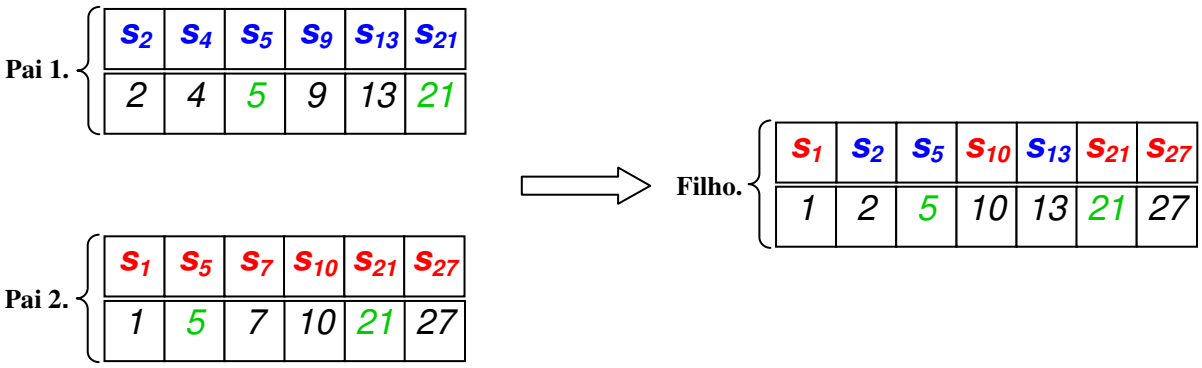


Figura 9 – Exemplo de *crossover* segundo a estratégia de Chang *et al.*

No exemplo acima, os ativos 5 e 21 foram incluídos no filho por estarem presentes em ambos os pais. O valor de investimento do ativo 5 foi herdado do Pai 1, enquanto s_{21} foi herdado do Pai 2.

Como os ativos que aparecem em apenas um pai são tratados individualmente, o número de ativos no cromossomo filho pode ser inferior ou superior ao tamanho da carteira, como aconteceu no exemplo da Figura 9. O algoritmo de Chang *et al.* considera esta possibilidade. Se o filho possuir mais ativos que o permitido, excluem-se os ativos com menor valor de investimento, de forma que o número de ativos no cromossomo atinja o valor correto. Por outro lado, se o filho possuir menos elementos que o desejado, insere-se no cromossomo outros ativos que não foram escolhidos no *crossover*, até que seja atingido o número correto.

A seleção dos indivíduos para *crossover* é feita por uma estratégia que Chang *et al.* chamam de torneio binário. Nesta estratégia, dois grupos são formados aleatoriamente a partir da população e o indivíduo com melhor *fitness* de cada grupo é selecionado. Deste modo, apenas dois cromossomos, dentre toda a população, são selecionados para o *crossover*, que ocorre apenas uma vez a cada iteração.

2.3.4 Mutação.

A mutação ocorre somente no filho gerado pelo *crossover* e corresponde a um decréscimo ou aumento de s_i. O único ativo alterado é escolhido aleatoriamente, tendo todos os ativos igual probabilidade de serem selecionados. Uma vez determinado o ativo, soma-se a s_i o valor $0,1(\varepsilon_i + s_i)$ ou $-0,1(\varepsilon_i + s_i)$. As probabilidades de ocorrência da soma e da subtração também são iguais. Um exemplo da mutação é dado na Figura 10.

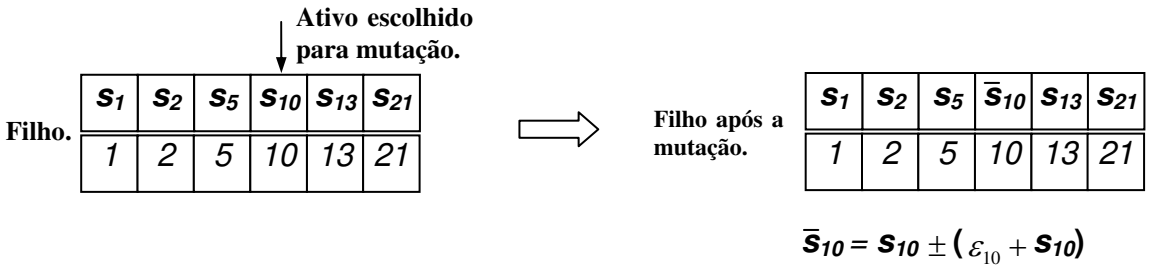


Figura 10 – Exemplo de mutação na estratégia de Chang *et al*.

2.3.5 Seleção.

Na seleção de novos indivíduos para a população, Chang *et al.* utilizam uma estratégia simples de inserção de indivíduos. Cada filho gerado é inserido na população, excluindo-se, em contrapartida, o pior cromossomo da população original.

2.3.6 Aptidão.

Para selecionar cromossomos para o *crossover*, bem como para eliminar cromossomos da população, é necessário definir um critério que meça a aptidão de cada indivíduo. Esta aptidão é representada pelo inverso aditivo do valor da função objetivo do modelo (1.8), ou seja,

$$Aptidão = -\left(\lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1-\lambda) \sum_{i=1}^{N} w_i \mu_i \right). \tag{2.2}$$

Assim, indivíduos mais aptos possuem função objetivo menor que os demais.

2.3.7 Algoritmo genético proposto por Chang *et alii*.

Os procedimentos básicos do algoritmo genético proposto por Chang *et al.* estão resumidos a seguir.

Algoritmo 2 (Algoritmo genético proposto por Chang *et al.*).

1. *Gerar uma população inicial com 100 indivíduos.*

2. *Calcular a aptidão dos indivíduos da população inicial.*

3. ***Em cada iteração, fazer:***

3.1. *Selecionar 2 pais da população para Crossover.*

3.2. *Realizar o Crossover.*

3.4. *Realizar Mutação no filho gerado pelo crossover.*

3.5. *Calcular a aptidão do filho.*

3.6. *Substituir o pior membro da população pelo filho.*

4. até *atingir o número máximo de iterações.*

Para gerar toda a F.E.I., o algoritmo é repetido para vários valores de λ pertencentes ao intervalo $[0,1]$. Como o algoritmo de Chang *et al.* realiza apenas um *crossover* e uma mutação a cada iteração, o número de iterações deve ser elevado. Em [3], recomenda-se parar o algoritmo após $T^* = 1000N$ iterações, em que N é o número de ativos disponíveis para investimento.

CAPÍTULO 3

ESPAÇO DE SOLUÇÕES SIMPLEX

3.1 INTRODUÇÃO

Uma solução factível para o modelo parametrizado (1.5) é composta pelos ativos selecionados para investimento e pelas frações de investimento w_i destes ativos. A soma das frações de investimento destes ativos é obrigatoriamente igual a 1. Assim, os possíveis valores de frações de investimento para uma carteira que contém n ativos formam um subconjunto de $\Re^n$ que é conhecido como simplex.

3.2 DEFINIÇÃO DE SIMPLEX

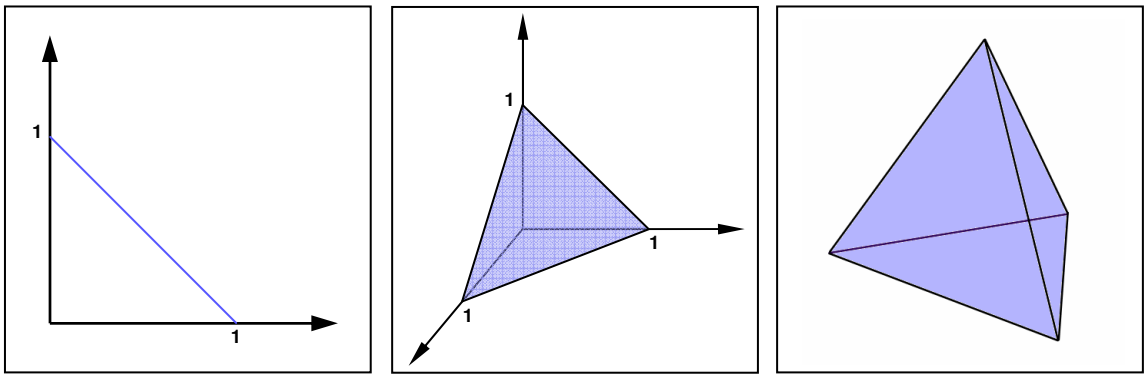
O simplex padrão de dimensão n-1 é definido como o subconjunto de $\Re^n$ dado por

$$\Delta^{n-1} = \left\{ (x_1, \ldots, x_n) \in \Re^n \,\middle|\, \sum_i x_i = 1 \;\; e \;\; x_i \geq 0 \;\; p/ \; \forall i \right\}.$$

O simplex Δ^{n-1} mora em um hiperplano definido pela restrição $\sum_{i=1}^{n} x_i = 1$. O simplex padrão é claramente regular. Os vértices do simplex padrão (n-1)-dimensional são os pontos

$$\begin{aligned} e_1 &= (1,0,0,\ldots,0) \\ e_2 &= (0,1,0,\ldots,0) \\ &\vdots \\ e_n &= (0,0,0,\ldots,1). \end{aligned} \tag{3.1}$$

É importante salientar que a dimensão deste simplex é sempre uma unidade a menos que a dimensão do espaço no qual ele se encontra. A Figura 11 ilustra os simplex que estão em $\Re^2$, $\Re^3$ e $\Re^4$, respectivamente:



Simplex de dimensão 1. Simplex de dimensão 2. Simplex de dimensão 3.

Figura 11 – Exemplos de simplex.

3.3 PROPRIEDADES GEOMÉTRICAS

O conteúdo[4] de um simplex regular (i.e. um simplex que possui arestas com mesmo comprimento) de tamanho s é dado por

[4] Conteúdo é a generalização de volume para sólidos n-dimensionais.

$$V_{n-1} = \frac{s^{n-1}}{(n-1)!}\sqrt{\frac{n}{2^{n-1}}} \ . \tag{3.2}$$

Doravante, trabalharemos apenas com simplex de dimensões regulares.

O conteúdo de um simplex (n-1)-dimensional com vértices descritos como em (3.1), cujas arestas têm tamanho igual a $\sqrt{2}$, é dado por

$$V_{n-1} = \frac{\sqrt{n}}{(n-1)!}. \tag{3.3}$$

O conteúdo de um simplex regular também pode ser calculado em função do conteúdo de um outro simplex regular, bastando para isso que se conheça a relação que existe entre as arestas destes dois simplex. Tomemos, por exemplo, um simplex de conteúdo V_{n-1}, com arestas de comprimento s. O conteúdo $\overline{V}_{n-1}$ de um outro simplex com arestas de comprimento $\overline{s} = c\,s$, em que $c \in \Re^+$, é dado por

$$\overline{V}_{n-1} = \frac{(c\,s)^{n-1}}{(n-1)!}\sqrt{\frac{n}{2^{n-1}}} = c^{n-1}\frac{s^{n-1}}{(n-1)!}\sqrt{\frac{n}{2^{n-1}}} = c^{n-1}V_{n-1}. \tag{3.4}$$

O simplex de dimensão n-1 com arestas de tamanho s e com altura H_{n-1} é formado pela sobreposição de um número infinito de simplex de dimensão n-2, com arestas que são menores ou iguais a s.

O comprimento das arestas dos simplex de dimensão n-2 mostrado na Figura 12 varia continuamente com a variação da altura do simplex de dimensão n-1. Assim, para cada valor da "altura" x (medida a partir do vértice superior do simplex de dimensão n-1), determinamos a aresta do simplex posicionado nesta altura através da fórmula

$$\overline{s} = \frac{x}{H_{n-1}}s \ .$$

Substituindo essa expressão na equação (3.4), obtemos o conteúdo do simplex de dimensão n-1, que é dado por

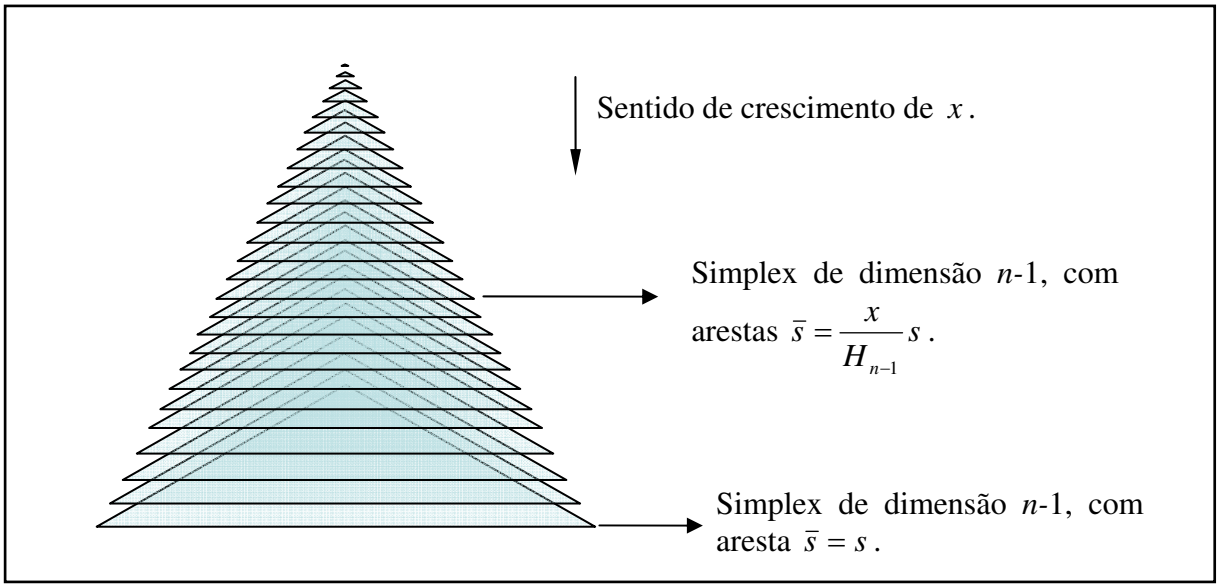


Figura 12 – Simplex de dimensão n-1 formado pela sobreposição de simplex de dimensão n-2.

$$V_{n-1} = \int_0^{H_{n-1}} \left(\frac{x}{H_{n-1}} \right)^{n-2} V_{n-2}\, dx = \frac{x^{n-1}}{n-1} \frac{V_{n-2}}{\left(H_{n-1}\right)^{n-2}} \Bigg|_0^{H_{n-1}} = \frac{H_{n-1}}{n-1} V_{n-2}.$$

onde V_{n-2} pode ser determinado pela fórmula (3.3).

Isolando a altura H_{n-1} na fórmula de V_{n-1}, chegamos a

$$H_{n-1} = (n-1)\frac{V_{n-1}}{V_{n-2}} = s\sqrt{\frac{n}{2(n-1)}}. \tag{3.5}$$

A equação (3.5) fornece a altura de um simplex de dimensão n-1 com arestas de comprimento s. No caso do simplex padrão, com vértices descritos em (3.1) e arestas de tamanho $\sqrt{2}$, a altura é igual a

$$H_{n-1} = \sqrt{\frac{n}{n-1}}. \tag{3.6}$$

3.4 A DISTRIBUIÇÃO NÃO UNIFORME NO ESPAÇO DE SOLUÇÕES SIMPLEX

No algoritmo genético proposto por Chang *et al.* para obtenção das frações de investimento w_i, utiliza-se a mudança de variável (2.1), reproduzida aqui para facilitar a exposição:

$$w_i = \varepsilon_i + s_i \frac{F}{L}, \forall i \in Q,$$

onde $L = \sum_{i \in Q} s_i$ e $F = 1 - \sum_{i \in Q} \varepsilon_i$.

Esse procedimento faz com que um conjunto de frações de investimentos tenha mais chance de ocorrer que outros. Como exemplo, vamos simular a geração da população inicial do algoritmo de Chang *et al.* para um problema na forma (1.8), com três ativos. Para ilustrar melhor a distribuição dos indivíduos, vamos considerar uma população composta por 15000 cromossomos. Como temos apenas três ativos, podemos associar, a cada indivíduo, um ponto em $\Re^3$, de tal forma que as coordenadas do ponto representem as frações de investimento. Após a mudança de variável, a soma dos valores w_i de cada cromossomo é igual a 1. Obtemos, assim, pontos que pertencem a um simplex bidimensional, conforme mostrado na Figura 13.

Observa-se, nesta figura, que há uma grande concentração de pontos no centro do simplex. Para compreender a origem dessa concentração, é preciso lembrar que, ao construirmos a população inicial, geramos pontos uniformemente distribuídos dentro de um hipercubo (ou caixa, como no exemplo). Em seguida, fazemos uma mudança de variável, transladando todos os pontos que estão dentro do hipercubo de dimensão *n* para um simplex de dimensão *n*-1. Essa mudança de variável leva mais pontos para o centro do simplex do que para suas extremidades. A Figura 14 ilustra esse problema.

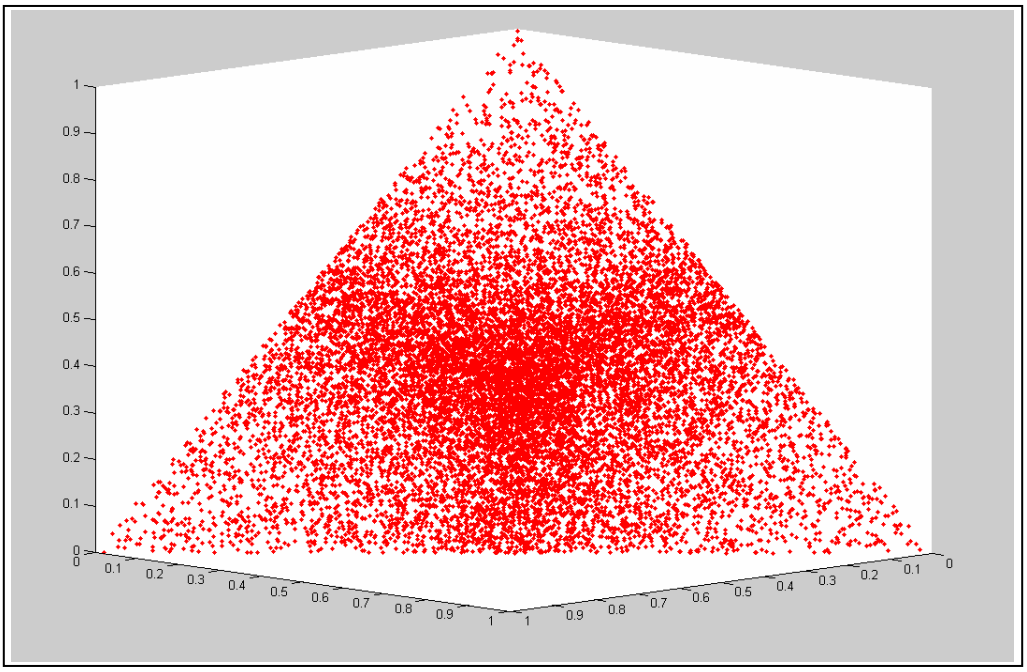


Figura 13 – Exemplo da distribuição não uniforme da população inicial.

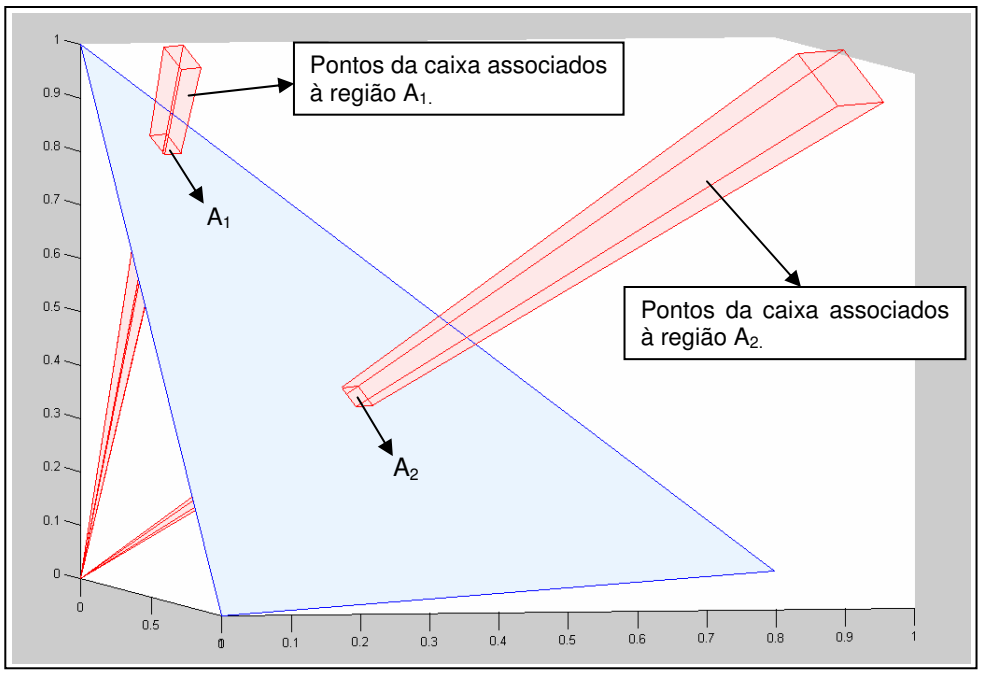


Figura 14 – O problema da concentração de soluções em regiões do simplex.

Na Figura 14, apesar da área da região A_2 ser igual à área da região A_1 existe um número maior de pontos da caixa associados a A_2. Como o método de Chang *et al.* gera pontos uniformemente distribuídos na caixa, a concentração de pontos na região A_2 será maior, fazendo com que situações iguais àquela mostrada na Figura 13 ocorram. Obviamente, esta má distribuição vale para qualquer espaço *n*-dimensional.

No próximo capítulo, apresentaremos uma proposta de eliminação dessa distribuição não uniforme da população.

CAPÍTULO 4

DISCRETIZAÇÃO DO ESPAÇO DE SOLUÇÕES

4.1 INTRODUÇÃO

Uma maneira de evitar o problema de distribuição não uniforme das soluções geradas pelo algoritmo de Chang *et al.* seria gerar pontos diretamente dentro do simplex, em lugar de projetá-los.

Entretanto, não é fácil criar números aleatórios com distribuição uniforme que satisfaçam as restrições $\sum_i x_i = 1$ e $x_i \geq 0, \forall i$. Para contornar essa dificuldade, podemos discretizar a região factível, associando a cada ponto do simplex discretizado um número inteiro único que o represente. Deste modo, somos capazes gerar números aleatórios com distribuição uniforme sem nos preocuparmos com a factibilidade.

Neste capítulo, apresentamos essa nova estratégia de representação dos pontos e discutimos como obter as coordenadas de um ponto a partir do número inteiro a ele associado.

4.2 O MÉTODO UTILIZADO PARA A DISCRETIZAÇÃO DO ESPAÇO

Suponhamos que um computador gaste d bits para armazenar um número real. Neste caso, o armazenamento das coordenadas de um ponto sobre um simplex em $\Re^n$ consumiria dn bits. Como se sabe, a precisão do computador é limitada, de modo que não seria possível representar, na prática, todos os infinitos pontos de um simplex. Esse inconveniente, entretanto, não costuma ser relevante, uma vez que o número de bits usados para armazenar números reais costuma ser grande.

Pretendemos usar o mesmo número de bits para armazenar pontos de um simplex regular, empregando uma estratégia de representação baseada na associação de um número inteiro a cada ponto. Para tanto, vamos supor que o simplex S^{n-1} (um simplex de dimensão n–1 em $\Re^n$) seja dividido em $\eta = 2^{(n-1)d}$ hipercubos, de forma que cada um destes esteja associado a uma pequena região de S^{n-1}, representada pelo centro do hipercubo. Assim, o $\hat{\eta}^{\text{ésimo}}$ hipercubo estará relacionado ao ponto

$$x^{\hat{\eta}} = \left(x_1^{\hat{\eta}}, x_2^{\hat{\eta}}, \ldots, x_n^{\hat{\eta}}\right),$$

onde $x_i^{\hat{\eta}}$ é a $i^{\text{ésima}}$ coordenada de seu centro.

Naturalmente, a aproximação de um simplex por um conjunto de hipercubos não é perfeita, pois é possível que um hipercubo possua pontos externos ao simplex. Entretanto, essa imprecisão perde relevância à medida que refinamos a discretização, aumentando o número de hipercubos. Se utilizarmos, por exemplo, $2^{(n-1)d}$ bits para indicar a posição do hipercubo dentro do simplex (n–1)-dimensional, teremos uma precisão comparável àquela obtida utilizando-se d bits para armazenar cada coordenada do ponto no sistema cartesiano.

Doravante, vamos supor que o número de hipercubos utilizados é grande o suficiente para que a aproximação do simplex seja aceitável. Neste caso, igualando o volume total dos

hipercubos a V_{n-1}, o conteúdo do simplex de dimensão n–1, dado pela equação (3.3), obtemos a seguinte expressão para o comprimento das arestas do hipercubo:

$$a = \left(\frac{V_{n-1}}{2^{(n-1)d}} \right)^{\frac{1}{(n-1)}}. \tag{4.1}$$

4.2.1 A divisão dos simplex em camadas.

Em nossa aproximação, dispomos os η hipercubos do simplex de dimensão n–1 em camadas de "altura" a. A divisão em camadas é feita a partir de um vértice do simplex — doravante chamado *vértice de referência da camada*[5] — em direção à face oposta a esse vértice, conforme ilustrado na Figura 15.

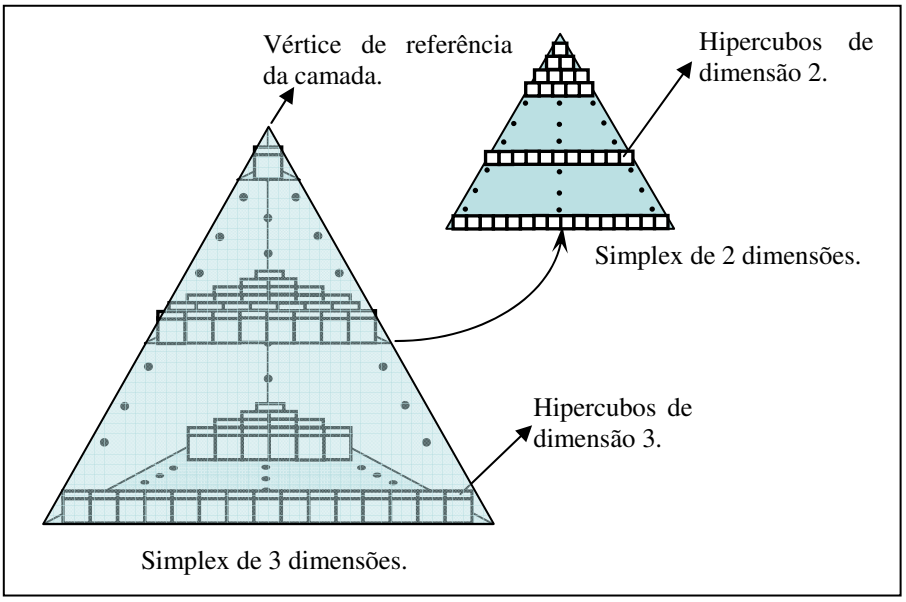


Figura 15 – Disposição dos hipercubos nos simplex de 3 e 2 dimensões.

[5] Definimos como o vértice de referência de um simplex regular aquele no qual a última coordenada (no sistema cartesiano) é diferente de zero. Assim, por exemplo, para um simplex bidimensional no espaço tridimensional, o vértice de referência será o ponto (0,0,1).

Cada camada está *apoiada*, ou *assentada*, sobre um simplex de dimensão *n*–2 que denominamos *base*. É esse simplex que determina o número de hipercubos que a camada contém. Assim, para saber quantos hipercubos possui a camada de índice *k*, cuja base está a uma distância *ka* do vértice de referência, contamos os hipercubos que podem ser apoiados sobre essa base. Deve-se observar que a base da última camada é um simplex regular de dimensão *n*–2 com arestas iguais a $\sqrt{2}$.

Para definir de forma única um hipercubo, é preciso, em primeiro lugar, identificar a camada à qual ele pertence. Em seguida, é preciso distinguí-lo dos demais hipercubos da camada. Isso é feito eliminando-se a dimensão do hipercubo já identificada pela camada e reduzindo-se o problema ao simplex de dimensão *n*–2 que caracteriza essa camada. Para o simplex tridimensional ilustrado na Figura 15, as camadas estão associadas a simplex bidimensionais, como o que é mostrado em destaque.

Agindo de forma análoga àquela adotada para o simplex original, o simplex de dimensão *n*–2 também é dividido em camadas que partem de um vértice e são paralelas à face oposta a este. Feita essa divisão, podemos identificar a camada na qual o hipercubo está localizado, reduzindo, em seguida, o problema a um simplex de dimensão *n*–3. Esse procedimento é repetido recursivamente até que atinjamos o simplex de dimensão 1, como mostrado na Figura 16.

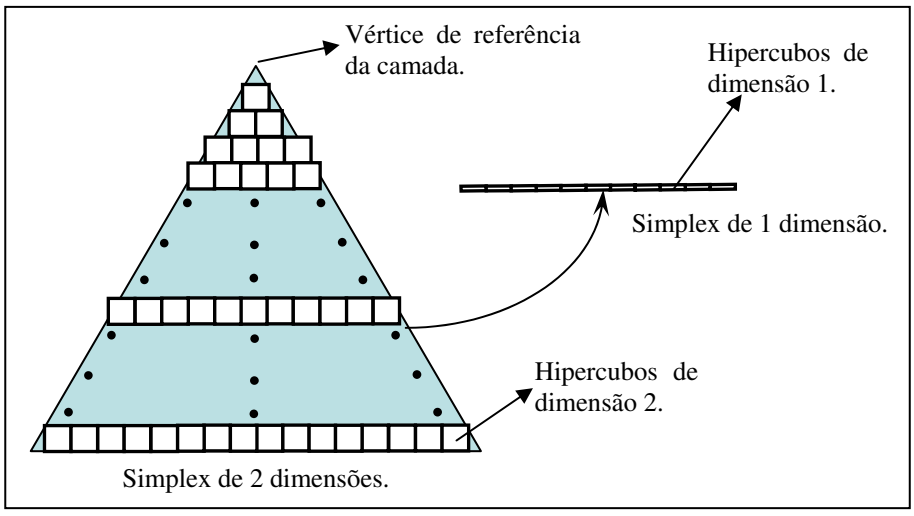


Figura 16 – Disposição dos hipercubos no simplex de 2 e 1 dimensão.

Obtemos, assim, os índices das n–1 camadas que identificam, de forma única, um hipercubo dentro do simplex (n–1)-dimensional. Essa notação é mais econômica que a adotada usualmente para representar os pontos de $\Re^n$ que pertencem ao simplex, uma vez que requer apenas n–1 coordenadas. Contudo, a principal vantagem dessa abordagem é o fato de que ela permite que associemos um número inteiro único a cada hipercubo, desde que sejamos capazes de relacioná-lo às camadas que o contêm. Felizmente, isso é possível, como veremos na próxima subseção.

4.2.2 A contagem das camadas.

Suponhamos que um simplex regular de dimensão n–1 tenha sido dividido em hipercubos com arestas iguais a a. Neste caso, o número de hipercubos usados para representar o simplex é dado aproximadamente por

$$\eta = \frac{V_{n-1}}{a^{n-1}}.$$

O número de hipercubos contidos na porção do simplex que vai da primeira até a camada de índice k, cuja base está a uma distância ka do vértice de referência, é definido por

$$\eta_{n-1}^{k} = \left(\frac{ka}{H_{n-1}}\right)^{n-1} \frac{V_{n-1}}{a^{n-1}} = k^{n-1} \frac{V_{n-1}}{\left(H_{n-1}\right)^{n-1}}, \tag{4.2}$$

onde H_{n-1} é a "altura" do simplex, ou seja, a distância entre o vértice de referência e a face do simplex, de dimensão n–2, oposta a este vértice.

Seja η_{n-1}^{k-1} o número de elementos até a camada k–1. Neste caso, o número de elementos da camada k é definido como

$$\eta_{n-1}^{k} - \eta_{n-1}^{k-1} = \left(k^{n-1} - (k-1)^{n-1} \right) \frac{V_{n-1}}{(H_{n-1})^{n-1}}. \tag{4.3}$$

Assim, o $\hat{\eta}^{\text{ésimo}}$ hipercubo do simplex (n–1)-dimensional estará na camada k do simplex se

$$(k-1)^{n-1} \frac{V_{n-1}}{(H_{n-1})^{n-1}} \le \hat{\eta} \le k^{n-1} \frac{V_{n-1}}{(H_{n-1})^{n-1}},$$

ou seja, quando

$$k - 1 < \sqrt[n-1]{\frac{\hat{\eta}}{V_{n-1}}}\, H_{n-1} \le k\ .$$

Logo, a camada do simplex (n–1)-dimensional à qual pertence o $\hat{\eta}^{\text{ésimo}}$ hipercubo é definida simplesmente por

$$k_{n-1} = \left\lceil \sqrt[n-1]{\frac{\hat{\eta}}{V_{n-1}}}\, H_{n-1} \right\rceil. \tag{4.4}$$

No algoritmo genético que descreveremos no próximo capítulo, os cromossomos estarão associados aos hipercubos nos quais dividimos o simplex. Cada hipercubo será representado por um único número inteiro $\hat{\eta} \in \{1,\ldots,\eta\}$ e a camada do simplex de dimensão n–1 que o contém será dada pela fórmula (4.4).

Uma vez determinada a camada do simplex (n–1)-dimensional, precisamos determinar a posição do hipercubo de índice $\hat{\eta}$ dentro desta camada. Para tanto, reduzimos o problema ao simplex de dimensão n–2 que forma a base da camada recém determinada.

Nosso objetivo, agora, passa a ser a determinação da camada, no simplex de dimensão n–2, do hipercubo de dimensão n–2 formado pela projeção do hipercubo (n–1)-dimensional sobre o novo simplex. Contudo, o volume e a altura desse simplex não são mais V_{n-2}

e H_{n-2} (dados pelas equações (3.3) e (3.6)), pois o simplex não é mais o padrão, ou seja, suas arestas têm comprimento menor que $\sqrt{2}$. Assim, empregamos as fórmulas

$$\overline{V}_{n-2} = \left(c_{n-2}\right)^{n-2} V_{n-2} \tag{4.5}$$

$$\overline{H}_{n-2} = c_{n-2} H_{n-2}\ , \tag{4.6}$$

onde c_{n-2} é um fator de escala definido como $k_{n-1} \big/ k_{\max_{n-1}}$, sendo k_{n-1} a camada do simplex (n–1)-dimensional determinada anteriormente e $k_{\max_{n-1}} = \lceil H_{n-1} / a \rceil$ o número de camadas nas quais o simplex foi dividido. Uma vez que, na prática, o número de hipercubos é muito grande (para garantir uma melhor discretização do simplex), temos

$$c_{n-2} \approx k_{n-1} a \big/ H_{n-1}\,. \tag{4.7}$$

Por simplicidade, usaremos esse valor aproximado em lugar da definição de c_{n-2}.

A partir da relação (4.3), obtemos o número de hipercubos do simplex de dimensão n–2, que é dado por

$$\overline{\eta}_{n-2} = \left(\left(k_{n-1}\right)^{n-1} - \left(k_{n-1} - 1\right)^{n-1} \right) \frac{\overline{V}_{n-1}}{\left(\overline{H}_{n-1}\right)^{n-1}}\,. \tag{4.8}$$

Para localizar um hipercubo na camada desejada, é preciso, antes de mais nada, atualizar $\hat{\eta}$, definindo o índice do hipercubo dentro do novo simplex. Esse índice é obtido pela diferença entre $\hat{\eta}$ e o número de hipercubos existentes até a camada k–1 do simplex (n–1)-dimensional. Ou seja,

$$\hat{\eta}_{n-2} = \hat{\eta} - \eta_{n-1}^{k-1}. \tag{4.9}$$

Temos, agora, um problema restrito ao simplex de dimensão n–2. Para achar a camada do hipercubo $\hat{\eta}_{n-2}$ neste simplex, usamos a fórmula

$$k_{n-2} = \left\lceil \sqrt[n-2]{\frac{\hat{\eta}_{n-2}}{\overline{V}_{n-2}}}\ \overline{H}_{n-2} \right\rceil,$$

que pode ser obtida de forma análoga a (4.4). Empregando, então, (4.5) e (4.6), obtemos, finalmente,

$$k_{n-2} = \left\lceil \sqrt[n-2]{\frac{\hat{\eta}_{n-2}}{V_{n-2}}}\ H_{n-2} \right\rceil. \tag{4.10}$$

De posse da camada do simplex de dimensão n–2 associada ao hipercubo de índice $\hat{\eta}$, repetimos o processo até a camada de dimensão 1.

4.2.3 A determinação das coordenadas de centro do hipercubo.

Veremos agora como determinar as coordenadas, em $\Re^n$, do centro de um hipercubo de índice $\hat{\eta}$, que pertence ao simplex definido pelas restrições $\sum_{i=1}^{n} x_i = 1$ e $x_i \geq 0$, i = 1, ..., n. Para tanto, vamos supor que as n–1 camadas associadas ao hipercubo foram obtidas conforme mostrado na subseção 4.2.2.

Iniciamos a determinação das coordenadas pela camada k_{n-1}, relativa ao simplex de dimensão n–1, cujos vértices correspondem aos pontos e_1, e_2, ..., e_n, como definido em (3.1). Esta camada fornecerá a $n^{\text{ésima}}$ coordenada de x.[6]

Na seção anterior, após discretizar o simplex em hipercubos de arestas a, determinamos D_{n-1}^{B}, a distância entre o vértice de referência e_n e a camada em que se encontra o

[6] De uma forma geral, a camada relacionada ao simplex de dimensão m corresponde à componente m+1 de x, como veremos adiante.

hipercubo $\hat{\eta}$ — aqui tratada como a "altura" da camada k_{n-1} — através da expressão $D_{n-1}^{B} = k_{n-1} a$. Esta distância, entretanto, corresponde à base da camada. Para encontrarmos a "altura" do centro do hipercubo, D_{n-1}^{*}, devemos usar a fórmula

$$D_{n-1}^{*} = (k_{n-1} - 1/2)\, a \tag{4.11}$$

A Figura 17 ilustra a correspondência entre D_{n-1}^{B} e D_{n-1}^{*}.

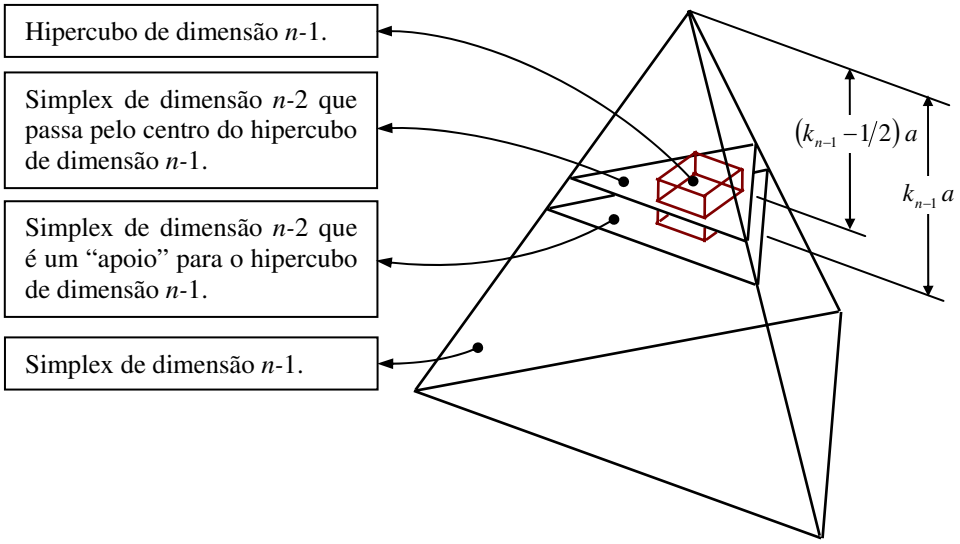


Figura 17 – Representação das camadas no simplex de dimensão n–1.

Os n vértices do simplex (n–1)-dimensional e a origem formam um novo simplex, $\overline{S}_n^{G}$, não regular, de dimensão n. Da mesma maneira, os vértices da camada de índice k_{n-1}, o ponto e_n e o ponto de coordenadas (0, 0, ..., $x_n^{\hat{\eta}}$) formam um simplex $\overline{S}_n^{P}$ semelhante a $\overline{S}_n^{G}$, como ilustrado na Figura 18.

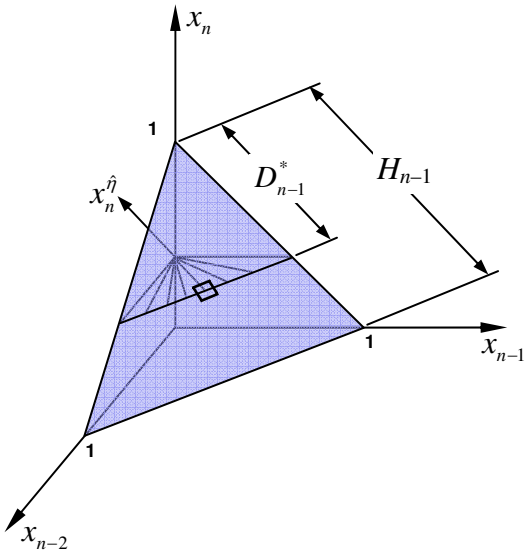


Figura 18 – Os simplex S_{n-1} (em destaque), $\overline{S}_n^G$ e $\overline{S}_n^P$.

Definamos como Δ_n o triângulo formado pela origem, por e_n e pelo ponto da base do simplex S_{n-1} que está mais próximo de e_n. Definamos, de forma análoga, o triângulo Δ_n^P, formado por e_n, pelo ponto (0, 0, ..., $x_n^{\hat{\eta}}$) e pelo ponto do simplex (n–2)-dimensional (que passa pelo centro do hipercubo de índice $\hat{\eta}$) que está mais próximo de e_n, como ilustrado na Figura 19.

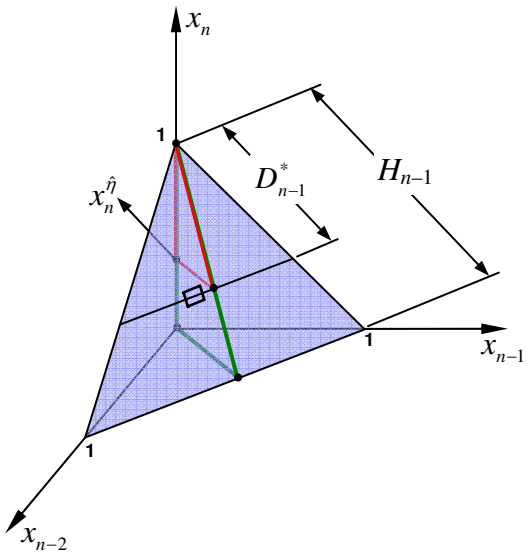


Figura 19 – Os triângulos Δ_n (em destaque verde) e Δ_n^P (em destaque vermelho).

Dada a semelhança entre os simplex, esses triângulos retângulos também são semelhantes, de modo que vale a relação

$$\frac{D^*_{n-1}}{H_{n-1}} = \frac{1 - x^{\hat{\eta}}_n}{1}.$$

Obtemos, assim, a $n^{\text{ésima}}$ coordenada do ponto $x^{\hat{\eta}}$, que é dada por

$$x^{\hat{\eta}}_n = 1 - \frac{D^*_{n-1}}{H_{n-1}} \tag{4.12}$$

Uma vez calculada a última coordenada de $x^{\hat{\eta}}$, restringimos nossa atenção ao simplex de dimensão n–2 definido pela camada k_{n-1}. Esse simplex, que denominamos $\overline{S}_{n-2}$, está contido no subespaço afim que passa pelo ponto (0, 0, ..., $x^{\hat{\eta}}_n$) e é paralelo ao hiperplano definido pela origem e pelos pontos e_1, e_2, ..., e_{n-1}. Assim, as primeiras n–2 coordenadas de $x^{\hat{\eta}}$ neste subespaço afim são iguais àquelas definidas em $\Re^n$, não sendo necessária qualquer conversão.

Passemos, então, à determinação da coordenada $x^{\hat{\eta}}_{n-1}$. A equação (4.9) indica a localização do hipercubo na camada k_{n-1}. Já a equação (4.10) fornece k_{n-2}, a camada na qual se encontra o hipercubo no simplex de dimensão n–2.

Apesar das arestas do simplex $\overline{S}_{n-2}$ terem mesmo comprimento, digamos s_{n-2}, elas não medem $\sqrt{2}$, como em S_{n-1}, de modo que as fórmulas usadas na obtenção de $x^{\hat{\eta}}_n$ não podem ser aplicadas diretamente no cálculo de $x^{\hat{\eta}}_{n-1}$. Para obter as novas fórmulas, vamos definir o simplex (n–1)-dimensional $\overline{S}^G_{n-1}$, formado pela origem e pelos vértices de $\overline{S}_{n-2}$, bem como o simplex $\overline{S}^P_{n-1}$, formado pelos vértices da camada de índice k_{n-2}, pelo ponto com coordenadas (0, 0, ..., $x^{\hat{\eta}}_{n-1}$) e pelo ponto $\overline{e}_{n-1}$, vértice de referência de $\overline{S}_{n-2}$. Estes dois simplex são mostrados na Figura 20.

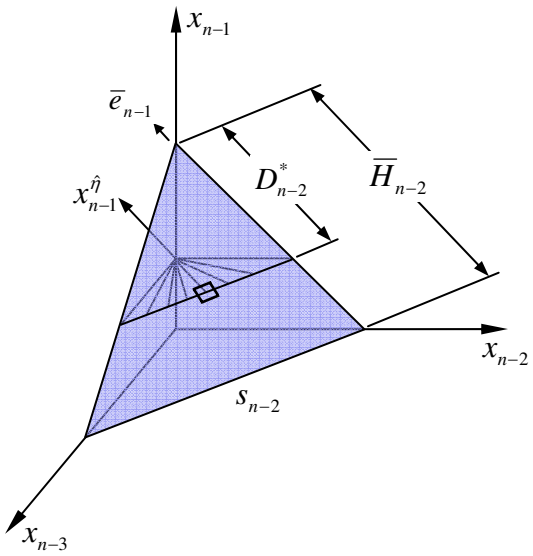


Figura 20 – Os simplex $\overline{S}_{n-2}$ (em destaque), $\overline{S}^G_{n-1}$ e $\overline{S}^P_{n-1}$.

Duas novas medidas, D^*_{n-2} e $\overline{H}_{n-2}$, foram introduzidas na Figura 20. D^*_{n-2} é a distância entre o nó de referência e a camada de índice k_{n-2}, e é dada pela fórmula

$$D^*_{n-2} = (k_{n-2} - 1/2)\, a, \tag{4.13}$$

que é análoga àquela usada para determinar $x^{\hat{\eta}}_n$. Por sua vez, o cálculo de $\overline{H}_{n-2}$, a "altura" do simplex $\overline{S}_{n-2}$, requer mais cuidado. Recorrendo às equações (4.6) e (4.7), obtemos a altura do simplex que serve de apoio para o hipercubo $\hat{\eta}$. Já para o simplex que passa pelo centro do hipercubo, a constante c_{n-2} precisa ser reduzida, de modo que definimos

$$\overline{c}_{n-2} = (k_{n-1} - 1/2)a / H_{n-1}$$

Assim, a partir de (4.6), temos

$$\overline{H}_{n-2} = \overline{c}_{n-2} H_{n-2} = \frac{(k_{n-1} - 1/2)a}{H_{n-1}} H_{n-2} = \frac{H_{n-2}}{H_{n-1}} D^*_{n-1} \tag{4.14}$$

De posse de D^*_{n-2} e $\overline{H}_{n-2}$, resta-nos apenas determinar as coordenadas do ponto $\overline{e}_{n-1}$ para que sejamos capazes de calcular $x^{\hat{\eta}}_{n-1}$. As primeiras *n*–2 coordenadas deste ponto são nulas, naturalmente. A coordenada de índice *n*–1 pode ser obtida diretamente a partir da fórmula (4.12). Como $x^{\hat{\eta}}_n$ vale $1 - D^*_{n-1}/H_{n-1}$, cada uma das demais coordenadas do ponto $x^{\hat{\eta}}$ pode valer, no máximo, D^*_{n-1}/H_{n-1}, de modo que $\overline{e}_{n-1} = \left(0, 0, \ldots, D^*_{n-1}/H_{n-1}\right)$.

Chamemos, então, de Δ_{n-1} o triângulo formado pela origem, por $\overline{e}_{n-1}$ e pelo ponto da base do simplex $\overline{S}_{n-2}$ que está mais próximo de $\overline{e}_{n-1}$. Definamos, também, o triângulo Δ^P_{n-1}, formado por $\overline{e}_{n-1}$, pelo ponto (0, 0, ..., $x^{\hat{\eta}}_{n-1}$) e pelo ponto do simplex de dimensão *n*–3 (que passa pelo centro do hipercubo de índice $\hat{\eta}$) que está mais próximo de $\overline{e}_{n-1}$. Dada a semelhança entre esses triângulos retângulos (análogo à Figura 19), obtemos

$$\frac{D^*_{n-2}}{\overline{H}_{n-2}} = \frac{\left(D^*_{n-1}/H_{n-1}\right) - x^{\hat{\eta}}_{n-1}}{D^*_{n-1}/H_{n-1}}.$$

Isolando $x^{\hat{\eta}}_{n-1}$ nesta equação, chegamos a

$$x^{\hat{\eta}}_{n-1} = \frac{D^*_{n-1}}{H_{n-1}} - \frac{D^*_{n-1} D^*_{n-2}}{H_{n-1}\overline{H}_{n-2}}. \tag{4.15}$$

Finalmente, substituindo (4.14) em (4.15), obtemos

$$x^{\hat{\eta}}_{n-1} = \frac{D^*_{n-1}}{H_{n-1}} - \frac{D^*_{n-2}}{H_{n-2}}. \tag{4.16}$$

Para calcular as demais coordenadas de $x^{\hat{\eta}}$, seguimos um procedimento análogo àquele utilizado para encontrar $x^{\hat{\eta}}_{n-1}$. Suponhamos, por exemplo, que as coordenadas $x^{\hat{\eta}}_{n-i+1}, \ldots, x^{\hat{\eta}}_n$ já tenham sido determinadas e que queiramos obter $x^{\hat{\eta}}_{n-i}$. Neste caso, a localização do hipercubo na camada k_{n-i} é feita usando uma fórmula equivalente a (4.9). Da mesma forma, D^*_{n-i} e $\overline{H}_{n-i}$

são obtidos usando equações similares a (4.13) e (4.14). Já a $i^{\text{ésima}}$ coordenada do ponto $\overline{e}_{n-i}$ é dada por

$$\overline{e}_{n-i,\,i} = 1 - \sum_{j=i+1}^{n} x_j^{\hat{\eta}} = \frac{D^*_{n-i}}{H_{n-i}}.$$

Utilizando, então, semelhança de triângulos, a exemplo do que foi feito anteriormente, chegamos à seguinte fórmula geral para $x_{n-i}^{\hat{\eta}}$:

$$x_{n-i}^{\hat{\eta}} = \frac{D^*_{n-i}}{H_{n-i}} - \frac{D^*_{n-i-1}}{H_{n-i-1}} \tag{4.17}$$

4.3 UM EXEMPLO DO MÉTODO DE DISCRETIZAÇÃO EM TRÊS DIMENSÕES

Ilustraremos com um exemplo como é feita a determinação da coordenada do centro do $\hat{\eta}^{\text{ésimo}}$ cubo dentre os η existentes em um simplex de três dimensões.

Inicialmente, localizamos a camada em que se encontra o cubo no simplex tridimensional. Isso é feito conforme mostrado na Figura 21.

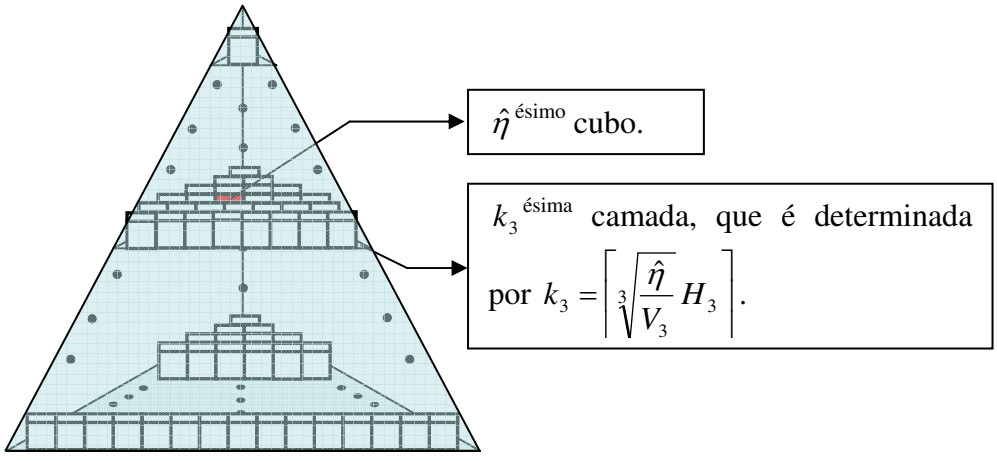


Figura 21 – Localização de um cubo em um simplex de 3 dimensões.

Em seguida, utilizando a camada de índice k_3, localizamos o quadrado no simplex bidimensional que serve de base para o $\hat{\eta}^{\text{ésimo}}$ cubo. Segundo a equação (4.9), a posição deste quadrado é dada por

$$\hat{\eta}_2 = \hat{\eta} - \left(\frac{k_3 - 1}{H_3} \right)^3 V_3 .$$

A camada em que se encontra este quadrado é ilustrada na Figura 22.

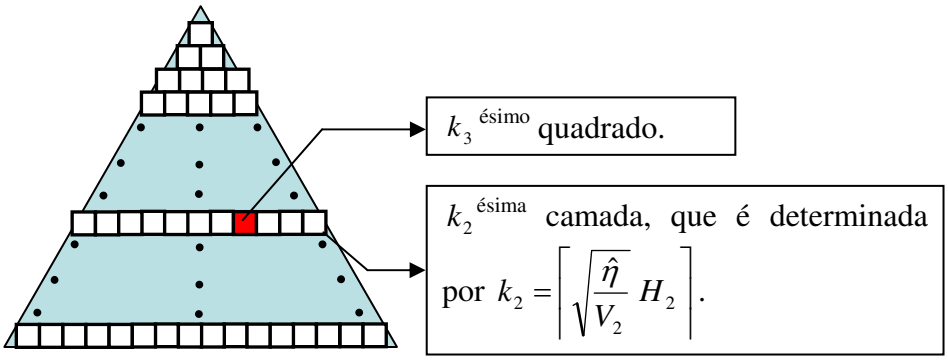


Figura 22 – Localização de um quadrado em um simplex de 2 dimensões.

Utilizando, agora, a camada de índice k_2, localizamos, no simplex de dimensão 1, o segmento de reta que sobre o qual se apóia o $\hat{\eta}_2^{\text{ésimo}}$ quadrado no simplex de dimensão 2. Novamente, a equação (4.9) é usada para determinar a posição deste segmento de reta, que é dada por

$$\hat{\eta}_1 = \hat{\eta}_2 - \left(\frac{K_2 - 1}{H_2} \right)^2 V_2 .$$

A camada em que se encontra este segmento de reta é ilustrada na Figura 23.

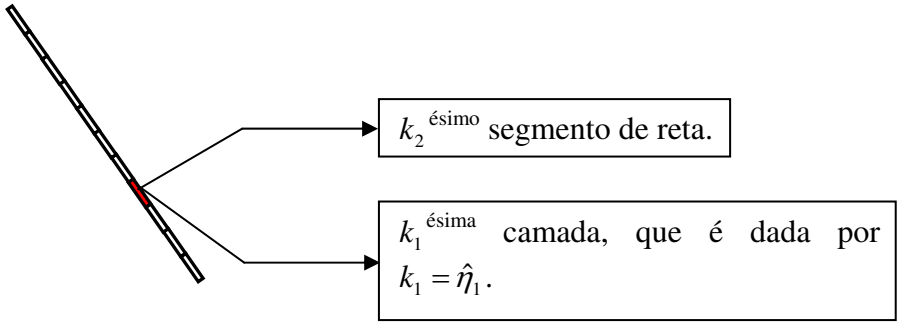


Figura 23 – Localização de um segmento de reta em um simplex de 1 dimensão.

Uma vez determinadas as camadas, é possível encontrar as coordenadas de centro do $\hat{\eta}^{\text{ésimo}}$ cubo utilizando as fórmulas (4.12) e (4.17). Essas camadas são

$$x_4^{\hat{\eta}} = 1 - \frac{D_3^*}{H_3}$$

$$x_{4-i}^{\hat{\eta}} = \frac{D_{4-i}^*}{H_{4-i}} - \frac{D_{3-i}^*}{H_{3-i}} \quad i = 1, 2 \ e \ 3,$$

onde $D_j^* = (k_j - 1/2)\,a$.

4.4 EVITANDO PONTOS INFACTÍVEIS

Como o número de hipercubos em uma camada *k* é calculado a partir da base da camada, que está a uma distância *ka* do vértice de referência do simplex, enquanto o centro do hipercubo está a uma distância (*k* – 1/2)*a* desse mesmo vértice (como mostrado na Figura 17), é possível que algumas coordenadas do centro de um hipercubo situado na fronteira do simplex assumam valores negativos ou superiores a 1, tornando infactível o ponto $x^{\hat{\eta}}$.

Para evitar que isso aconteça, podemos usar um artifício que consiste em reduzir as arestas dos hipercubos, de modo que o centro deste esteja sempre dentro do simplex. Assim, em lugar de usarmos *a* constante, empregamos

$$\begin{aligned} a_{n-1} &= a, \\ a_{n-i} &= \left[\prod_{j=n-i+1}^{n-1} \frac{\left(k_j - 1/2\right)}{k_j} \right] a. \end{aligned} \tag{4.18}$$

A Figura 24 ilustra o efeito dessa alteração sobre os valores de D^*, para um problema no qual temos três ativos e dividimos o simplex bidimensional em 2^7 hipercubos.

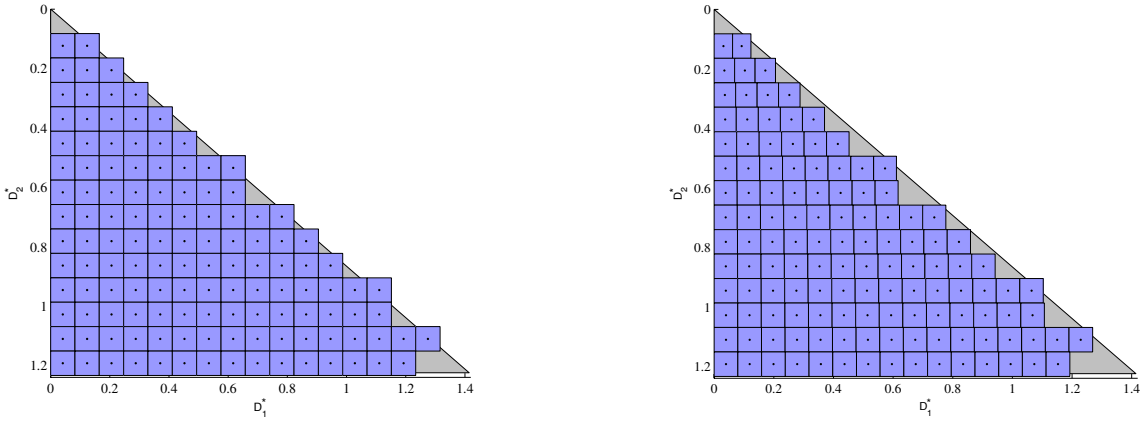


Figura 24 – Hipercubos em $\Re^2$.

O gráfico à esquerda mostra a estratégia original de cálculo de D^*, enquanto o gráfico à direita mostra o efeito da nova estratégia, em que o termo *a* em (4.11) é dado por (4.18). Nas duas figuras, o eixo horizontal está associado a D_1^* e o eixo vertical a D_2^*. Os retângulos azuis representam os hipercubos nos quais o simplex foi decomposto. O centro do $i^{\text{ésimo}}$ retângulo é o ponto $\left(D_1^*, D_2^*\right)$. O triângulo ao fundo é a figura que obteríamos fazendo *a* tender a zero.

Como se observa Figura 24, o artifício aqui apresentado promoveu uma redução do comprimento da aresta horizontal do hipercubo. No gráfico esquerdo, os hipercubos são quadrados de lado *a*. Neste caso, as arestas de base dos hipercubos pertencem à região cinza, mas

alguns dos centros dos hipercubos não. Já no gráfico à direita, todos os centros dos hipercubos pertencem à região desejada.

A Figura 24 ilustra outra característica da aproximação por hipercubos. As fronteiras esquerda e inferior do triângulo cinza são aproximadas de forma adequada, mas a fronteira direita apresenta um recorte indesejável. Naturalmente, esse efeito é minimizado aumentando-se o número de bits utilizados para representar os pontos do simplex. Deve-se observar que, no exemplo apresentado, gastamos apenas 7 bits (menos de 1 byte) para armazenar pontos em $\Re^3$.

A Figura 25 mostra os mesmos hipercubos da Figura 24, agora sobre o simplex em $\Re^3$. Os pontos no centro dos retângulos são aqueles passíveis de representação usando-se 7 bits. No gráfico à esquerda, associado à estratégia usual de cálculo de D^*, alguns desses pontos estão fora da região factível. Por outro lado, os pontos centrais da figura à direita, construída com a nova estratégia, são todos factíveis. Mais uma vez, a aproximação por um número pequeno de hipercubos tornou recortada uma das arestas do simplex.

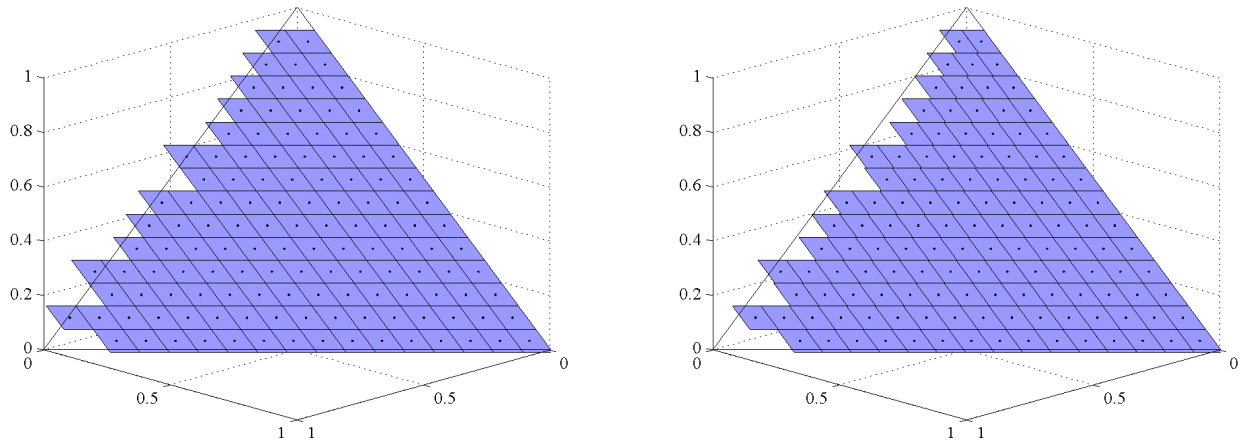


Figura 25 – Os hipercubos da Figura 24 mostrados sobre o simplex em $\Re^3$.

4.5 A BOA DISTRIBUIÇÃO GERADA PELO MÉTODO DE DISCRETIZAÇÃO

A Figura 13, mostrada no capítulo 3, sugere que o método proposto por Chang *et al.* não apresenta uma boa distribuição de pontos. Repetiremos os procedimentos adotados naquele capítulo utilizando, agora, o método de discretização do espaço recém apresentado. Para tanto, vamos dividir o simplex bidimensional em 1677216 pequenos quadrados, número este obtido pela fórmula $\eta = 2^{(n-1)d}$, com $n = 3$ e $d = 12$. Empregamos, assim, o mesmo número de bits que o algoritmo proposto por Chang *et al*.

Feita a geração de 15000 números aleatórios uniformemente distribuídos entre 1 e 1677216, podemos determinar as coordenadas dos centros dos quadrados a eles associados através da fórmula (4.17), utilizando a estratégia para evitar pontos infactíveis. Obtemos, assim, os pontos mostrados na Figura 26.

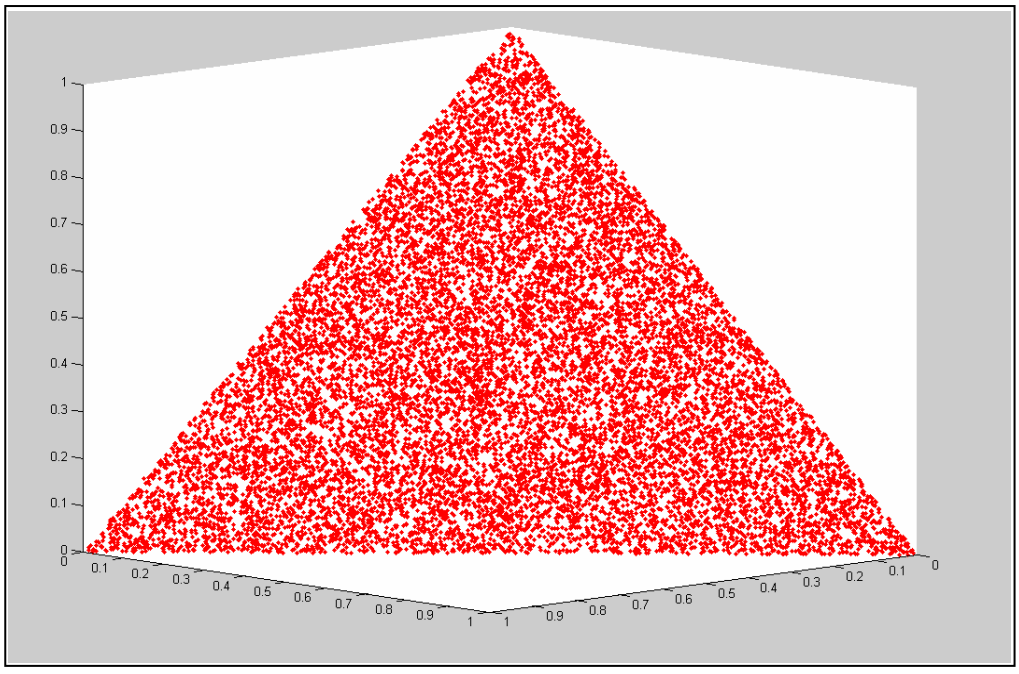


Figura 26 – Distribuição dos 15000 indivíduos sobre um simplex de 2 dimensões.

Observa-se, na Figura 26, que os pontos gerados possuem uma distribuição muito mais uniforme que aquela exibida na Figura 13, evidenciando a eficiência do método que trabalha com a discretização do espaço de soluções.

CAPÍTULO 5

UM NOVO ALGORITMO GENÉTICO

5.1 INTRODUÇÃO

Com a discretização do espaço de soluções, é preciso reformular o algoritmo genético proposto por Chang *et al.*, pois este trabalha com soluções que podem não pertencer ao simplex, ou seja, à região factível.

As mudanças se fazem mais necessárias na representação das soluções e na forma de realização do *crossover* e da mutação, de modo a adequá-las ao novo tratamento de soluções e aproveitar algumas de suas vantagens. Já o cálculo da aptidão dos indivíduos (equação (2.2)), por exemplo, não exige alterações.

Neste capítulo, apresentaremos uma nova formulação do algoritmo genético para a determinação da fronteira eficiente de investimento.

5.2 NOVO ALGORITMO GENÉTICO PARA DETERMINAÇÃO DA F.E.I.

5.2.1 Representação.

A representação das soluções proposta por Chang *et al.* [3] divide cada cromossomo em duas partes, sendo uma referente aos valores de investimentos e a outra aos ativos escolhidos para investimento. Na nova representação, essa divisão é mantida. Entretanto, os valores referentes aos investimentos já são as proporções de investimento, não sendo necessária a mudança de variável como na formulação anterior.

As proporções de investimento são as coordenadas do centro do hipercubo que está em uma determinada posição do simplex. Assim, as proporções de investimento são representadas apenas pela posição do hipercubo no simplex, ou seja, de forma implícita. Como esta posição é dada por um único número inteiro positivo, optou-se pela representação deste na base 2. Logo, a parte do cromossomo associada às proporções de investimento é um vetor binário.

Com relação à parte de cromossomo que contém os ativos escolhidos para investimentos, adotou-se a representação através de um vetor inteiro, como sugerido por Chang *et al.* Um exemplo de representação do cromossomo é dado na Figura 27.

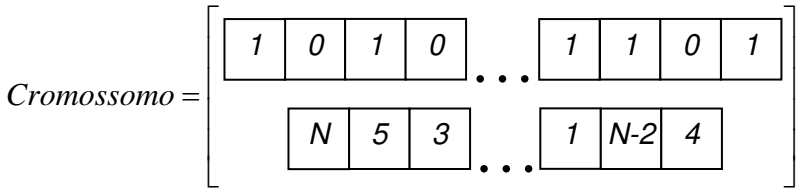


Figura 27 – Nova representação do cromossomo.

É importante salientar que, após a conversão do número binário para as coordenadas do centro do hipercubo, a $j^{\text{ésima}}$ coordenada estará associada à $j^{\text{ésima}}$ componente do vetor de ativos (segunda parte do cromossomo). A Figura 28 mostra um exemplo no qual fica clara a associação entre as partes do cromossomo.

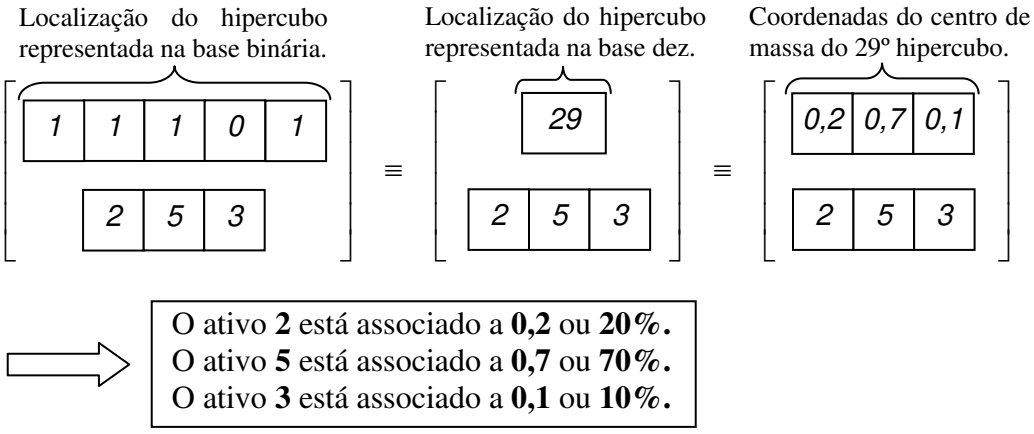


Figura 28 – Exemplo de transformação da primeira parte do cromossomo para o vetor de coordenadas.

No exemplo da Figura 28, a transformação da localização do hipercubo para o vetor de coordenada foi feita sem a consideração das restrições de limite inferior. Caso o modelo tenha restrição de limite inferior, ainda será preciso fazer uma mudança de variável. Assim, após a determinação das coordenadas do centro do hipercubo selecionado, as proporções de investimentos w_i, para os ativos que pertencem à carteira, serão obtidas utilizando-se a fórmula

$$w_i = \varepsilon_i + x_j^{\hat{\eta}} F \ , \tag{5.1}$$

onde ε_i é a mínima proporção de investimento admitida para o ativo *i*, $x_j^{\hat{\eta}}$ é a $j^{\text{ésima}}$ componente do vetor de coordenadas do $\hat{\eta}^{\text{ésimo}}$ hipercubo selecionado e *F* é a proporção livre para investimento dada por $F = 1 - \sum_{i \in Q} \varepsilon_i$.

Para tornar clara esta representação, tomemos o exemplo da Figura 28 para o caso em que $\varepsilon_i = 0{,}1, \forall i$. Naturalmente, não podemos associar diretamente o vetor de coordenadas

$(0{,}2\ \ 0{,}7\ \ 0{,}1)$ aos ativos $\{2, 3, 5\}$, já que existe a restrição de limite inferior ($\varepsilon_i = 0{,}1, \forall i$). Assim, dada a proporção livre de investimento, que é $F = 1 - \sum_{i \in \{2,3,5\}} \varepsilon_i = 0{,}7$, aplicamos a fórmula (5.1), obtendo as proporções de investimento

$$w_2 = 0{,}1 + 0{,}7 \,.\, 0{,}2 = 0{,}24,$$
$$w_3 = 0{,}1 + 0{,}7 \,.\, 0{,}7 = 0{,}59,$$
$$w_5 = 0{,}1 + 0{,}7 \,.\, 0{,}1 = 0{,}17.$$

Nota-se que esses valores satisfazem às restrições de limite inferior e de soma igual a 1.

5.2.2 Geração da população inicial.

Na criação da população inicial, a primeira parte do cromossomo é obtida através da geração de um número aleatório inteiro entre 1 e o número de hipercubos contidos no simplex, dado por $\eta = 2^{(n-1)d}$. A segunda parte, que representa os ativos escolhidos para investimento, é obtida pela geração de K números aleatórios inteiros não repetidos entre 1 e N.

5.2.3 *Crossover*.

O tipo de *crossover* escolhido foi o de 1-ponto[7], o que permitiu que as duas partes do cromossomo pudessem ser submetidas ao mesmo operador genético. Assim, para dois pais selecionados, escolhe-se aleatoriamente um bit (comum aos dois pais) e troca-se, entre eles, a

[7] Este tipo de *crossover*, proposto por Holland [7], é o mais simples da família dos cruzamentos com n-pontos.

parte do cromossomo a partir deste bit.

Para aplicar o *crossover* na primeira parte do cromossomo (vetor binário), deve-se escolher um bit aleatoriamente e realizar uma simples troca de segmentos de cromossomos, gerando assim dois filhos que são inseridos na população. A Figura 29 mostra um exemplo deste tipo de *crossover*.

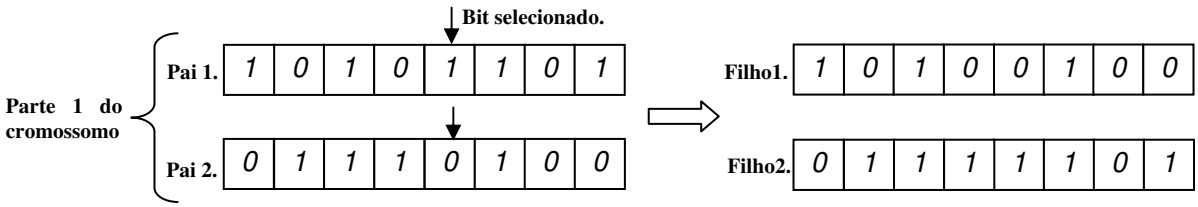


Figura 29 – Exemplo de *crossover* na parte 1 do cromossomo.

Neste caso, a segunda parte do cromossomo não é modificada, de modo que os filhos 1 e 2 herdam, respectivamente, a segunda parte do cromossomo dos pais 1 e 2.

Caso a segunda parte do cromossomo seja selecionada para o *crossover*, deve-se escolher um ativo aleatoriamente dentre aqueles que não são comuns aos dois cromossomos pais e realizar a troca dos segmentos de cromossomo apenas com os ativos não repetidos. Por fim, deve-se inserir nos filhos os genes repetidos nas mesmas posições em que eles estavam nos pais. Esta estratégia, conhecida como *crossover OX*, proposta por Davis [5], evita a possibilidade de um determinado filho possuir ativos repetidos. Veja na Figura 30, um exemplo de *crossover* na parte 2 do cromossomo.

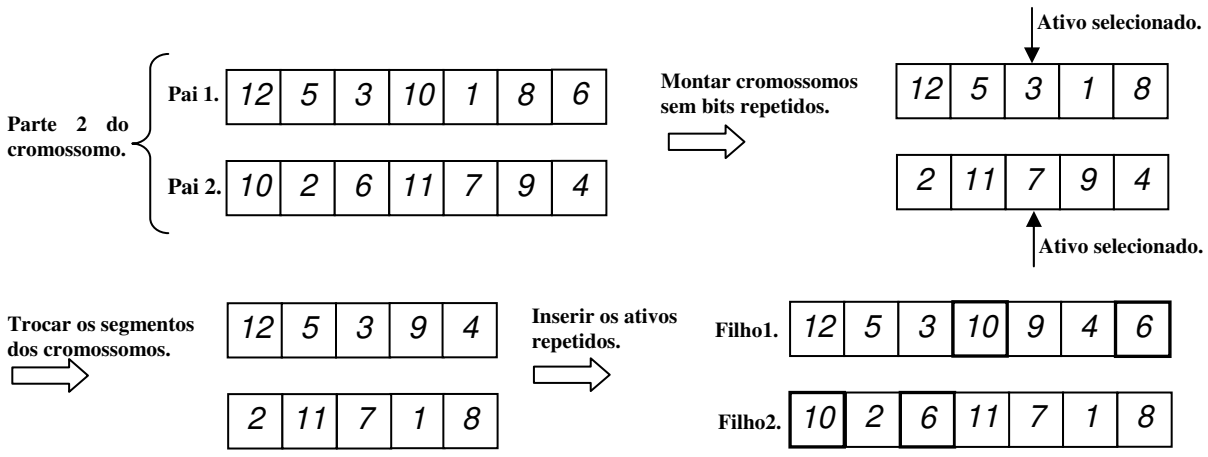


Figura 30 – Exemplo de *crossover* na parte 2 do cromossomo.

Neste caso, a primeira parte do cromossomo não é modificada. Assim, os filhos 1 e 2 herdam, respectivamente, a primeira parte do cromossomo dos pais 1 e 2.

Nesta nova formulação, o *crossover* não é aplicado apenas sobre dois indivíduos da população, como na formulação de Chang *et al*. Define-se uma fração de ocorrência de *crossover*, *PerCross*, que indica qual percentual da população passará por esse procedimento. Além disso, define-se também a probabilidade de ocorrência de *crossover* na parte 1 do cromossomo, associada ao parâmetro *ProbCross*. Naturalmente, a probabilidade de ocorrência de *crossover* na parte 2 do cromossomo é igual a 1–*ProbCross*. Desta forma, o *crossover* não ocorre simultaneamente nas duas partes do cromossomo.

Os indivíduos que não foram selecionados para o *crossover* são duplicados e suas cópias são inseridas na população. Este procedimento garante que, ao final da iteração, a população tenha o dobro de indivíduos, o que será útil para a estratégia de seleção, que será apresentada na subseção 5.2.5.

5.2.4 Mutação.

Como o cromossomo possui duas partes, uma representada por um vetor binário e outra por um vetor de inteiros positivos, o procedimento de mutação não pode ser o mesmo nas duas partes.

Para a mutação na parte 1 do cromossomo, utiliza-se o procedimento clássico de mutação para vetores binários, no qual um dado bit b_i selecionado aleatoriamente é substituído por $1 - b_i$. Veja, na Figura 31, um exemplo de mutação na parte 1 do cromossomo.

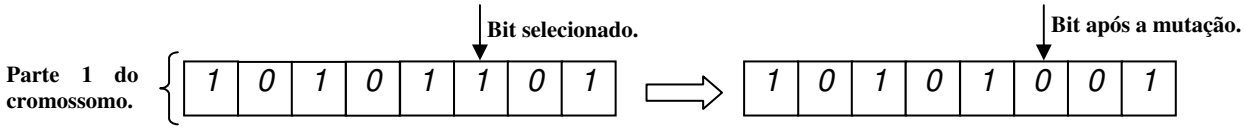


Figura 31 – Exemplo de mutação na parte 1 do cromossomo.

Já a mutação na parte 2 do cromossomo pode ser feita através da permutação de ativos ou da substituição de ativos. A permutação foi utilizada para permitir uma troca de frações de investimento entre ativos. A substituição de ativos é um procedimento útil no caso em que a população está estabilizada, contendo muitos cromossomos com os mesmos ativos. Neste caso, os ativos que não estão presentes em nenhum cromossomo da população só terão chances de entrar na população através de um procedimento de substituição. Veja, na Figura 32, um exemplo de mutação na parte 2 do cromossomo.

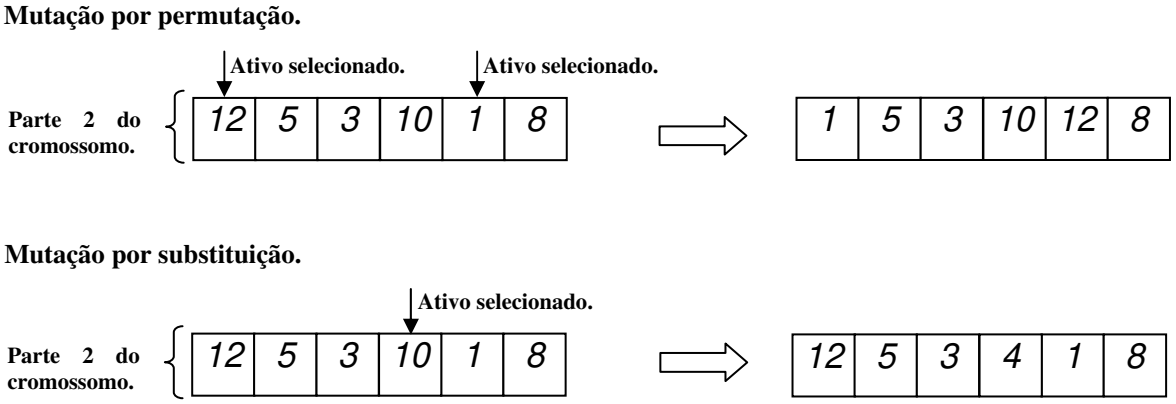


Figura 32 – Exemplos de mutações na parte 2 do cromossomo.

Nesta nova formulação, a mutação não ocorrerá em apenas um indivíduo e somente nos cromossomos filhos como na formulação de *Chang et al.* Adotou-se um percentual de ocorrência de mutação, *PerMut*, que indica o percentual da população, incluindo os filhos, que passará pela mutação. Existem também as probabilidades de ocorrência de mutação nas partes 1 e 2 do cromossomo, dadas, respectivamente, por *ProbMut* e 1-*ProbMut*. Observa-se, assim, que mutação não ocorre simultaneamente nas duas partes do cromossomo.

Escolhida a parte a ser alterada, a variável *PerMutP*1 ou *PerMutP*2 determina o número de bits ou ativos da parte 1 ou 2 do cromossomo que sofrerão mutação. A mutação na parte 2 será do tipo substituição, com probabilidade *ProbMutP*2, ou permutação, com probabilidade 1-*ProbMutP*2.

5.2.5 Seleção.

Após o término dos procedimentos de *crossover* e mutação, a população possui o dobro do número de indivíduos, haja vista que dois filhos são gerados em cada *crossover*, e os cromossomos não selecionados para o *crossover* são duplicados. Deste modo, é possível realizar a seleção por torneio binário, conforme proposta de Brindle [2]. Neste procedimento, pares de cromossomos são agrupados aleatoriamente e, em seguida, o melhor indivíduo de cada par é selecionado para a nova geração. A vantagem deste tipo de seleção é a manutenção da diversidade de soluções na população, facilitando a fuga de ótimos locais. A Figura 33 ilustra o procedimento de seleção.

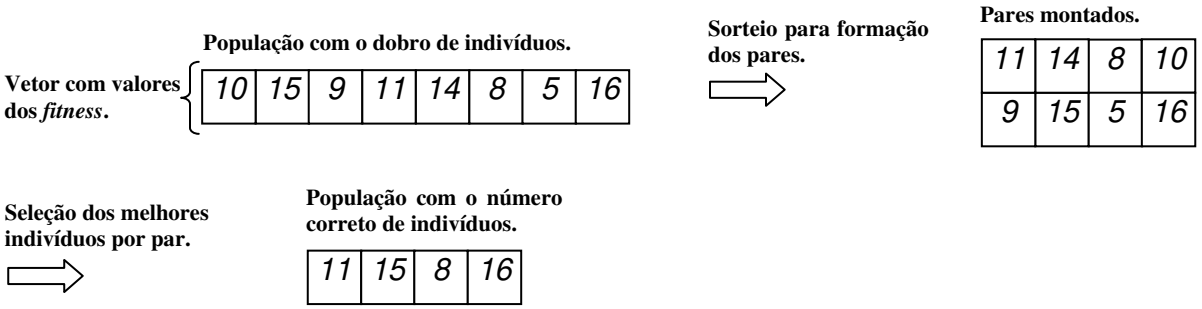


Figura 33 – Exemplo de seleção por torneio binário.

5.2.6 O algoritmo.

Resumimos a seguir os passos básicos do novo algoritmo genético.

Algoritmo 3 (Novo Algoritmo Genético).

1. *Gerar uma população inicial com 100 indivíduos como em 5.2.2.*

2. *Calcular o fitness dos indivíduos da população inicial.*

3. ***Em cada iteração:***

3.1. *Selecionar um percentual PerCros indivíduos da população para crossover.*

3.2. *Duplicar os indivíduos que não foram selecionados para crossover e inserir suas cópias na população.*

3.3. *Realizar o Crossover nos indivíduos selecionados, obedecendo as probabilidades ProbCross e 1-ProbCross.*

3.5 *Inserir os dois filhos gerados na população.*

3.5. *Selecionar um percentual PerMut da população para mutação.*

3.6. ***Se** a mutação selecionada for na parte 2 (com probabilidade 1-ProbMut)**:***

3.6.1 *Realizar mutação na parte 2 (com percentual de ocorrência nos ativos de PerMutP2). A mutação tipo substituição de ativos é aplicada com probabilidade ProbMutP2 e a mutação do tipo permutação é aplicada com probabilidade 1-ProbMutP2.*

3.7 ***Senão** (com probabilidade ProbMut)**:***

3.7.1 *Realizar mutação na parte 1, sobre um percentual de PerMutP1 bits.*

3.8 ***Fim.***

3.9 *Calcular o fitness dos filhos e dos cromossomos que sofreram mutação.*

3.10 *Selecionar os indivíduos que permanecerão na população.*

4. ***Até** atingir o número máximo de iterações.*

O algoritmo acima é repetido para cada valor de $\lambda \in [0,1]$. O número máximo de iterações deste algoritmo, e sua relação com os percentuais de *crossover* e mutação, que são dados de entrada, será discutido no capítulo 7.

CAPÍTULO 6

REFORMULAÇÃO DA LOCALIZAÇÃO DOS HIPERCUBOS

6.1 O ERRO ASSOCIADO À DISCRETIZAÇÃO DO ESPAÇO

Uma vez que o novo algoritmo genético trabalha com a discretização do espaço, existe a possibilidade de surgimento de um erro decorrente da aproximação de uma região (que pode ser segmento de reta, quadrado, cubo ou hipercubo) por apenas um ponto. Tomaremos como medida deste erro a norma infinito do vetor $\vec{v}$, que tem como origem o centro do hipercubo e que termina em um dos vértices deste. Assim, o erro de discretização será dado por

$$\xi_D = \left\| \vec{v} \right\|_\infty .$$

A vantagem deste critério de medição é que ele fornece a maior variação de proporção de investimento dentro de um hipercubo.

Obviamente o erro é menor à medida que diminuímos o tamanho das arestas dos hipercubos. Porém, a redução das arestas implica no aumento do número de hipercubos usados para representar todo o espaço. Como exemplo, se estivermos trabalhando com um simplex de dimensão 2 (três ativos) e o número de quadrados for $\eta = 1048576$ ($\eta = 2^{(n-1)d}$, onde n = 3 e d = 10), então o tamanho das arestas dos quadrados será, segundo a fórmula (4.1), $a = 0{,}00090879$. Neste caso, o erro será igual a 0,00037101, que corresponde a uma variação de cerca de 0,0371% na proporção de investimento de cada ativo. Caso estivéssemos trabalhando com um simplex de dimensão 9 (dez ativos) e com $\eta = 1048576$ hipercubos (mesmo número do anterior), então o tamanho das arestas dos hipercubos seria $a = 0{,}05872845$, gerando um erro igual a $\xi_D = 0{,}02397579$, equivalente a uma variação de 2,398% na proporção de investimento. Para mantermos o mesmo nível de erro do exemplo anterior (0,0371%), teríamos que elevar o número de hipercubos para um valor acima de $1{,}9.10^{22}$.

Segundo a norma para ponto flutuante 754-1985 da IEEE (*Institute for Eletrical and Eletronic Engineers*), os computadores padrão devem trabalhar com um co-processador numérico com 64 bits (dígitos binários), sendo 12 bits destinados ao sinal e à parte exponencial, sobrando apenas 52 bits para a mantissa, o que equivale a um número com precisão de 16 a 17 dígitos decimais. Neste trabalho, adotamos o programa Matlab, que utiliza esta representação com ponto flutuante, mesmo para números inteiros. Como o número de hipercubos necessários para trabalhar com uma carteira de 10 ativos com um erro abaixo de 0,0371% é superior a $1{,}9.10^{22}$, a proposta de discretização do espaço com enumeração de hipercubos pode tornar-se inviável.

6.2 REFORMULAÇÃO DO MÉTODO DE LOCALIZAÇÃO DOS HIPERCUBOS

O principal problema decorrente da representação proposta no capítulo 5 é que é necessária uma quantidade elevada de hipercubos, mesmo que trabalhemos com uma carteira de

pequeno porte. Entretanto, a precisão da máquina em ponto flutuante só nos permite usar 17 dígitos. Como a enumeração dos hipercubos é feita utilizando-se números naturais, poderíamos tentar utilizar os 64 bits do processador numérico apenas para a representação da localização dos hipercubos. Não obstante, o número inteiro obtido teria, no máximo, 20 dígitos na base decimal, o que ainda seria insuficiente para resolver, com precisão aceitável, um problema com 10 ativos.

Para contornar esse problema, seria necessário trabalhar com uma biblioteca específica capaz de armazenar e executar operações aritméticas com números com precisão infinita. Entretanto, apesar desse tipo de biblioteca existir para linguagens como C ou C++, não há nada semelhante para o Matlab. Tentamos, então, usar vetores para representar números inteiros, associando cada algarismo do número a uma componente do vetor. Assim, por exemplo, o número 977452342656813982 passou a ser armazenado na forma

$$\begin{bmatrix} 9 & 7 & 7 & 4 & 5 & 2 & 3 & 4 & 2 & 6 & 5 & 6 & 8 & 1 & 3 & 9 & 8 & 2 \end{bmatrix}.$$

Esta forma de representação exigiu que criássemos funções particulares para efetuar as operações aritméticas básicas, o que, infelizmente, tornou muito lento o novo algoritmo. De fato, o simples produto entre dois números com representação de vetores exigiu diversas operações usuais para escalares (multiplicação, adição e subtração), o que impediu que a nova proposta fosse comparável, em termos de tempo computacional, com o algoritmo de Chang *et al.* [3]. Por esse motivo, vimo-nos obrigados a adotar uma nova estratégia de localização dos hipercubos, utilizando camadas.

6.2.1 Localização por camadas.

No processo de determinação das proporções de investimento a partir da discretização da região factível, apresentado no capítulo 4, o número inteiro associado a um hipercubo $\hat{\eta}$ era

convertido em um vetor contendo os índices das camadas que continham esse hipercubo. Em seguida, determinavam-se as coordenadas do centro do hipercubo e, a partir delas, as proporções de investimento.

Dada a dificuldade em trabalhar com números inteiros grandes, vamos eliminar o primeiro passo desse processo e representar cada hipercubo diretamente através das camadas às quais ele pertence. Segundo essa nova estratégia, a localização do $\hat{\eta}^{\text{ésimo}}$ hipercubo será fornecida pelo vetor $\left(\hat{k}_1,\ldots,\hat{k}_{n-1}\right)$, em que $\hat{k}_i$ é a camada na dimensão *i* na qual se encontra o hipercubo. Assim, a determinação das proporções de investimento exigirá apenas o uso das fórmulas (4.12) e (4.17) para determinação do centro do hipercubo, e a mudança de variável utilizada para que a solução satisfaça os limites inferiores estabelecidos.

A representação por camadas mantém a idéia da discretização do espaço com hipercubos, sem usar esta representação explicitamente, o que nos permite trabalhar com números inteiros pequenos. Para ilustrar essa característica, vamos supor que queiramos representar $1{,}9.10^{22}$ hipercubos em um simplex de dimensão 9. Neste caso, o número de camadas na dimensão 1 (a dimensão com a maior quantidade de camadas) será igual a 1550, um número muito baixo[8].

6.3 ALTERAÇÃO NO NOVO ALGORITMO GENÉTICO

Poucas alterações precisam ser feitas no novo algoritmo genético para que possamos usar o método de localização do hipercubo por camadas. Apenas a representação e a geração da população inicial devem ser modificadas. Mantém-se, portanto, o mesmo algoritmo proposto na subseção 5.2.6.

[8] Esse número é dado por $\lceil H_1/a \rceil$, onde H_1 é calculado segundo (3.6) e a é fornecido por (4.1).

6.3.1 Representação.

A representação continua sendo feita por um cromossomo dividido em duas partes, sendo a primeira delas composta por um vetor binário e a segunda por um vetor de números inteiros, conforme a Figura 27. Entretanto, a primeira parte não mais fornece um número inteiro que representa a localização do hipercubo, mas um conjunto de números inteiros que representarão as camadas.

A Figura 34 fornece um esquema do vetor binário que armazena a primeira parte do cromossomo. O vetor é dividido em $n-1$ segmentos de igual tamanho. O $i^{\text{ésimo}}$ segmento deste vetor é um número inteiro β_i que, depois de escalado convenientemente, fornece, para a $i^{\text{ésima}}$ dimensão, a camada na qual se encontra o hipercubo.

$$\Big[\underbrace{0\ 1\ \ldots\ 1\ 1}_{\beta_1}\ \underbrace{1\ 0\ \ldots\ 1\ 0}_{\beta_2}\quad\cdots\quad\underbrace{1\ 1\ \ldots\ 1\ 1}_{\beta_{n-2}}\ \underbrace{0\ 1\ \ldots\ 0\ 0}_{\beta_{n-1}}\Big]$$

Figura 34 – Segmentos do cromossomo que determinam as camadas.

Para determinar as camadas a partir dos valores β_i, usamos a fórmula

$$k_i = \left\lceil \frac{k_{\max_i}}{k_{\max}} \beta_i \right\rceil, \tag{6.1}$$

onde $k_{\max_i} = \lceil H_i / a \rceil$ representa o número máximo de camadas existentes na dimensão i e $k_{\max} = 2^{\alpha} - 1$. O valor α é obtido a partir de

$$\alpha = \left\lceil \frac{\log\left(\max_i \left\{ k_{\max_i} \right\} + 1\right)}{\log(2)} \right\rceil,$$

onde $\max_i \{k_{\max_i}\}$ fornece o maior número de camadas usadas em uma dimensão. Na prática, como já comentado na subseção 6.2.1, este valor será sempre $\lceil H_1/a \rceil$, pois H_1 é a maior altura.

O termo α fornece o número de bits necessários para representar $\max_i \{k_{\max_i}\}$ em uma base binária. Deste modo, $k_{\max}$ representa o maior número inteiro representável na base binária com número de bits igual ao utilizado para representar $\max_i \{k_{\max_i}\}$. O objetivo dessa representação é garantir que todos os bits do vetor binário do cromossomo possam ser alterados e que cada segmento deste cromossomo tenha o mesmo comprimento.

Vamos recorrer mais uma vez a um exemplo para ilustrar o escalamento definido acima. Tomemos, então, um simplex de 4 dimensões e um cromossomo cuja primeira parte é dada pelo vetor binário da Figura 35. Os valores binários convertidos para a base dez são denotados β_i.

$$[\underbrace{1\ 1\ 1\ 0\ 1}_{\beta_1 = 29}\ \underbrace{0\ 0\ 1\ 1\ 0}_{\beta_2 = 6}\ \underbrace{1\ 0\ 0\ 1\ 0}_{\beta_3 = 18}\ \underbrace{1\ 1\ 1\ 1\ 0}_{\beta_4 = 30}]$$

Figura 35 – Exemplo das partes do cromossomo que representarão as camadas.

Neste exemplo, as dimensões 1, 2, 3 e 4 do simplex discretizado possuem, respectivamente, 26, 25, 24 e 23 camadas. Logo, o número de bits necessários para representar o maior número de camadas é $\alpha = 5$. De fato, com esse número de bits somos capazes de representar $k_{\max} = 31$ camadas. Usando, então, a fórmula (6.1) para escalar os valores β_i, obtemos as camadas reais correspondentes ao cromossomo da Figura 35, que são $k_1 = 25$, $k_2 = 5$, $k_3 = 14$ e $k_4 = 23$.

6.3.2 Geração da população inicial.

Na proposta de geração da população inicial apresentada na seção 5.2.2, a primeira parte do cromossomo, que continha a localização do hipercubo, era determinada através do sorteio de um número inteiro compreendido entre 1 e o número total de hipercubos do simplex. Como, agora, a localização do hipercubo é representada por um conjunto de camadas, devemos fazer um sorteio para cada uma das dimensões do simplex.

Entretanto, este sorteio precisa ser feito com cuidado, pois as camadas possuem números diferentes de hipercubos. No simplex da Figura 36, por exemplo, a camada k' é menor que a camada k'', contendo, conseqüentemente, menos hipercubos. Naturalmente, as camadas que possuem mais hipercubos devem ter mais chance de serem sorteadas que outras que possuem menos hipercubos. Vejamos, então, como mensurar esta chance.

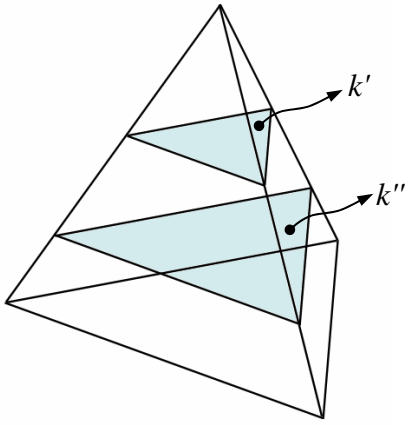


Figura 36 – Diferença nos tamanhos das camadas.

Analisemos, inicialmente, um simplex de dimensão 1, como aquele mostrado na Figura 37. Neste simplex, todas as camadas contêm apenas um hipercubo (ou segmento de reta).

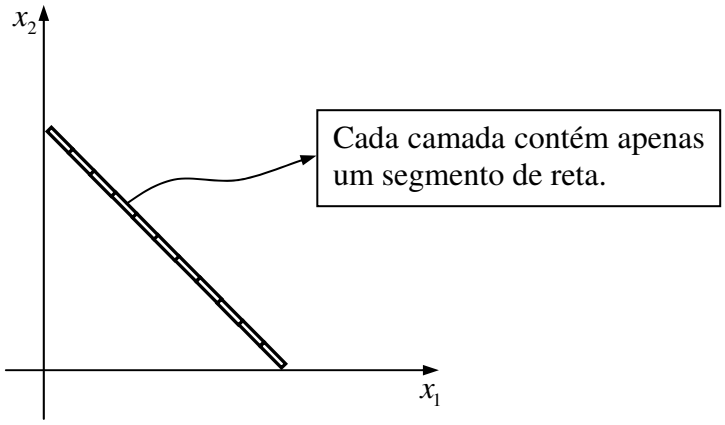


Figura 37 – Camadas de um simplex unidimensional.

Neste caso, todas as camadas devem ter igual probabilidade de serem escolhidas. Portanto, a variável aleatória correspondente ao sorteio das camadas deve ter como função de densidade de probabilidade

$$f(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & c.c., \end{cases}$$

que corresponde à geração de números aleatórios uniformemente distribuídos. Deste modo, a função de distribuição será

$$F(x) = \begin{cases} x & 0 \le x \le 1 \\ 0 & c.c. \ . \end{cases}$$

Assim, o gerador de números aleatórios usado para sortear a camada no simplex de dimensão 1 deve gerar números uniformemente distribuídos em $[0,1]$.

O simplex de dimensão 2, por sua vez, possui camadas que variam proporcionalmente de tamanho com a altura do simplex, como mostra a Figura 38.

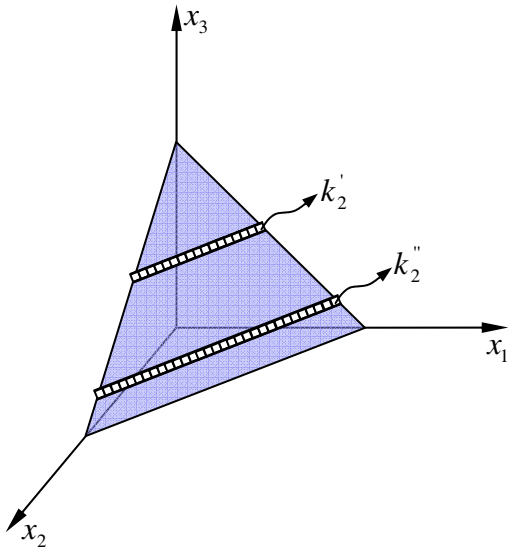


Figura 38 – Camadas de um simplex bidimensional.

Neste caso, a quantidade aproximada de hipercubos (quadrados) contidos nas camadas $k_2^{'}$ e $k_2^{''}$ é dada, respectivamente, por

$$NumHip_2^{'} = \frac{V_1^{B'}}{a} = k_2^{'} \frac{V_1}{H_2} \quad \text{e} \quad NumHip_2^{''} = \frac{V_1^{B''}}{a} = k_2^{''} \frac{V_1}{H_2},$$

onde $V_1^{B'}$ e $V_1^{B''}$ são os conteúdos (neste caso, os comprimentos) dos simplex de dimensão 1, obtidos a partir da equação (3.4).

Como o número de hipercubos varia proporcionalmente à camada, a probabilidade de escolha de uma determinada camada deve ser proporcional à posição desta camada. Neste caso, a variável aleatória correspondente ao sorteio das camadas obedece, aproximadamente, a função de densidade de probabilidade

$$f(x) = \begin{cases} 2x & 0 \le x \le 1 \\ 0 & c.c., \end{cases}$$

que corresponde à geração de números aleatórios com função de distribuição

$$F(x)=\begin{cases} x^2 & 0 \le x \le 1 \\ 0 & c.c. \ . \end{cases}$$

Assim, um gerador de números aleatórios para sorteio da camada no simplex de dimensão 2 pode ser dado por

$$Z_2 = F^{-1}(Z_u) = \sqrt{Z_u} \ ,$$

onde $Z_u \in [0\,,1]$ é um número aleatório com distribuição uniforme $[0\,,1]$.

Generalizando, para a camada k_{n-1} de um simplex de dimensão n-1, o número de hipercubos é dado por

$$\frac{V_{n-2}^{B}}{a^{n-2}} = (k_{n-1})^{n-2} \frac{V_{n-2}}{(H_{n-1})^{n-2}} \ ,$$

onde V_{n-2}^{B} é o conteúdo do simplex de dimensão n-2, definido, segundo a equação (3.4), como

$$V_{n-2}^{B} = \left(\frac{k_{n-1} a}{H_{n-1}} \right)^{n-2} V_{n-2} \ .$$

Como se observa, o número de hipercubos em cada camada é proporcional ao índice da camada elevado a (n-2)$^{\text{ésima}}$ potência. Deste modo, podemos fazer com que a variável aleatória correspondente ao sorteio das camadas obedeça, aproximadamente, a seguinte função de densidade de probabilidade

$$f(x)=\begin{cases}(n-1)x^{n-2} & 0\le x\le 1\\ 0 & c.c.,\end{cases}$$

que corresponde à geração de números aleatórios com função de distribuição

$$F(x)=\begin{cases}x^{n-1} & 0\le x\le 1\\ 0 & c.c.\ .\end{cases}$$

Para escolher aleatoriamente camadas em um simplex de dimensão n-1 $(n=1,2,3,\ldots)$, favorecendo, de forma adequada, as que têm mais hipercubos, um gerador de números aleatórios deve ser definido por

$$Z_{n-1}=F^{-1}(Z_u)=\sqrt[n-1]{Z_u}\ , \tag{6.2}$$

onde Z_u é um de número aleatório com distribuição uniforme $[0\,,1]$.

Em nosso algoritmo genético alterado, a geração aleatória da primeira parte do cromossomo é feita de forma que cada componente do vetor de camadas é gerada com o auxílio da fórmula (6.2). Em seguida o vetor é convertido para a notação binária, como representado na subseção 6.3.1. Já a segunda parte do cromossomo, que representa os ativos escolhidos para investimento, continua sendo determinada pela geração de números aleatórios inteiros não repetidos entre 1 e *N*.

CAPÍTULO 7

RESULTADOS COMPUTACIONAIS

7.1 INTRODUÇÃO

Neste capítulo apresentamos os resultados computacionais obtidos aplicando-se o novo algoritmo genético aos cinco problemas[9] selecionados por Chang *et al.* [3].

O novo algoritmo genético foi codificado em linguagem Matlab e executado em um computador com processador AMD Athlon 64 bits 2000MHz e com memória principal de 1 GB.

7.2 OS PROBLEMAS

Os cinco problemas práticos selecionados correspondem a cinco grandes bolsas de valores: Hang Seng (Hong Kong), com 31 ativos, DAX (Alemanha), com 85 ativos, FTSE (Reino

[9] Os dados dos problemas estão disponíveis em http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html.

Unido), com 89 ativos, S&P (Estados Unidos), com 98 ativos e Nikkei (Japão), com 225 ativos. Cada uma das instâncias possui restrição de cardinalidade $K = 10$ e limite mínimo de investimento de $\varepsilon_i = 0,01$ para os ativos que compõem a carteira.

7.3 IMPLEMENTAÇÃO DO ALGORITMO DE CHANG *ET ALII*

O algoritmo de Chang *et al.* também foi implementado em linguagem Matlab e executado no mesmo computador que o novo algoritmo genético. O número de iterações adotados por Chang *et al.* foi $T^* = 1000\,N$ (como explicado na seção 2.3), em que N é o número de ativos disponível para investimentos. Este número de iterações é executado para cada valor de λ na resolução do modelo (1.8).

Com relação ao número de operações de *crossover*, Chang *et al.* prescrevem apenas um por iteração, conforme explicado na subseção 2.3.3. A mutação também é realizada apenas uma vez por iteração (vide subseção 2.3.4) em um valor de investimento s_i de um cromossomo selecionado. Portanto, no algoritmo de Chang *et al.* não existem parâmetros de entrada referentes ao número de operações de *crossover* e mutação.

7.4 PÂRAMETROS DE ENTRADA DO NOVO ALGORITMO GENÉTICO

Os parâmetros de entrada do novo algoritmo genético foram determinados com base em testes numéricos. Assim, por exemplo, o número de iterações utilizadas foi o dobro do maior número de iterações gastas pelo novo algoritmo genético para chegar a uma boa solução, garantindo uma boa margem de segurança para fuga de ótimos locais. Os parâmetros relativos ao

crossover e a mutação são aqueles que produziram, na média, as melhores soluções para as partições 2, 25 e 49 das duas primeiras instâncias.

Após a execução desses testes, observou-se que seria conveniente definir o número de iterações como $T^* = 13N$, em que *N* é o número de ativos disponíveis para investimento. Este valor é bastante inferior àquele empregado pelo método de Chang *et al.*, que é $T^* = 1000N$. Essa diferença se deve ao fato deste último algoritmo trabalhar com apenas um *crossover* e uma mutação por iteração.

O parâmetro *PerCross* (percentual da população que passará pelo *crossover*) foi definido como 60%. Atribuiu-se o valor de 50% à probabilidade de ocorrência de *crossover* na parte 1 do cromossomo, definida através de *ProbCross*.

O parâmetro *PerMut* (percentual da população que será selecionada para mutação) utilizado foi de 60%. Já a probabilidade de ocorrência de mutação na parte 1 dos cromossomos foi definida como 50%, ou seja, *ProbMut* = 0,5. Aos parâmetros *PerMutP*1 e *PerMutP*2 (percentual de bits e de ativos da parte 1 e 2 do cromossomo selecionado que passarão pela mutação) atribuiu-se o valor de 5%. Quando a parte 2 do cromossomo é selecionada para mutação, a probabilidade de ocorrência de mutação do tipo substituição de ativos, *ProbMutP*2, é de 10%. Conseqüentemente, a probabilidade de haver permutação de ativos é igual a 90%.

O número de hipercubos utilizados para a discretização do espaço foi 2^{225} (*n* = 10 e *d* = 25), um valor acima de 10^{67}. Apesar desse número extrapolar a precisão da máquina, trabalhamos com o método de localização de hipercubos por camadas (como explicado no capítulo 6), de modo que o maior valor numérico que tratamos explicitamente é $k_{\max} = 268435455$ (vide subseção 6.3.1).

O tamanho das arestas dos hipercubos, dado pela fórmula (4.1), é igual a $a = 8{,}16684317.10^{-9}$. Obtemos, assim, um erro de discretização de $\xi_D = 3{,}33409976.10^{-9}$, o que corresponde a uma variação de proporção de investimento da ordem de $3{,}334.10^{-7}$ %.

A população inicial utilizada pelo novo algoritmo foi a mesma empregada por Chang *et al.,* ou seja, 100 indivíduos gerados aleatoriamente.

7.5 COMPARAÇÃO COM O ALGORITMO DE CHANG *ET ALII*

Apresentamos, nesta seção, as fronteiras eficientes obtidas tanto pelo método de Chang *et al.* quanto pelo novo algoritmo genético, para as cinco instâncias mencionadas anteriormente. Para traçarmos a fronteira eficiente, resolvemos o modelo (1.8) para 50 valores de λ igualmente espaçados no intervalo $[0,1]$. A cada um desses valores daremos o nome de partição.

Durante a resolução do modelo (1.8) para cada valor de λ, os dois algoritmos (aquele proposto por Chang *et al.* e o novo algoritmo genético) armazenam em uma matriz *H* todas as soluções que resultem na melhora do valor da função objetivo. No final de cada partição, a melhor solução obtida, para um dado valor de λ, é armazenada em uma matriz *V*. Os pontos guardados nessas duas matrizes são usados na construção da fronteira eficiente de investimento.

7.5.1 Primeira instância - Hang Seng.

Nesta instância, que é composta por 31 ativos, o algoritmo de Chang *et al.* executou 31000 iterações para cada uma das 50 partições, consumindo 5526,347*s*, ou seja, 1*h*32*min*6,347*s*. A fronteira eficiente obtida pelo algoritmo é dada na Figura 39. Já o novo algoritmo utilizou 400 iterações[10], com um tempo computacional de 3678,873*s*, ou seja, 1*h*1*min*18,873*s*. A fronteira eficiente obtida é dada na Figura 40.

[10] $T^* = 13 \cdot N = 13 \cdot 31 \approx 400$, conforme seção 7.4.

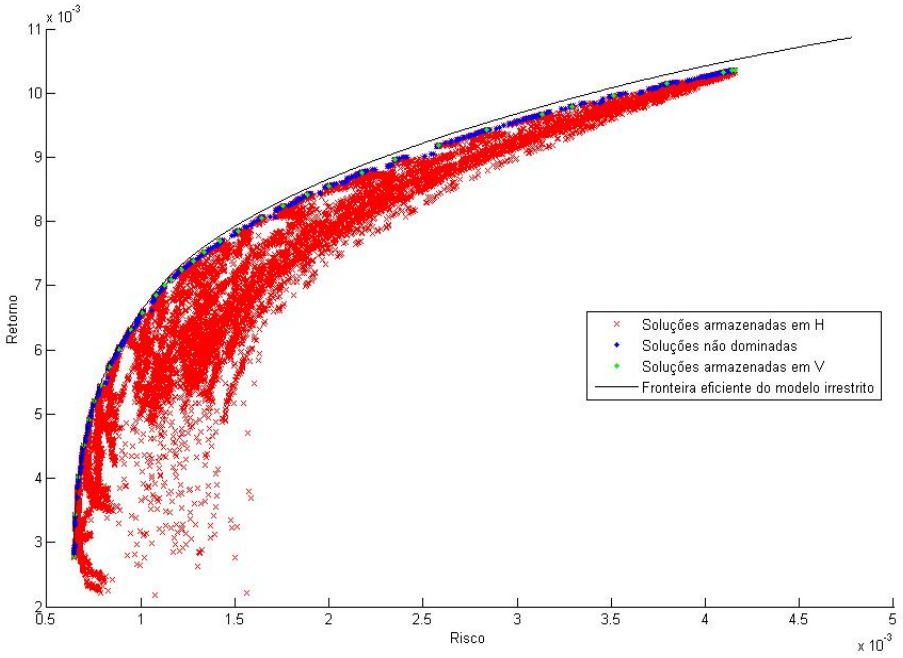


Figura 39 – Solução obtida pelo método de Chang *et al.* para a instância Hang Seng.

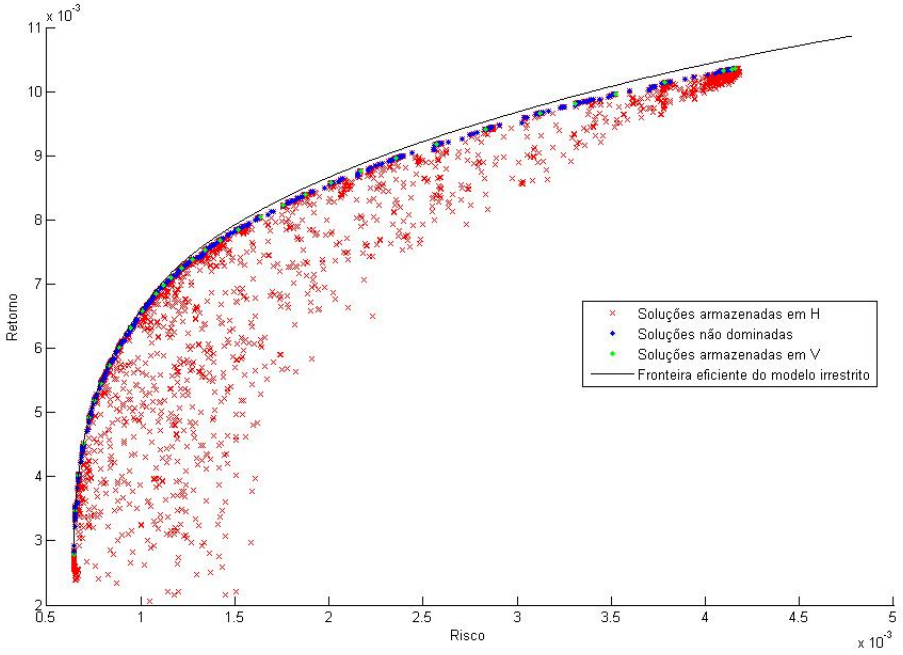


Figura 40 – Solução obtida pelo novo algoritmo genético para a instância Hang Seng.

A linha contínua nas figuras 39 e 40 representa a fronteira eficiente para o modelo (1.5), que não inclui as restrições de cardinalidade e de limite inferior. Os pontos em vermelho representam todas as soluções armazenadas em *H*. Desses pontos, os que não são dominados são apresentados em azul. Alguns destes pontos são do tipo invisível (veja seção 1.6), o que os torna úteis na construção da fronteira eficiente restrita.

Os pontos verdes são as soluções do algoritmo de otimização para os 50 valores de λ. Pode-se observar que estes pontos, sozinhos, não seriam suficientes para a construção da fronteira eficiente de investimento restrita. Daí a grande utilidade do algoritmo meta-heurístico, que transita por diversas soluções antes de chegar à melhor solução.

7.5.1.1 Análise dos resultados obtidos pelas duas propostas – Instância Hang-Seng.

Naturalmente, não seria correto afirmar que o algoritmo de Chang *et al.* executa um número de iterações maior que o novo algoritmo (1000*N* contra 13*N*), uma vez que aquele algoritmo realiza apenas um *crossover* e uma mutação por iteração, enquanto o novo algoritmo genético utiliza 60% da população no *crossover* e na mutação, ou seja, realiza 30 de cada um destes procedimentos por iteração (veja seção 7.4). Mais adequado seria comparar as duas propostas pelo número de operações de *crossover* e mutação realizadas. Assim, como o número de iterações sugeridas por Chang *et al.* para seu algoritmo é $T^* = 1000N$ e o número de iterações sugeridas para o novo algoritmo é $T^* = 13N$, a razão entre o número de procedimentos de *crossover* e mutação executados pelo algoritmo de Chang *et al.* e pelo novo algoritmo é $1000 \cdot N/(13 \cdot 30 \cdot N) \approx 2{,}6$, ou seja, a proposta de Chang *et al.* realiza um número 2,6 vezes superior de procedimentos de *crossover* e mutação que o novo algoritmo.

No tocante ao tempo de execução, o novo algoritmo genético mostrou-se mais eficiente, tendo sido cerca de 33% mais rápido que o algoritmo de Chang *et al.* Além disso, o novo algoritmo também obteve, em geral, as melhores soluções, conforme mostra o Gráfico 1. Neste gráfico, as barras azuis correspondem às partições em que o novo algoritmo genético

encontrou uma solução melhor, e as barras vermelhas indicam as partições em que o algoritmo de Chang *et al.* foi melhor. O eixo das ordenadas apresenta, na forma percentual, o quanto uma solução foi superior à outra.

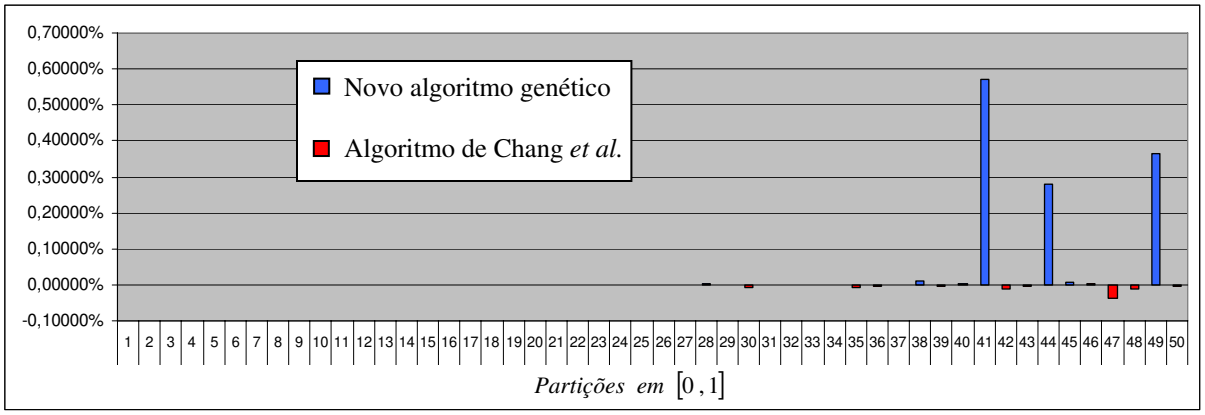


Gráfico 1 – Comparação dos resultados das duas propostas (instância Hang Seng).

Pode-se observar que, para a maioria dos valores de λ, os dois algoritmos alcançaram a mesma solução. Porém, nas últimas partições, o novo algoritmo genético foi claramente superior, apesar de haver casos em que a solução foi levemente pior.

7.5.2 Segunda instância - DAX.

Nesta instância, que é composta por 85 ativos, o algoritmo de Chang *et al.* efetuou 85000 iterações para cada uma das 50 partições. O tempo computacional foi de 16121,231s, ou seja, 4h28min41,231s. A fronteira eficiente obtida pelo algoritmo é dada na Figura 41.

Já o novo algoritmo genético executou 1100 iterações para resolver esta instância, com um tempo computacional de 10030,23s, ou seja, 2h47min10,23s. A fronteira eficiente obtida por este algoritmo é dada na Figura 42.

7.5.2.1 Análise dos resultados obtidos pelas duas propostas – Instância DAX.

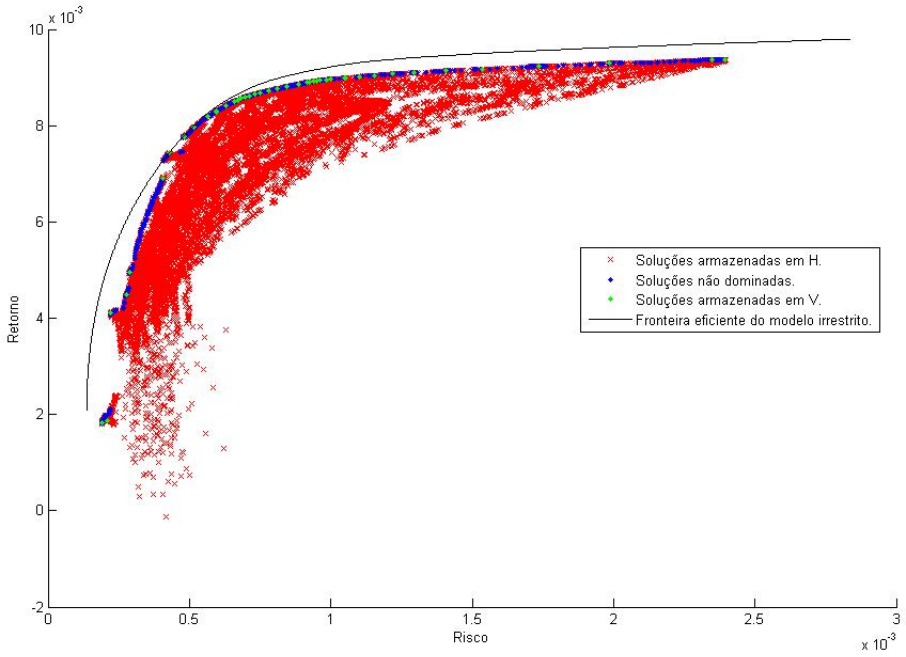


Figura 41 – Solução obtida pelo método de Chang *et al.* para a instância DAX.

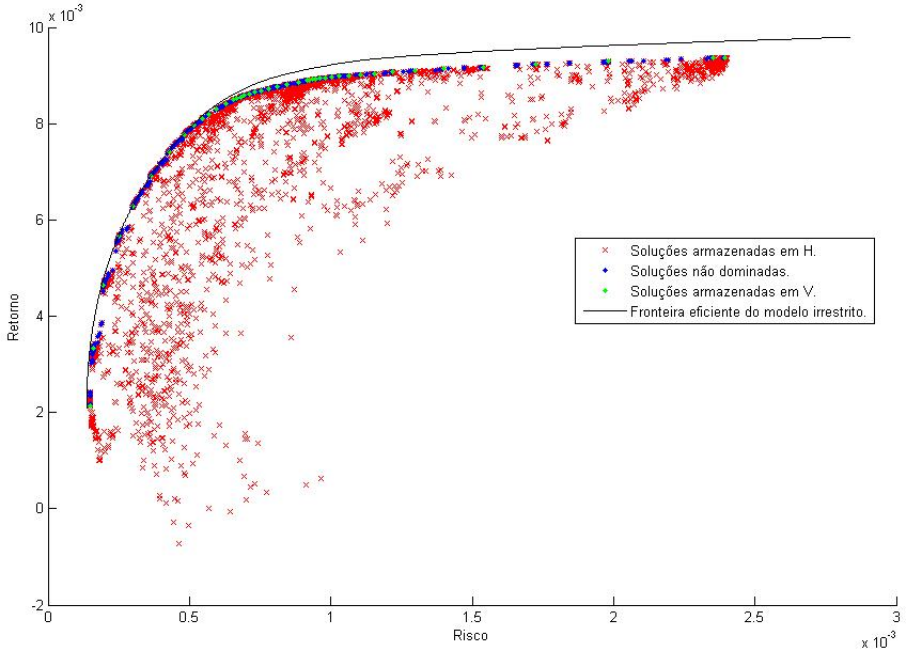


Figura 42 – Solução obtida pelo novo algoritmo genético para a instância DAX.

O novo algoritmo genético também se mostrou mais eficiente no tocante ao tempo de execução, tendo sido cerca de 38% mais rápido que o algoritmo de Chang *et al.* Com relação às soluções obtidas, o novo método também foi superior, como ilustra o Gráfico 2.

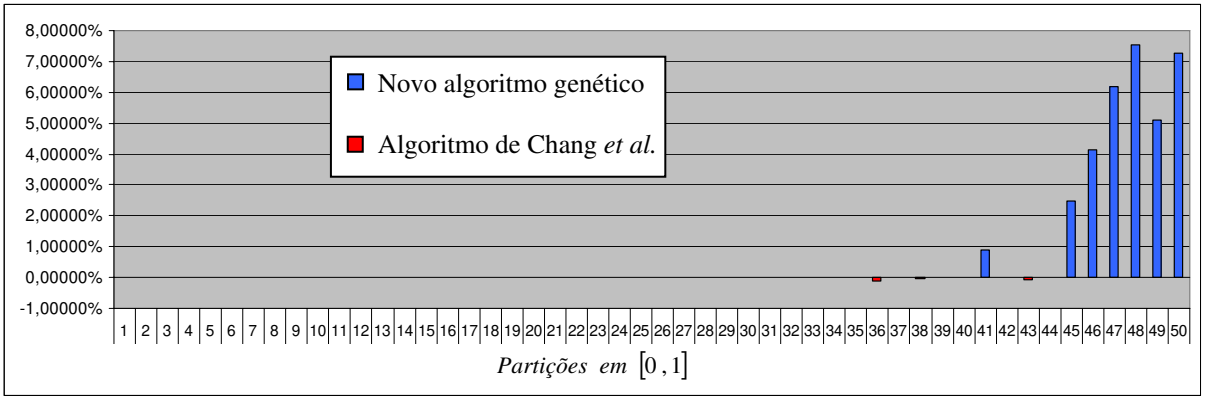


Gráfico 2 – Comparação dos resultados das duas propostas (instância DAX).

Observa-se, neste gráfico, que os dois algoritmos obtiveram a mesma solução para a maioria dos valores de λ. Porém, à semelhança do ocorrido na primeira instância, o novo algoritmo genético gerou soluções consideravelmente melhores nas últimas partições, sugerindo que o algoritmo de Chang *et al.* encontra dificuldades na resolução do modelo (1.8) para valores de λ próximos de 1.

7.5.3 Terceira instância – FTSE.

Nesta instância, que é composta por 89 ativos provenientes do Reino Unido, o número de iterações utilizadas pelo algoritmo de Chang *et al.* foi de 89000 para cada uma das 50 partições. Seu tempo computacional foi de 17279,387s, ou 4h47min59,387s. Já o novo algoritmo genético utilizou 1200 iterações, com um tempo computacional de 10856,248s, ou 3h56,248s. As fronteiras eficientes obtidas pelos algoritmos são dadas nas figuras 43 e 44.

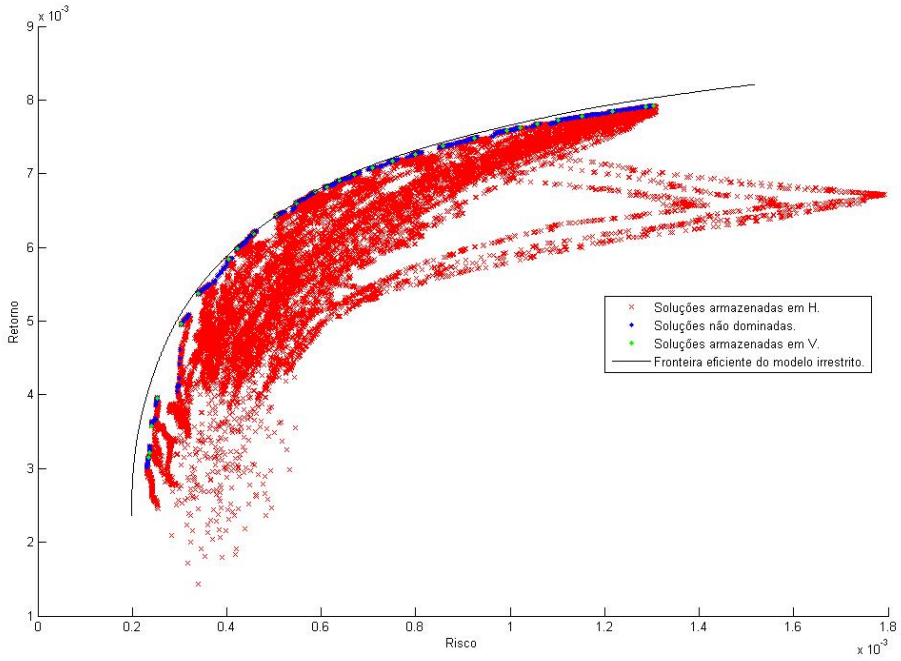


Figura 43 – Solução obtida pelo método de Chang *et al.* para a instância FTSE.

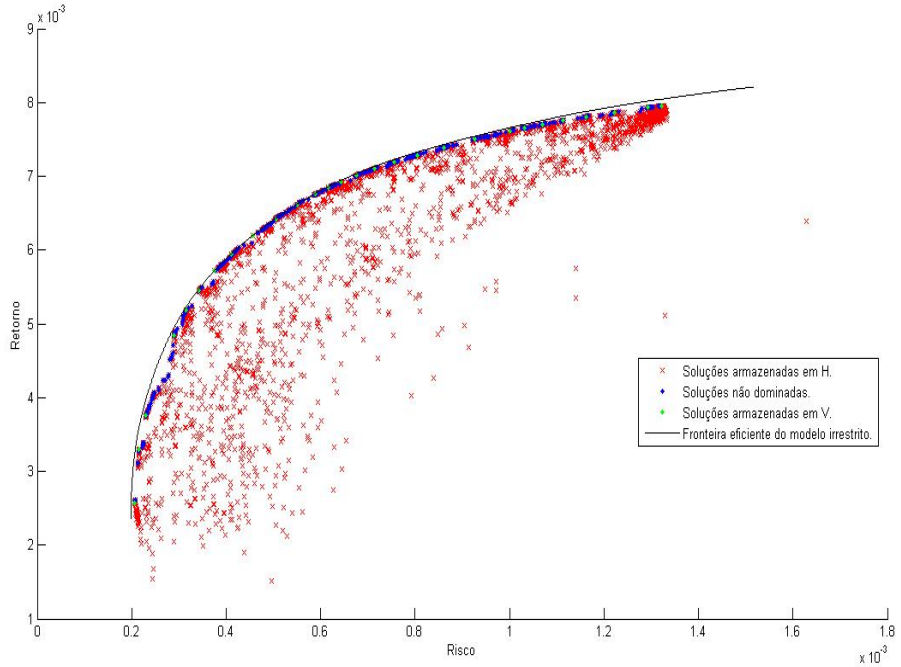


Figura 44 – Solução obtida pelo novo algoritmo genético para a instância FTSE.

7.5.3.1 Análise dos resultados obtidos pelas duas propostas – Instância FTSE.

O tempo computacional do novo algoritmo genético foi 37% menor que o do algoritmo de Chang *et al.* O Gráfico 3 ilustra as partições em que cada um dos algoritmos obteve a melhor solução.

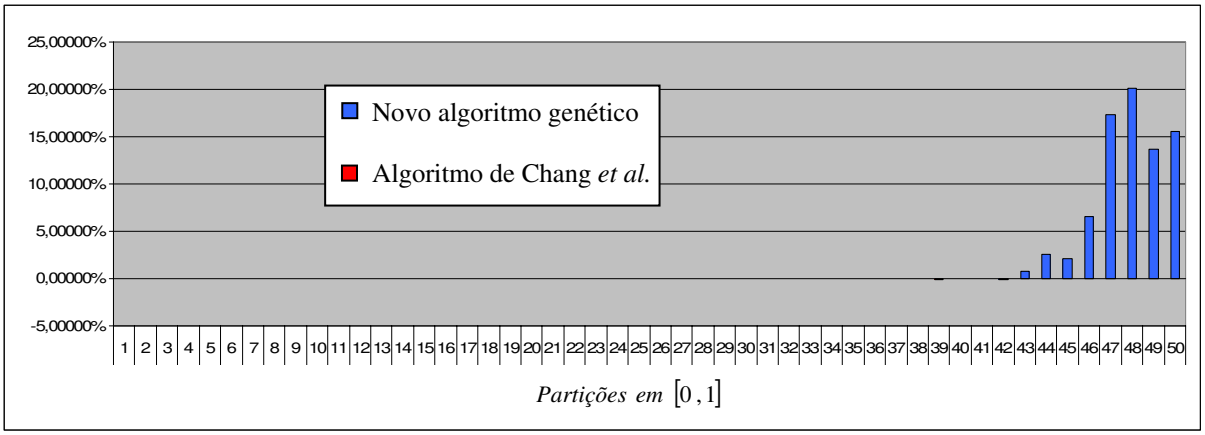


Gráfico 3 – Comparação dos resultados das duas propostas (instância FTSE).

Assim como aconteceu nas instâncias anteriores, o novo algoritmo obteve soluções melhores nas últimas partições, chegando a diferença a atingir 20% em alguns casos.

7.5.4 Quarta instância – S&P.

Na quarta instância, que é composta por 98 ativos, o algoritmo de Chang *et al.* consumiu 98000 iterações para cada uma das 50 partições. Seu tempo de execução foi de 18984,681*s*, ou 5*h*16*min*24,681*s*. Já o novo algoritmo genético utilizou 1300 iterações, com um tempo computacional de 12264,265*s*, o equivalente a 3*h*24*min*24,265*s*. As fronteiras eficientes obtidas são apresentadas nas figuras 45 e 46 a seguir.

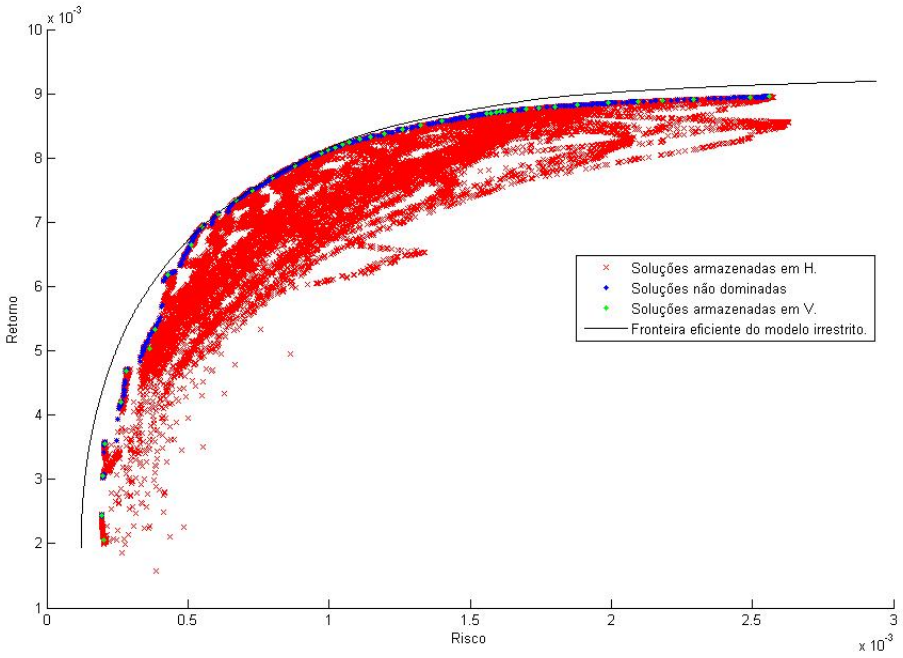


Figura 45 – Solução obtida pelo método de Chang *et al.* para a instância S&P.

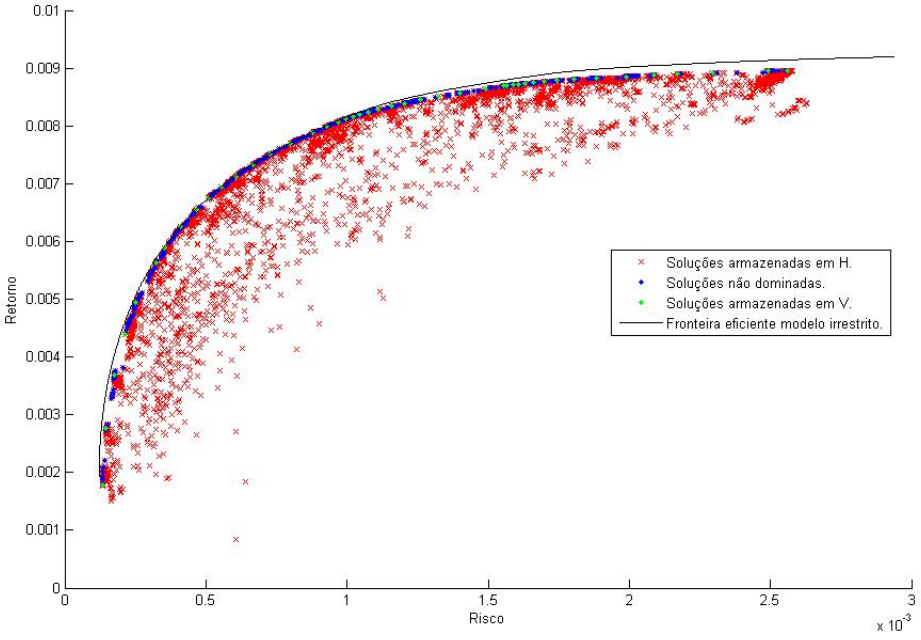


Figura 46 – Solução obtida pelo novo algoritmo genético para a instância S&P.

7.5.4.1 Análise dos resultados obtidos pelas duas propostas – Instância S&P.

Nesta instância o tempo computacional do novo algoritmo genético foi cerca de 35% menor que o do algoritmo proposto por Chang *et al.*

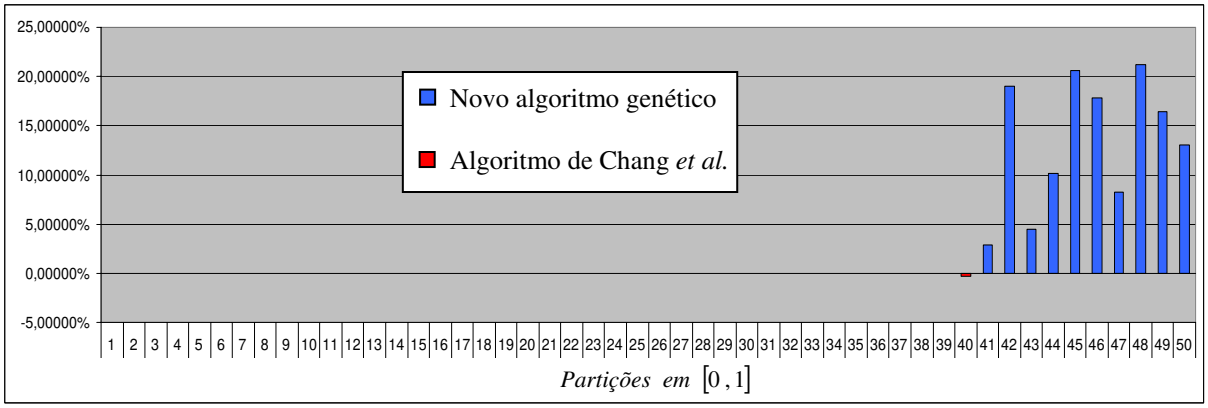


Gráfico 4 – Comparação dos resultados das duas propostas (instância S&P).

A diferença entre as soluções obtidas pelos algoritmos também se concentra nas últimas partições. Novamente, observa-se a dificuldade do algoritmo de Chang *et al.* ao lidar com o modelo (1.8) para valores de λ próximos de 1.

7.5.5 Quinta instância – Nikkei.

Esta última instância é a maior de todas, possuindo um universo de 225 ativos disponíveis para investimento. Para resolvê-la, o algoritmo de Chang *et al.* executou 225000 iterações para cada uma das 50 partições, gastando para isso 57415,838*s*, ou 15*h*56*min*55,838*s*. O novo algoritmo genético utilizou apenas 3000 iterações, com um tempo computacional de 27612,633*s*, ou 7*h*40*min*12,633*s*. As fronteiras eficientes obtidas pelos algoritmos são dadas nas figuras 47 e 48 a seguir.

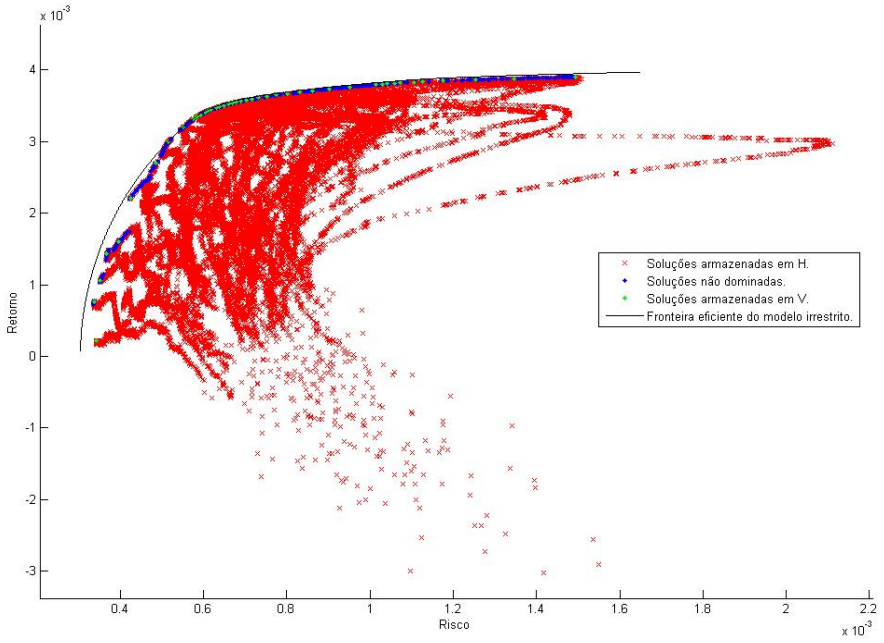


Figura 47 – Solução obtida pelo método de Chang *et al.* para a instância Nikkei.

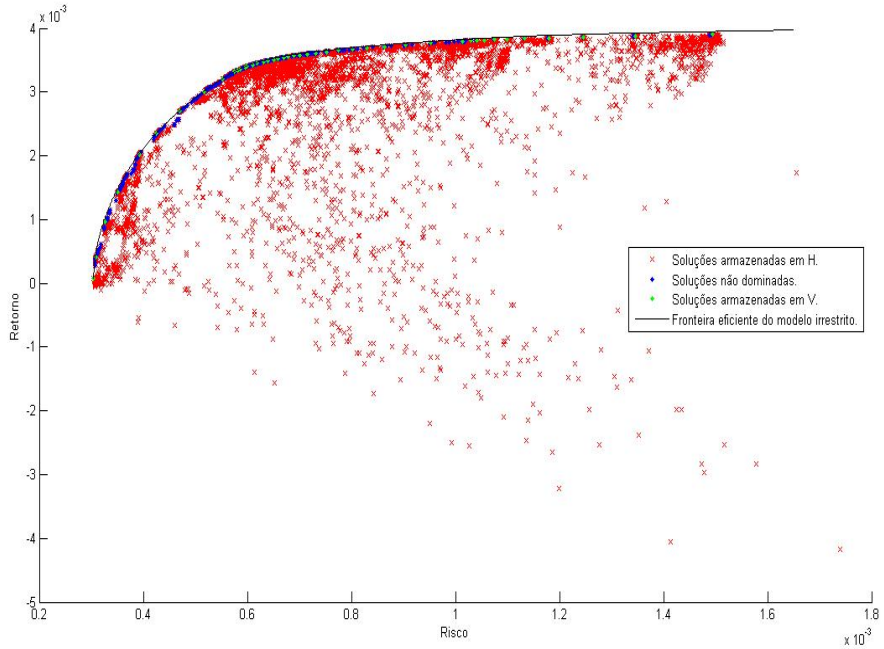


Figura 48 – Solução obtida pelo novo algoritmo genético para a instância Nikkei.

7.5.5.1 Análise dos resultados obtidos pelas duas propostas – Instância Nikkei.

A diferença entre os tempos computacionais das duas propostas foi mais visível nesta instância, tendo o novo algoritmo atingido uma redução superior a 50%. Já o padrão de qualidade das soluções foi mantido, ou seja, o novo algoritmo obteve soluções melhores para valores λ próximos de 1. O Gráfico 5 ilustra as partições em que cada um dos algoritmos foi superior.

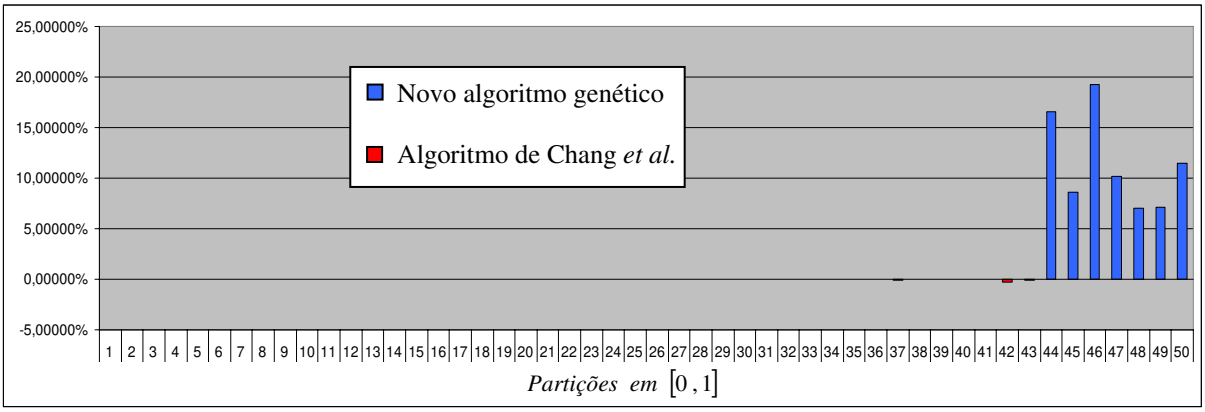


Gráfico 5 – Comparação dos resultados das duas propostas (instância Nikkei).

Pode-se observar no Gráfico 5 que, apesar de apresentar algumas soluções levemente piores para as partições 37, 42 e 43, com diferença menor que 1%, o novo algoritmo genético foi significativamente superior a partir da partição 44, chegando a obter soluções 15% melhores que aquelas alcançadas por Chang *et al*.

7.6 A QUALIDADE DA SOLUÇÃO ENCONTRADA

Com intuito de verificar a qualidade das soluções obtidas pelo novo algoritmo genético, comparamos estas com aquelas obtidas por Villela [19], que utilizou um método exato

baseado na estratégia *branch and bound* para resolver o modelo de otimização de carteiras de investimento. Entretanto, o modelo de otimização utilizado por Villela não é exatamente igual ao utilizado neste trabalho, definido em (1.8), pois ele trata as restrições de capital investido e de cardinalidade (tamanho da carteira) como desigualdades. Mais especificamente, o modelo de otimização utilizado por Villela é dado por

$$
\begin{aligned}
\min_{w} \quad & \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1-\lambda) \sum_{i=1}^{N} w_i \mu_i \\
s.a \quad & \sum_{i=1}^{N} w_i \leq 1 \\
& \sum_{i=1}^{N} z_i \leq K \\
& w_i \geq \varepsilon_i z_i, \qquad i = 1, \ldots, N \\
& z_i \in \{0, 1\}, \quad i = 1, \ldots, N.
\end{aligned}
$$

Com a inclusão das desigualdades, o algoritmo de Villela retornou soluções com carteiras com menos de 10 ativos, valor fixo adotado tanto pelo algoritmo de Chang *et al.* como pelo novo algoritmo genético na resolução das cincos instâncias clássicas. Assim, para possibilitar a comparação entre os dois algoritmos, executamos o novo algoritmo genético com o mesmo tamanho de carteira que aquele retornado pelo método exato.

Para tanto, implementamos em Matlab um programa que executa, para cada valor de λ, o algoritmo de Villela e, em seguida, utiliza o valor de *K* obtido por este para executar o novo algoritmo genético.

Contudo, nem todas as soluções puderam ser comparadas, pois a restrição de capital investido usada por Villela também é de desigualdade ($\sum_{i=1}^{N} w_i \leq 1$) e o novo algoritmo genético trabalha com a restrição de igualdade, o que significa, em outras palavras, que ele exige que se invista todo o capital disponível. Deste modo, não comparamos os dois algoritmos para os valores de λ em que o capital investido sugerido pelo método exato era inferior a 100% do montante disponível. Como, para algumas instâncias, o método exato retornou soluções em que o

capital investido é menor que 100% a partir de λ = 0,75510, comparamos os dois métodos somente para as partições menores que 38 (ou seja, para λ<0,75).

Além disso, quando os valores de λ são inferiores a 0,5, o algoritmo de Villela retorna, muitas vezes, soluções que têm poucos ativos. Neste caso, as soluções ótimas são fáceis de se obter e quase não há diferença entre as soluções obtidas pelo algoritmo genético e pelo algoritmo exato. Como exemplo, quando comparamos os resultados obtidos pelos dois algoritmos (novo algoritmo genético e algoritmo exato), nas cinco instâncias, para valores de λ menores que 0,5, a maior diferença percentual obtida foi de 0,00070695%. Por esse motivo, os valores de λ utilizados para comparar os dois algoritmos são aqueles maiores que 0,5 e menores que 0,75, que correspondem às partições 26 a 37.

O Gráfico 6 mostra as diferenças percentuais entre o novo algoritmo genético e o método exato nas cinco instâncias clássicas consideradas (Hang Seng, DAX, FTSE, S&P e Nikkei) para os valores de λ selecionados.

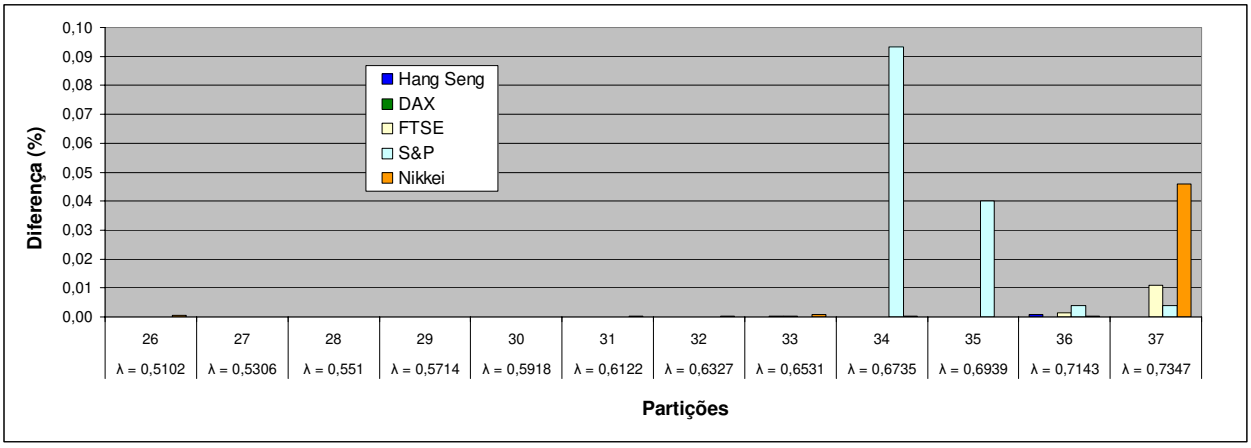


Gráfico 6 – Diferença percentual entre o método exato e o novo algoritmo para as cinco instâncias clássicas.

Pode-se observar no Gráfico 6 que a diferença entre soluções dos dois algoritmos é muito pequena, não ultrapassando 0,094% no pior caso. As maiores diferenças ocorreram nas partições 34, 35 e 37, principalmente na instância S&P. A Tabela 2 mostra a diferença média entre as soluções do método exato e do novo algoritmo genético.

Tabela 2 – Diferença média entre o método exato e o novo algoritmo genético.

Instância	Diferença Média
Hang Seng	0,0000789%
DAX	0,0000184%
FTSE	0,0003530%
S&P	0,0117960%
Nikkei	0,0040297%

A Tabela 2 confirma a qualidade das soluções encontradas pelo novo algoritmo genético para as cinco instâncias analisadas, já que a maior diferença média encontrada (correspondente à instância S&P) não passa de 0,0118%.

7.7 A INFLUÊNCIA DO TAMANHO DA CARTEIRA

Nesta seção, analisamos o comportamento do novo algoritmo genético e do algoritmo de Chang *et al.* a partir da variação do tamanho da carteira de investimento. Para tanto, executamos os dois algoritmos atribuindo os valores 5, 10, 20, 30 e 40 para *K*. Apenas a instância S&P foi utilizada nestes testes. Foram mantidas as restrições de limite mínimo de investimento ($\varepsilon_i = 0,01$).

Novamente, a superioridade do novo algoritmo genético concentrou-se nos últimos valores de λ, a partir da partição 30. Para as partições menores que 30 os dois algoritmos obtiveram praticamente as mesmas soluções. O Gráfico 8 ilustra as partições em que cada um dos algoritmos foi superior. Neste gráfico, as barras acima do eixo das abscissas mostram as partições em que o algoritmo genético foi superior e as abaixo deste eixo mostram aquelas em que o algoritmo de Chang *et al.* foi superior.

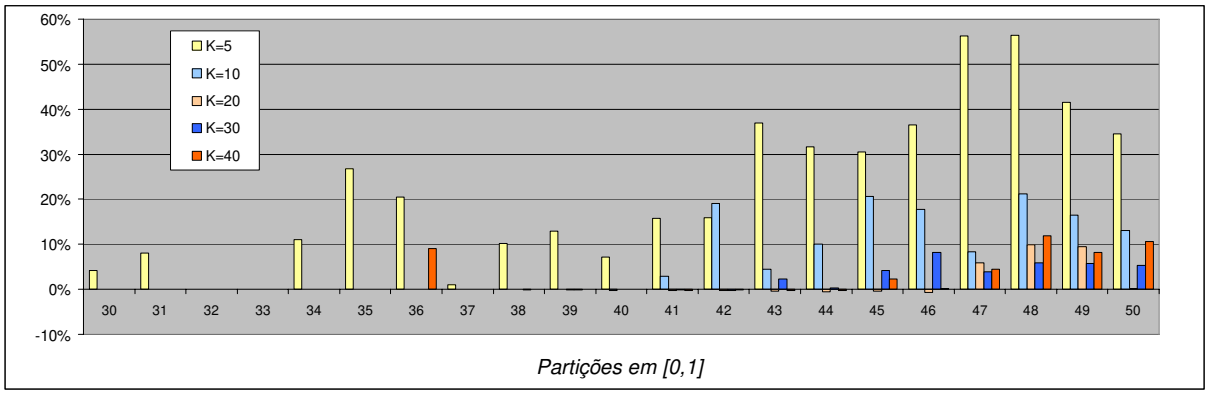


Gráfico 7 – Comparação entre o algoritmo de Chang *et al.* e o novo algoritmo para K=5, 10, 20, 30 e 40.

Observa-se, neste gráfico, que as melhoras mais intensas ocorreram para o tamanho de carteira K=5, passando, em algumas partições, de 50%. Além disso, para este tamanho de carteira, as melhoras se estenderam por mais partições, iniciando na partição 30. Para K=20 as melhoras ocorreram apenas em três partições (47, 48 e 49), chegando a 10%. A carteira de tamanho 30 foi a que gerou as menores melhoras percentuais, não passando de 8,5%.

7.8 A INFLUÊNCIA DA POPULAÇÃO INICIAL

Para verificar os efeitos da geração de uma população inicial uniforme no algoritmo de Chang *et al.*, introduzimos neste o gerador de população inicial do novo algoritmo genético (que, como verificado na seção 4.5, gera uma distribuição uniforme da população). Assim, a população inicial do algoritmo de Chang *et al.* foi criada com base na discretização do espaço de soluções por meio de hipercubos, conforme explicado no capítulo 4. A instância utilizada nos testes foi a S&P, que é composta por 98 ativos. Na resolução desta instância utilizamos a restrição de cardinalidade $K = 10$ e $\varepsilon_i = 0,01$ como limites mínimos de investimento.

A fronteira eficiente obtida pelo algoritmo de Chang *et al.* com a utilização da população inicial uniforme é apresentada Figura 49. Para efeito de comparação, a Figura 50

apresenta a fronteira eficiente obtida pelo método original de Chang *et al.*, figura esta que foi apresentada na subseção 7.5.4 como na Figura 45.

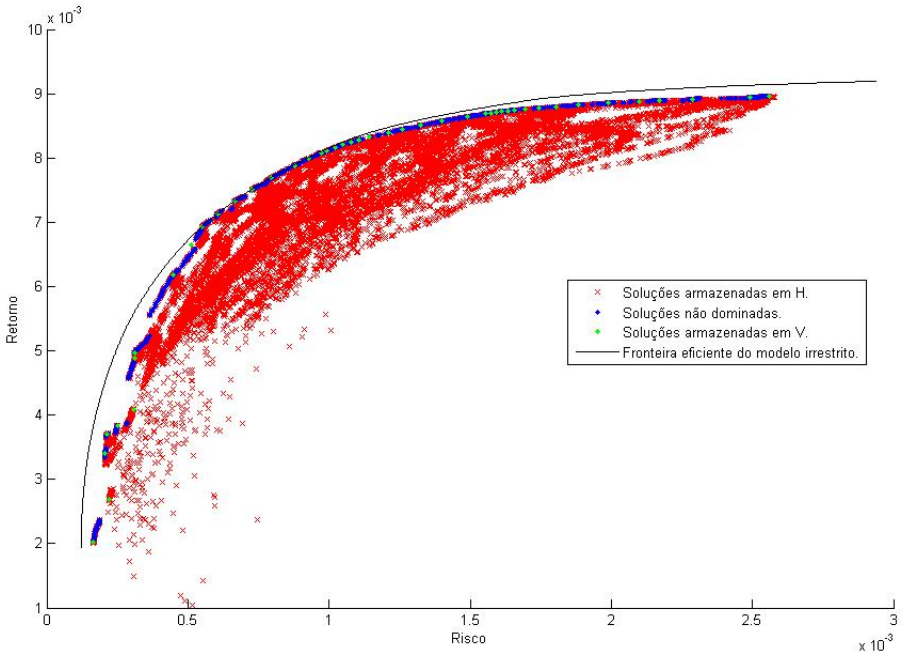


Figura 49 – Solução obtida pelo método de Chang *et al.* com população inicial uniforme na instância S&P.

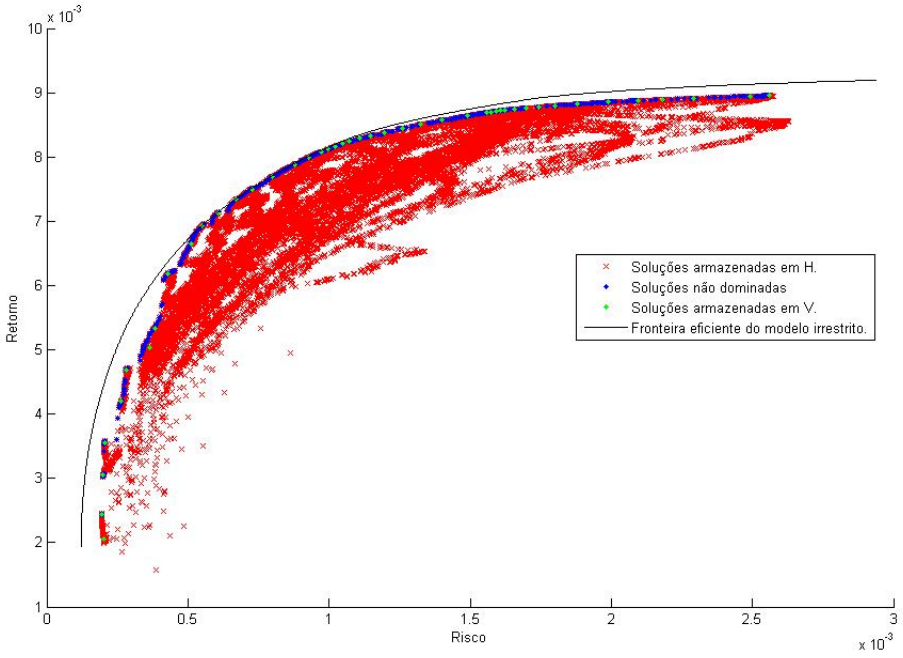


Figura 50 – Solução obtida pelo método original de Chang *et al.* na instância S&P.

Apesar da utilização de uma população inicial uniforme, a melhoria obtida pelo algoritmo de Chang *et al.* foi pequena, como mostra o Gráfico 8. Neste gráfico, as barras de cor laranja correspondem às partições em que o algoritmo de Chang *et al.* com distribuição da população uniforme encontrou uma solução melhor, enquanto as barras vermelhas indicam as partições em que o algoritmo original de Chang *et al.* foi melhor. O eixo das ordenadas apresenta, na forma percentual, o quanto uma solução foi superior à outra.

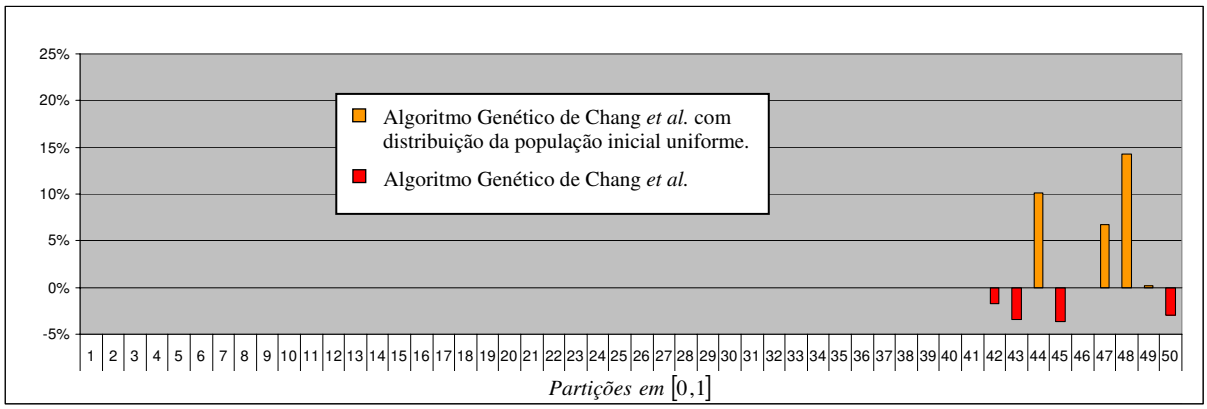


Gráfico 8 – Comparação dos resultados de Chang *et al.* com e sem a distribuição uniforme da população inicial.

O Gráfico 9 compara os resultados obtidos pelo novo algoritmo genético com o algoritmo de Chang *et al.* com distribuição uniforme da população inicial.

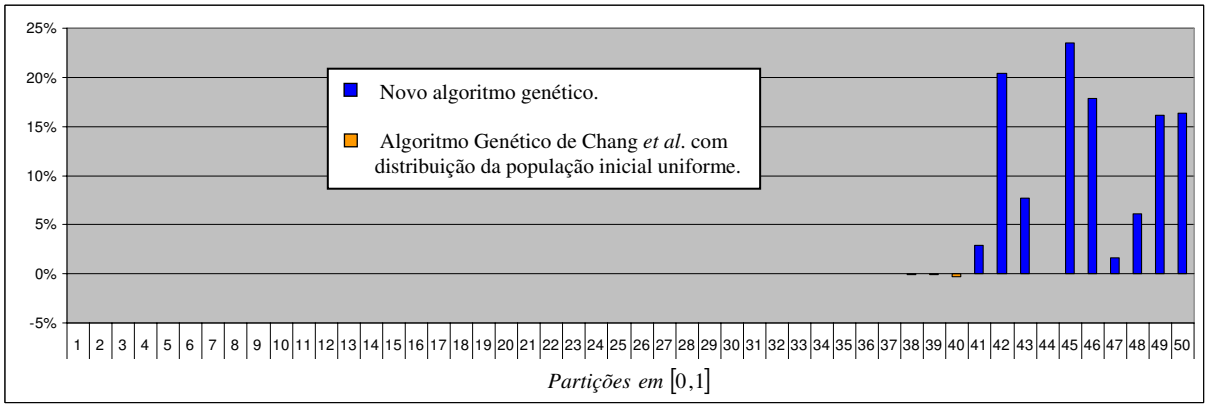


Gráfico 9 – Comparação do novo algoritmo com o algoritmo de Chang *et al.* com população inicial uniforme.

Como se observa, a população inicial não é o único fator que torna o novo algoritmo mais eficiente que o algoritmo de Chang *et al.*

7.9 ANÁLISE DOS RESULTADOS OBTIDOS

Os gráficos mostrados na seção 7.5 deixam evidente a superioridade das soluções obtidas pelo novo algoritmo genético nas últimas partições. Este comportamento decorre do fato de que, para as partições associadas a valores de λ próximos de 1, a redução do risco é priorizada, ou seja, dá-se preferência às carteiras mais diversificadas. A obtenção deste tipo de carteira requer a análise de uma vasta porção da região factível, algo difícil de se obter usando o algoritmo de Chang *et al.*

Na seção 7.6 apesar do Gráfico 6 mostrar que as maiores diferenças percentuais com o método exato ocorreram nas duas maiores instâncias (S&P e Nikkei), seria precoce concluir que houve uma deterioração das soluções do algoritmo genético com o aumento do número de ativos disponíveis para investimento, já que a instância S&P gerou maior divergência com o método exato que a instância Nikkei, mesmo esta sendo 2,3 vezes maior que a S&P.

Na seção 7.7 o Gráfico 7 mostrou que o novo algoritmo genético gera soluções melhores que o algoritmo de Chang *et al.* para outros valores de restrição de tamanho de carteira investimento (5, 20, 30 e 40). Este fato evidencia que o problema da distribuição não uniforme da população inicial do algoritmo de Chang *et al.* também pode prejudicar qualquer problema de otimização de *portfolio* independente da restrição de tamanho de carteira investimento.

Os resultados obtidos na seção 7.8 indicam que a distribuição não uniforme da população não ocorre somente na população inicial, mas em outros procedimentos do algoritmo de Chang *et al.* De fato, como explicado na seção 3.4, a mudança de variável dada pela equação (2.1) faz com que algumas regiões do simplex tenham mais chance de serem selecionadas que

outras, porém o algoritmo de Chang *et al*. utiliza esta mudança não apenas na geração da população inicial, mas também em procedimentos como *crossover* e mutação.

Como conseqüência de sua melhor cobertura da região factível, o novo algoritmo apresentou uma fronteira eficiente mais próxima daquela relativa ao modelo sem restrição de cardinalidade. Além disso, sua fronteira também é mais completa, pois apresenta um número maior de pontos azuis na parte mais curva do gráfico.

Observa-se também que o algoritmo de Chang *et al.* gasta um número 2,6 vezes maior de operações de *crossover* e mutação para chegar a soluções que, na melhor das hipóteses, são muito próximas daquelas obtidas pelo novo algoritmo genético. Por esse motivo, o tempo computacional do novo algoritmo foi, em média, 38% menor para as cinco instâncias analisadas. De um modo geral, o novo algoritmo genético apresentou, na resolução das cincos instâncias propostas, um crescimento aproximadamente linear do gasto de tempo computacional com o aumento do universo de ativos disponíveis para investimento. O Gráfico 10 compara o crescimento do tempo computacional na execução dos dois algoritmos.

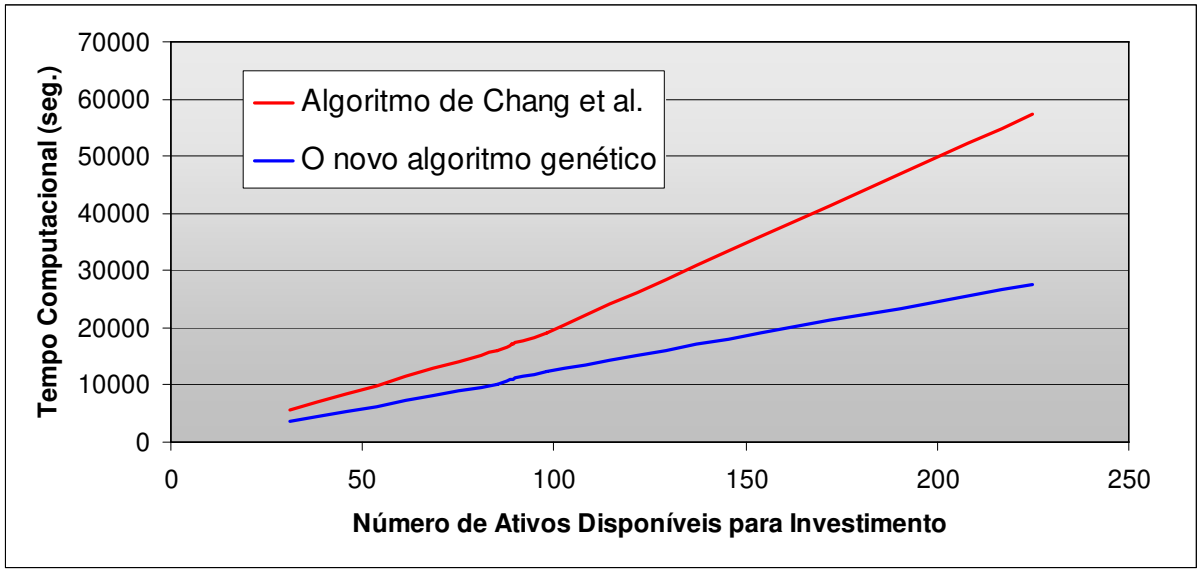


Gráfico 10 – Crescimento do tempo computacional em função do número de ativos disponíveis para investimento.

7.10 UMA TENTATIVA DE MELHORA

Em [6], di Gaspero *et al.* propõem um algoritmo híbrido de otimização, combinando uma meta-heurística com um algoritmo exato de programação quadrática. A meta-heurística é usada somente na determinação dos valores binários z_i do modelo (1.8), que indicam os ativos que farão parte da carteira, ficando a cargo do algoritmo quadrático a determinação dos valores das proporções de investimento w_i. Assim, para cada cromossomo da população, os valores z_i são fixados e apenas as variáveis w_i associadas a $z_i = 1$ são consideradas na otimização.

No novo algoritmo genético, esta idéia foi adaptada de modo que o algoritmo quadrático fosse aplicado apenas uma vez por partição, partindo da melhor solução obtida pelo algoritmo genético. Assim, após o final da execução do algoritmo genético, os valores de z_i da melhor solução foram fixados e aplicou-se um método de programação quadrática para obter o vetor *w*.

Apesar da utilização deste método exato ter trazido alguma melhora às soluções obtidas, esta foi pouco significativa. A Tabela 3 apresenta, para as cinco instâncias analisadas, a média de melhora das soluções e o desvio padrão obtidos com o emprego do algoritmo de programação quadrática.

Tabela 3 – Média de melhora e desvio padrão para as cinco instâncias apresentadas.

Instância	**Média**	**Desvio Padrão**
Hang Seng	0,0033%	0,0075%
DAX	0,0117%	0,0531%
FTSE	0,0263%	0,0638%
S&P	0,0304%	0,0705%
Nikkei	0,0248%	0,0939%

A tabela deixa claro que a melhoria obtida realmente foi modesta. Para algumas partições, como aquela em que $\lambda = 1$ na instância Hang Seng, a diferença foi menor que 0,000005%. Em casos como esse, a diferença encontrada não está associada a uma má distribuição dos recursos entre os ativos, mas apenas ao erro de aproximação cometido pelo algoritmo genético devido à discretização do espaço.

CONCLUSÕES

Neste trabalho, analisamos os problemas relativos à determinação da fronteira eficiente restrita, incluindo o tratamento dos pontos ditos "invisíveis". Discutimos, também, a proposta de Chang *et al.* [3], destacando seus pontos fracos. Com base na dificuldade apresentada pelo método de Chang *et al.* na determinação dos pontos da porção extrema esquerda da fronteira eficiente (a porção associada à priorização da redução do risco), propusemos uma nova estratégia de discretização do espaço de soluções, juntamente com um novo algoritmo genético para determinação da fronteira eficiente restrita.

As principais características do novo algoritmo genético são a distribuição uniforme da população inicial e a utilização de operadores clássicos de mutação e *crossover* sobre vetores binários.

Os resultados computacionais apresentados para cinco instancias do problema de otimização de carteiras de investimento indicam que a nova proposta é bastante promissora, já que forneceu soluções melhores que o algoritmo de Chang *et al.*, consumindo menos tempo.

Tentando reduzir o efeito da discretização sobre as proporções de investimento obtidas, aplicamos um algoritmo de programação quadrática ao final da execução do algoritmo genético. Esta modificação, entretanto, provocou uma melhoria pouco significativa da função objetivo, confirmando a eficiência do algoritmo genético proposto.

Como possíveis desdobramentos do trabalho ora desenvolvido, destacamos:

- A incorporação de algumas estratégias usuais para a aceleração da convergência do algoritmo genético, tais como a inclusão, na população inicial, de cromossomos de boa qualidade e a aplicação de um método de busca local a cada iteração do algoritmo genético.
- A implementação do novo algoritmo genético em uma linguagem como C ou C++, o que tornaria possível a resolução de problemas de grande porte, além de permitir o uso da discretização do espaço de soluções proposta no capítulo 4, na qual se associa a cada hipercubo um único número inteiro grande.
- O desenvolvimento de um algoritmo genético para a resolução do modelo clássico de média-variância (1.4) com as restrições de cardinalidade e limite inferior, como fazem di Gaspero *et al.* [6] e Schaerf [18]. Segundo di Gaspero *et al.*, este modelo propicia uma distribuição dos pontos da fronteira eficiente melhor que aquela obtida a partir do modelo parametrizado.

REFERÊNCIAS

[1] BÄCK T.; FOGEL D. B.; MICHALEWICZ, Z. *Handbook of Evolutionary Computation***.** Philadelphia: IOP, 1997.

[2] BRINDLE, A. *Genetic Algorithms for Function Optimization*. Tese de doutorado, Department of Computer Science, University of Alberta, Alberta, Canada, 1981.

[3] CHANG, T. J.; MEADE, N.; BEASLEY, J. E.; SHARAIHA, Y. M. Heuristics for cardinality constrained portfolio optimization. *Computer and Operations Research,* 27, p. 1271–1302, 2000.

[4] DARWIN, C. *The Origin of Species*. London: Penguin, 1985.

[5] DAVIS, L. Applying Adaptive Algorithms to Epistatic Domains. *Proceedings of the International Joint Conference on Artificial Intelligence*, p. 136-140, 1985.

[6] DI GASPERO, L.; DI TOLLO, G.; ROLI, A.; SCHAERF A. A hybrid solver for constrained portfolio selection problems. Em *Proceedings of Learning and Intelligent Optimization (LION2007)*, 2007.

[7] HOLLAND, J.H. – *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan, 1975.

[8] KELLERER, H. & MARINGER, D. Optimization of cardinality constrained portfolios with an hybrid local search algorithm. Em: *Annals of the 4th Metaheuristics International Conference – MIC'2001*, 2001.

[9] KONNO, H. Piecewise linear risk-function and portfolio optimization. *Journal of the Operations Research Society of Japan*, 33, p. 139-156, 1990.

[10] KONNO, H. & SUZUKI, K. I. A mean-variance-skewness portfolio optimization model. *Journal of the Operations Research Society of Japan*, 38, p. 173-187, 1995.

[11] LUENBERGER, D. G. *Investment Science*. New York: Oxford University, 1998.

[12] MARKOWITZ, H. Potfolio Selection. *Journal of Finance*, **7**, p. 77-91, 1952.

[13] MARKOWITZ H.; TODD, P.; XU, G.; YAMANE, Y. Computation of mean-semivariance efficient sets by the critical line algorithm. *Annals of Operations Research*, 45, p. 307-317, 1993.

[14] MICHALEWICZ, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. 3.ed. Berlin: Springer, 1996.

[15] MITRA, G.; KYRIAKIS, T.; LUCAS, C. A review of portfolio planning: Models and systems. Em: *Advances in Portfolio Construction and Implementation.* S.E. Satchell and A.E. Scowcroft (Eds.), Oxford: Butterworth & Heinmann, 2003, p. 1-39.

[16] MORAL-ESCUDERO, R.; RUIZ-TORRUBIANO, R.; SUÁREZ, A. Selection of optimal investment with cardinality constraints. Em: *Annals of the IEEE World Congress on Computational Intelligence*, 2006.

[17] ROLLAND, E. A tabu search method for constrained real-number search: applications to portfolio selection. Technical report, Dep. of Accounting and Management Information Systems, Ohio State University, Columbus, 1997.

[18] SCHAERF, A. Local search techniques for constrained portfolio selection problems. *Computational Economics*, 20, p. 177-190, 2002.

[19] VILLELA, P. F. *Um algoritmo exato para a otimização de carteiras de investimento*. Dissertação de Mestrado em Matemática Aplicada, em desenvolvimento. Departamento de Matemática Aplicada, Universidade de Campinas, Campinas, Brasil, 2008.