UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Um estudo sobre o problema de *Portfolio* de Ações com custos fixos de transação

---

Daniela Arai Yamanaka

São Carlos - SP

Um estudo sobre o problema de *Portfolio* de Ações com custos fixos de transação

Daniela Arai Yamanaka

Orientadora: ***Franklina Maria Bragion de Toledo***

Monografia final de projeto de graduação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP - para obtenção do título de Engenheira da Computação.
Área de concentração: Otimização e modelos estocásticos

USP - São Carlos
Outubro de 2008

Agradecimentos

A USP, por ter fornecido o conhecimento necessário que possibilitou este projeto.

Aos professores, que conseguiram passar seus conhecimentos com empenho.

Resumo

Na área financeira, um dos problemas clássicos é a composição de uma carteira de ações – portfólio. Um portfólio é composto por diferentes ações do mercado distribuídas em diferentes quantidades. Toda carteira tem um retorno e um risco estimado e, em geral, quanto maior o retorno maior o risco a ela associado. O objetivo básico do investidor na seleção de uma carteira de ações é aumentar sua rentabilidade e reduzir o risco a ela associado.

O objetivo principal deste trabalho é estudar a aplicabilidade do modelo proposto em Mansini e Speranza [2005] ao mercado brasileiro. Como objetivo secundário, é proposta a utilização apenas de software livre para o desenvolvimento do projeto. Logo, para avaliar o comportamento do modelo quando aplicado ao mercado brasileiro, o método proposto em Mansini e Speranza [2005] foi implementado em linguagem C utilizando as bibliotecas do GLPK, para resolver os problemas inteiro-mistos resultantes. Além disso, o método proposto para determinar o portfólio ótimo utiliza como entrada dados históricos sobre as ações a serem consideradas. Para obter esses dados, foi desenvolvido um *parser* capaz de interpretar os arquivos históricos das ações disponíveis no *site* da BOVESPA. Testes computacionais mostraram que os métodos implementados são eficientes.

Índice:

Índice de figuras

Índice de tabelas

1. Introdução

Na área financeira, um dos problemas clássicos é a escolha de uma carteira de ações – portfólio, a qual pode ser composta por vários ativos, tais como ações, obrigações e derivados [Konno e Kobayashi, 1997]. O objetivo básico do investidor na seleção de uma carteira de ativos é aumentar sua rentabilidade e reduzir seu risco, pois toda carteira tem um retorno e um risco estimado e, geralmente, quanto maior o retorno, maior o risco a ela associado.

Uma carteira é dita eficiente se para um dado limite de risco imposto pelo investidor apresentar o máximo retorno possível ou, ainda, se para um nível mínimo de retorno exigido pelo investidor apresentar o mínimo risco.

Visando atingir tal objetivo Markowitz [1952] propôs um modelo matemático para determinar um portfólio ótimo. Tal modelo é baseado em um único período, tratando especificamente do compromisso (*trade off*) entre a taxa de retorno esperada e a variância da taxa de retorno do portfólio.

Segundo Benati [2003], o interesse neste problema tem crescido por dois motivos: o avanço tecnológico que tem permitido estudar problemas cada vez maiores e o fato dos gerenciadores de risco acreditarem que a variância pode não ser a medida mais apropriada para o risco. Muitos trabalhos podem ser encontrados, tais como, Konno [1991], Speranza [1996], Konno e Kobayashi [1997], Papahristodoulou e Dotzauer [2004] e Mansini e Speranza [2005].

Neste trabalho, por simplicidade e sem perda de generalidade, são consideradas carteiras compostas apenas por ações, ainda que os modelos sejam aplicáveis a situações de maior abrangência como, por exemplo, incluindo papéis do governo e fundos de renda fixa.

O objetivo principal deste trabalho é estudar a aplicabilidade do modelo proposto em Mansini e Speranza [2005] ao mercado brasileiro. As autoras propuseram um modelo que considera o problema da escolha de portfólio ótimo quando há custos fixos de transação e um limite máximo de investimento em cada uma das ações. Como objetivo secundário, é proposta a utilização apenas de software livre para o desenvolvimento do projeto. Logo, para avaliar o comportamento do modelo quando aplicado ao mercado brasileiro, o método proposto em Mansini e Speranza [2005] foi implementado em linguagem C utilizando as bibliotecas do

GLPK[1](GNU Linear Programming Kit), para resolver os problemas inteiro-mistos resultantes. O GLPK é um *software* livre de otimização que faz parte do projeto GNU. Além disso, o método proposto para determinar o portfólio ótimo utiliza como entrada dados históricos sobre as ações a serem consideradas. Tais dados podem ser obtidos utilizando o pacote proprietário ECONOMÁTICA[2]. No entanto, neste trabalho, foi desenvolvido um *parser* capaz de interpretar os arquivos históricos das ações disponíveis no *site* da BOVESPA[3].

Este projeto dá continuidade à iniciação científica iniciada em agosto e interrompida em dezembro de 2007, para dedicação ao estágio. Durante a iniciação científica foi estudado o problema de portfólio de ações e começamos a estudar o software GLPK.

Esta monografia está organizada da seguinte forma: no Capítulo 2 são descritos conceitos básicos sobre o tema abordado, o modelo proposto e é apresentada uma revisão bibliográfica sobre o tema. O método estudado e o algoritmo Branch-and-Bound são detalhados no Capítulo 3. Detalhes e implementação do método são apresentados no Capítulo 4. Descrição dos dados históricos, implementação do *parser*, testes computacionais com o *parser* e dificuldades encontradas em seu desenvolvimento estão no Capítulo 5. No Capítulo 6 são apresentados os testes realizados para o modelo estudado e dificuldades encontradas. As conclusões, principais contribuições do trabalho e sugestões para pesquisas futuras são descritas no Capítulo 7. Finalmente, no Capítulo 8, é feita uma análise crítica sobre o curso de Engenharia de Computação.

[1] http://www.gnu.org/software/glpk/

[2] http://www.economatica.com.br/

[3] http://www.bovespa.com.br/Principal.asp

2. O problema estudado

Markowitz [1952] propôs um modelo matemático para determinar o portfólio ótimo. Porém, apesar de ter uma grande reputação teórica este modelo não é muito usado na prática, pois possui uma complexidade computacional alta, dado que este modelo é quadrático e para utilizá-lo é necessário calcular a matriz de covariância entre as ações. O tamanho do problema cresce de acordo com o número de ações consideradas.

Tentando obter um resultado próximo ao proposto por Markowitz [1952], porém usando uma função linear para o cálculo do risco, Konno [1991] propôs a utilização do desvio absoluto da média (MAD – *Mean Absolute Deviation*) no lugar da covariância entre as ações. Com isso, não era mais necessário calcular a matriz de covariância entre todas as ações para iniciar o modelo. Além disso, resolver um problema linear é significativamente mais simples que resolver um problema quadrático e o número de restrições é mantido constante.

Em Konno [1991], o autor mostra que o retorno do modelo linear proposto é muito próximo ao retorno ótimo obtido pelo modelo de Markowitz [1952], porém o modelo linear possui uma complexidade computacional menor que permite a resolução do problema mais rapidamente.

Utilizando as transformações para tornar o problema linear, Speranza [1996] inseriu características do problema que normalmente eram negligenciadas, como custo de transação, quantidade mínima de ações a serem compradas e quantidade mínima para transação. Devido a essas características o problema proposto pela autora é inteiro-misto, sendo uma generalização do problema linear proposto na literatura. Com isso foi possível aumentar o realismo do modelo, porém foi necessário incluir um conjunto de variáveis inteiras, o que aumenta consideravelmente sua complexidade computacional. Para viabilizar a solução do problema, a autora propôs uma heurística. Primeiro o problema é relaxado, fazendo com que as variáveis inteiras sejam contínuas. A heurística então arredonda os valores das variáveis e verifica se todas as restrições são obedecidas. Caso sejam, essa solução é adotada, porém se não forem obedecidas, a heurística vai diminuindo o valor dessas variáveis até conseguir que todas as restrições sejam obedecidas ou chegar numa solução inviável. Essa heurística apresentou uma variação de 0,6 a 4,1% do valor ótimo, sendo que esse valor tende a diminuir com o aumento do capital a ser investido.

Em Konno e Kobayashi [1997], foi estudada a inclusão de obrigações ao modelo, não escolhendo apenas ações para compor a carteira. Obrigações são títulos de uma dívida de renda fixa ou variável emitida por um Estado, governo, município entre outros, com o objetivo de adquirir fundos diretamente do mercado financeiro. O emissor compromete-se a devolver o valor nominal junto com os juros. Os testes computacionais mostraram que essa nova abordagem pode obter resultados mais confiáveis.

Em Papahristodoulou e Dotzauer [2004], são estudados dois modelos de programação linear para encontrar o portfólio ótimo, o *maximin* e a minimização do Desvio Absoluto da Média. O modelo *maximin* busca maximizar o retorno mínimo desejado, pois considera que a visão de risco do investidor não é linear, o investidor não gosta de perder dinheiro, mas é necessário um grande lucro para que ele fique satisfeito. O modelo tenta então maximizar o retorno em cada período, sendo este valor pelo menos igual ao retorno mínimo exigido pelo investidor. Desse modo, pode ser que o modelo não consiga encontrar uma solução, por exemplo, quando em um determinado período qualquer a combinação das ações não consiga atingir o retorno mínimo desejado. Já o modelo de minimização do Desvio Absoluto da Média é o mesmo usado por Konno [1991]. A comparação desses dois modelos com o modelo de Markowitz [1952] para as ações da bolsa de Estocolmo mostrou que o modelo de Markowitz obtém o menor risco, o *maximin* obtém o maior retorno, porém com o maior risco e o modelo de minimização do Desvio Absoluto da Média obtém resultados próximos a Markowitz.

Neste trabalho estudamos o modelo proposto em Mansini e Speranza [2005], que tem por objetivo maximizar o lucro e minimizar o risco de uma carteira de ações. O lucro é calculado levando-se em consideração o imposto de renda, o preço da ação, o retorno médio mensal e o custo fixo de transação. Já o risco é calculado com base no desvio padrão dos retornos mensais das ações.

Para escolher o portfólio ótimo, primeiramente é necessário obter o conjunto de ações do qual se deseja escolher algumas ações, o valor máximo que deve ser gasto em uma determinada ação, os retornos mensais dessas ações, o valor de compra da ação, o capital que se deseja investir e o retorno esperado da carteira.

O modelo escolhido assume que o investidor pode distribuir seu capital entre n ações, com taxas de retorno aleatórias. $R = \{R_1,R_2,...,R_n\}$ é definido como um vetor de n variáveis aleatórias discretas que representam as taxas de retorno das ações. São considerados T cenários

possíveis com probabilidade p_t ($t = 1, 2... T$) de ocorrer, ou seja,

$$p_t = P\{(R_1, R_2, ..., R_n) = (r_{1t}, r_{2t}, ..., r_{nt})\} \quad (1)$$

em que os valores r_{jt} são as realizações de R_j sob os diferentes cenários. Mais precisamente, estes valores são extraídos de dados históricos, ou seja, situações ocorridas no passado podem ser repetidas. Para as autoras, os T períodos são considerados como cenários igualmente prováveis, ou seja, $p_t = 1/T$.

O modelo estudado é dado pelo seguinte problema de programação inteiro-mista:

$$\text{Maximizar} \quad \sum_{j \in n} \left[(1-g)\, r_j s_j x_j - c_j z_j \right] - \sum_{t=1}^{T} p_t y_t \quad (2)$$

$$\text{Sujeito a :} \quad y_t \geq \sum_{j \in n} \left(r_j - r_{jt} \right) s_j x_j, \qquad t = 1, ..., T \quad (3)$$

$$\sum_{j \in n} \left[(1-g) r_j s_j x_j - c_j z_j - \mu_0 s_j x_j \right] \geq 0, \quad (4)$$

$$\sum_{j \in n} s_j x_j \leq C, \quad (5)$$

$$x_j \leq u_j z_j, \qquad j = 1, ..., n \quad (6)$$

$$y_t \geq 0, \qquad t = 1, ..., T; \quad (7)$$

$$x_j \geq 0, \quad \text{inteiro} \qquad j = 1, ..., n; \quad (8)$$

$$z_j \in \{0,1\} \qquad j = 1, ..., n. \quad (9)$$

em que:

- j - índice que representa as ações, $j = 1, ..., n$;
- t - índice que representa os cenários, $t = 1, ..., T$;
- g - imposto pago sobre o retorno obtido (no Brasil é 15%);
- c_j - custo fixo de investimento na ação j;
- r_j - retorno médio da ação j;
- s_j - preço de cotação da ação j na data da escolha do portfolio;
- r_{jt} - retorno da ação j sob o cenário t;
- u_j - limitante superior do número de unidades da ação j que podem ser compradas;
- μ_0 - taxa mínima de retorno imposta pelo investidor;
- y_t - indica o risco da ação e é calculado usando o desvio médio;
- C - capital disponível para investimento;

x_j - quantidade de ações j adquiridas (variável);

z_i - variável binária que assume o valor 1 se a ação j for adquirida e 0 caso contrário;

A função objetivo (2), juntamente com as restrições (3) e (4), determina a maximização da diferença entre o retorno do portfólio e o máximo desvio do retorno médio. A restrição (4) garante que o retorno esperado, calculado como a diferença entre o retorno observado depois da cobrança da taxa (g) e da soma dos custos de transação associados a seleção das diversas ações, deve ser maior ou igual a taxa mínima de retorno especificada pelo investidor. A restrição (5) estabelece que o total investido no portfolio não pode exceder o capital disponível para aplicação. O conjunto de restrições (6) impõe um limite máximo de investimento em cada uma das ações selecionadas (z_j = 1), o que complementa a diversificação da carteira. Finalmente, as restrições (7) a (9) são as restrições de não-negatividade, e de integralidade das variáveis.

3. Método Estudado

Para resolver o problema estudado, utilizamos o método de solução proposto por Mansini e Speranza [2005]. Para implementarmos o método de solução utilizamos duas técnicas para solução de problemas de otimização: método *simplex* e método *branch-and-bound*.

O método *simplex* foi proposto por George Dantzig em 1947 para solução de problemas de otimização linear. Embora, no pior caso, sua complexidade seja exponencial [Klee *et al.,* 1972], o método, para a maior parte dos problemas, é muito eficiente na prática, conseguindo normalmente resolver o problema em *3*n* interações, onde *n* é a quantidade de restrições do problema, e convergindo em tempo polinomial.

A programação linear representa um caso especial de problemas de programação matemática em que se busca encontrar a solução que maximize (ou minimize) uma função satisfazendo uma série de restrições, no caso específico da programação linear, a função objetivo e as restrições do problema são lineares.

O problema estudado possui equações que seguem o padrão da programação linear, porém possui, além de variáveis contínuas, algumas variáveis inteiras. No caso em que todas as variáveis do problema são inteiras, o problema é chamado de "inteiro-puro", já no caso em que temos variáveis inteiras e contínuas, o problema é denominado "inteiro-misto". Existem também casos onde as variáveis inteiras podem assumir apenas os valores '0' e '1' nesses casos o problema é chamado de "binário".

Apesar de semelhante à programação linear, a programação inteira é mais abrangente, onde na maioria das vezes os problemas pertencem a classe NP-Difícil ou NP-Completo. Um algoritmo exato muito utilizado para solução de problemas inteiros é o *Branch and Bound*.

3.1 Branch-and-Bound

O método baseia-se em dois princípios básicos: partição e relaxação. A partição consiste em dividir o conjunto de soluções factíveis do problema original em diversos conjuntos disjuntos. A cada conjunto é associado um problema de otimização (P^i), pois a otimização de conjuntos menores é mais simples e eficiente.

Normalmente faz-se a partição de forma recursiva e podemos representá-la através de uma árvore. A Figura 1 ilustra uma árvore de partição completa para o caso em que duas variáveis binárias são consideradas.

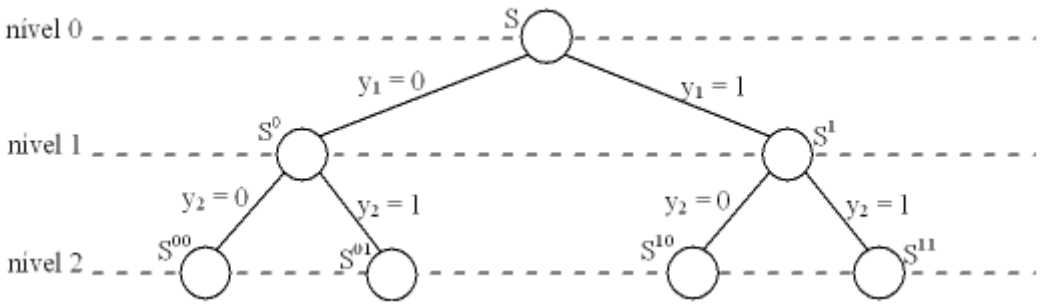

Figura 1 – Árvore de partição completa

A partição de um conjunto S^i é interrompida quando:

- $S^i = \emptyset$;

- Uma solução ótima P^i é encontrada;

- O valor de qualquer solução factível de P^i é pior que o valor da melhor solução factível atual.

Quando uma situação dessas ocorre o nó S^i pode ser descartado e, portanto, não se executa a enumeração completa. É mais fácil verificar essas condições quando se usa relaxação. Neste trabalho será considerada a relaxação linear, que considera as variáveis inteiras como reais. Desse modo é obtido um problema relaxado de programação linear P^i_R. Desse modo as condições são modificadas para:

- $S^i_R = \emptyset$;

- A solução ótima de $P^i_R{}^i$ é inteira;

- O valor de qualquer solução factível de P^i_R é pior que o valor da melhor solução factível atual.

Se o problema inteiro possuir um limitante inferior elevado (para problemas de maximização) ou um limitante superior baixo (para problemas de minimização) então é mais fácil eliminar um nó. Portanto, o ideal é que a relaxação seja de fácil resolução e forneça um limitante bom.

3.2 Método de Mansini e Speranza (2005)

Mansini e Speranza (2005) propuseram uma estratégia de solução para o problema que resultou, em média, na redução de 60% do tempo necessário para resolver o problema inteiro. O método inicialmente relaxa linearmente a modelagem original. Apesar da solução do modelo poder não resultar em uma solução factível para o problema original, pois não existem frações de ações, o problema relaxado é de fácil solução. Utilizando o resultado dessa modelagem as autoras através de uma heurística conseguem achar uma solução factível que será usada como limitante inferior do problema inteiro.

O primeiro passo do método proposto pelas autoras é então resolver a relaxação linear do problema proposto, ou seja, as variáveis inteiras passam a ser consideradas contínuas, como descrito a seguir:

$$\text{Maximizar} \quad \sum_{j\in n}\left[(1-g)\, r_j s_j x_j - c_j z_j\right] - \sum_{t=1}^{T} p_t y_t \qquad (11)$$

$$\text{Sujeito a :} \quad y_t \geq \sum_{j\in n}\left(r_j - r_{jt}\right) s_j x_j, \qquad t = 1,...,T \qquad (12)$$

$$\sum_{j\in n}\left[(1-g) r_j s_j x_j - c_j z_j - \mu_0 s_j x_j\right] \geq 0, \qquad (13)$$

$$\sum_{j\in n} s_j x_j \leq C, \qquad (14)$$

$$x_j \leq u_j z_j, \qquad j = 1,...,n \qquad (15)$$

$$x_j \geq 0, \qquad j = 1,...,n; \qquad (16)$$

$$0 \leq z_j \leq 1 \qquad j = 1,...,n. \qquad (17)$$

A solução do problema linear [11]-[17] é obtida facilmente pelo método simplex. No entanto, pode resultar em valores não inteiros para as variáveis x_i, as quais originalmente deveriam ser inteiras e valores não binários para as variáveis z_j, ou seja, o custo fixo de transação é considerado pelo modelo como se fosse proporcional à quantidade investida.

Na seqüência do método, são obtidos os valores dos custos reduzidos das variáveis de decisão z_j que serão chamados de $\{q_1, \ldots, q_n\}$. A partir desses valores e de um parâmetro chamado *THRESHOLD* (as autoras adotam $THRESHOLD = 10^{-4}$), se obtém um conjunto M de ações dado por:

$$M := \{ j \in N \mid |q_j| \leq THRESHOLD \}.$$

Note que o conjunto M é composto por todas as variáveis básicas da solução relaxada (custo reduzido igual a zero) mais as variáveis cujos custos reduzidos são muito pequenos, ou seja, que poderiam passar a fazer parte da solução alterando minimamente o valor da função objetivo. Usando o conjunto M e a solução relaxada, gera-se um limitante inferior do problema a partir da heurística resumida a seguir:

Passo 1. Defina o conjunto M e faça $x_j^*, z_j^*, j \in M$ ser a solução ótima correspondente.

Passo 2. Faça $x_j^+ = x_j^*$ e $x_j^- = 0$ para todo $j \in M$.

Passo 3. Defina $M^1 := \{ j \in M \mid z_j^* > 0{,}5 \}$.

Passo 4. Enquanto $\lfloor x_j^+ \rfloor \neq \lfloor x_j^- \rfloor$ para algum $j \in M^1$

$$Faça \quad k = \frac{\sum_{j \in M} s_j x_j^+}{\sum_{j \in M^1} s_j \lfloor x_j^+ \rfloor}$$

$$x_j^- = x_j^+, \; j \in M^1$$
$$x_j^+ = \min\{u_j, k\lfloor x_j^* \rfloor\}, \; j \in M^1$$

Passo 5. Faça $\bar{x}_j = \lfloor x_j^+ \rfloor (\bar{z}_j = 1) \; se \; j \in M^1 \; e \; \bar{x}_j = 0$ caso contrário.

Passo 6. Se $\bar{x}$ for factível, então calcule o valor da função objetivo. Esse valor é adotado como limitante inferior.

Com base no conjunto M, são gerados dois subproblemas:

$$S^{(1)} = Problema\ Original \cup \sum_{j \in N \setminus M} z_j = 0.$$

$$S^{(2)} = Problema\ Original \cup \sum_{j \in N \setminus M} z_j \geq 1$$

O Subproblema 1 é o problema inteiro original com mais uma restrição que indica que as variáveis que fazem parte do conjunto M deverão ser igualadas a zero e, portanto, não entrarão na solução.

O Subproblema 2 é o problema inteiro original com mais uma restrição que indica que pelo menos uma das variáveis que fazem parte do conjunto M deve fazer parte da solução.

Resolve-se então o problema $S^{(1)}$ usando o algoritmo *branch-and-bound* tendo por limitante inferior o valor obtido pela heurística descrita acima. Dada a solução ótima $S^{(1)}$ ($x^{(1)}$) e

seu valor $h^{(1)}$, atualiza-se o valor do limitante inferior e segue-se com a resolução do subproblema $S^{(2)}$. A solução ótima do problema original é dada pelo maior valor obtido no processo.

Na Figura 2 é resumido o método proposto em Mansini e Speranza (2005).

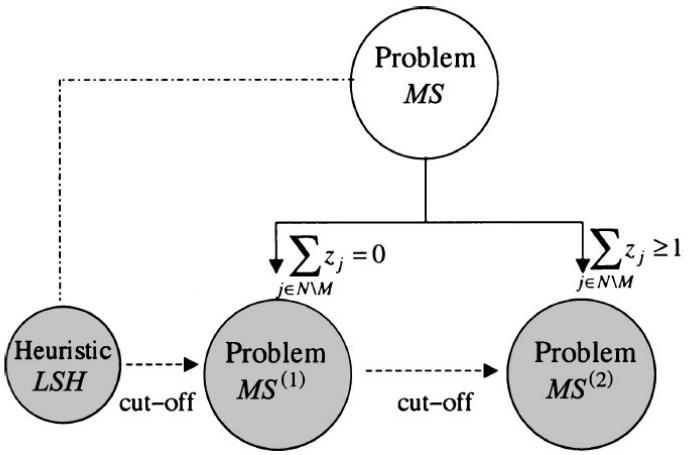

Figura 2. O algoritmo exato. Fonte: Mansini e Speranza [2005].

4. Implementação do método

4.1. GLPK

Para resolver os problemas de otimização linear e inteiro-mista que são demandados pelo método proposto por Mansini e Speranza [2005] foi utilizado o pacote GNU Linear Programming Kit (GLPK).

O GLPK é um pacote de software utilizado para resolver problemas de Programação Linear, Programação Inteira Mista e problemas correlacionados. O pacote possui diversas funções escritas em ANSI C para auxiliar na resolução desses problemas, essas funções estão organizadas em diversas bibliotecas.

O pacote faz parte do GNU Project e é lançado com a GNU General Public License. O GLPK possui duas aplicações, *glpsol* e *tspsol*. O *glpsol* é uma aplicação que resolve problemas de Programação Linear e Programação Inteira Mista, enquanto o *tspsol* é uma aplicação que resolve o problema do Caixeiro Viajante. Nesse trabalho foram estudadas e utilizadas as funções da biblioteca do GLPK.

O GLPK utiliza o método simplex e o método primal-dual para problemas lineares, e também o algoritmo *branch-and-bound* com cortes de Gomory para problemas inteiro-mistos.

O pacote GLPK pode ser utilizado de duas formas: apenas como *solver* (aplicação *glpsol*) ou desenvolvendo um aplicativo em C que utilize as bibliotecas que o pacote oferece para resolver os problemas dinamicamente.

Caso o usuário opte por utilizar o solver *glpsol*, o problema a ser resolvido deve ser descrito em um arquivo texto que é utilizado como entrada para a aplicação. Este problema de programação matemática deve ser escrito utilizando a linguagem de modelagem GNU Math Prog. A descrição dos modelos nessa linguagem possui um conjunto de declarações e dados que são descritos no documento *lang.pdf*[4]que acompanha a versão completa do pacote. O texto passa então por um processo de tradução em que as informações são lidas, o problema é resolvido e os resultados são apresentados no console (terminal).

O desenvolvimento da aplicação utilizando as bibliotecas é simples, as funções estão

[4] http://gnuwin32.sourceforge.net/packages/glpk.htm 17/01/2008.

definidas no documento *refman48.pdf*[5] que acompanha a versão completa do pacote. Nesse documento há uma pequena introdução aos problemas de Programação Linear e Programação Inteira Mista e também um breve exemplo de como se desenvolve um programa para resolver um problema de Programação Linear. Além disso, as bibliotecas permitem que o problema que está sendo definido seja exportado para um arquivo texto que pode ser interpretado por outros pacotes de otimização tais como, pelo CPLEX[6].

Neste trabalho, o método proposto em Mansini e Speranza [2005] foi implementado em linguagem C utilizando as bibliotecas do GLPK.

4.2. Detalhes de implementação

O método e a heurística foram implementados em C utilizando funções do software GLPK para resolver os problemas lineares e os problemas inteiros. Inicialmente para conseguir utilizar as funções do GLPK é necessário importar a biblioteca "glpk.h". Além disso, é necessário que o problema possua seja definido no seguinte formato:

Maximizar/Minimizar	$c^T x$	(18)
Sujeito a :	$l \leq Ax \leq u$	(19)
	$l \leq x \leq u$	(20)

4.2.1 Problemas de Programação Linear

Para utilizar as funções do GLPK para resolver um problema de programação linear é preciso seguir os seguintes passos:

- Criar um ponteiro para uma variável LPX que conterá os dados do problema.

- Utilizar a função *lpx_create_prob()* (esta função retorna um LPX *) para criar o problema e armazená-lo na variável.

- Definir o nome do problema utilizando a função *lpx_set_prob_name(LPX *lp, char *name)*.

[5] http://gnuwin32.sourceforge.net/packages/glpk.htm 17/01/2008.

[6] Pacote de otimização desenvolvido pela ILOG - http://www.ilog.com/products/cplex/

- Indicar se o problema é de máximo ou de mínimo utilizando a função *lpx_set_obj_dir(LPX *lp, int dir)*, onde dir pode ser LPX_MAX ou LPX_MIN.

- Indicar a quantidade de linhas que o problema terá usando a função *lpx_add_rows(LPX *lp, int row)*.

- Para cada linha do problema deve-se dar um nome e definir os limites inferiores e superiores dessa linha, como pode ser visto em (19), utilizar as funções:

•*lpx_set_row_name(LPX *lp, int nrow, char *name)*

•*lpx_set_row_bnds(LPX *lp, int nrow, int type, double lb, double up)*, onde *type* pode ser LPX_FR (variável livre), LPX_LO (variável com limite inferior), LPX_UP (variável com limite superior), LPX_DB (variável com limite inferior e superior), LPX_FX (valor fixo).

- Indicar a quantidade de colunas que o problema terá usando a função *lpx_add_cols(LPX *lp, int col)*.

- Cada coluna do problema representa uma variável. Para cada coluna é preciso definir: um nome, seu limite inferior e seu limite superior (ver (20)). Também devemos indicar o valor do coeficiente dessa variável na função objetivo. Para tanto são utilizadas as funções:

•*lpx_set_col_name(LPX *lp, int ncol, char *name)*,

•*lpx_set_col_bnds(LPX *lp, int ncol, int type, double lb, double up)*, onde type pode ser LPX_FR, LPX_LO, LPX_UP, LPX_DB, LPX_FX ;

•*lpx_set_obj_coef(LPX *lp, int ncol, double coef)*.

- Definir a matriz A inserindo apenas seus valores não-nulos, para tanto são definidos três vetores: *ia* (contém os índices da linha), *ja* (contém os índices da coluna), *ar* (contém os valores). Nesta etapa é utilizada a função *lpx_load_matrix(LPX *lp, int num, int *ia, int *ja, double *ar)*, onde *num* é o total de elementos não-nulos da matriz A.

- Utilizar a função *lpx_simplex(LPX *lp)* para resolver o problema.

- Utilizar a função *lpx_get_obj_val(LPX *lp)* para obter o valor da função objetivo.

- Utilizar a função *lpx_get_col_prim(LPX *lp, int col)* para obter o valor de cada variável.

- Utilizar a função *lpx_get_col_dual(LPX *lp, int col)* para obter os custos relativos de cada variável.

- Utilizar a função *lpx_delete_prob(LPX *lp)* para apagar o problema.

4.2.2 Problemas de Programação Inteiro-Mista

Para utilizar as funções do GLPK para resolver um problema de programação inteiro-mista é preciso seguir os seguintes passos:

- Primeiro é necessário ter o problema relaxado e resolver o problema relaxado seguindo os passos descritos anteriormente.

- Mudar a classe do problema para inteira-mista utilizando a função *lpx_set_class(LPX *lp, int LPX_MIP).*

- Utilizar a função *lpx_set_col_kind(LPX *lp, int col, int parm)* onde *parm* pode ser LPX_CV (para variáveis contínuas) ou LPX_IV (para variáveis inteiras).

- Utilizar a função *lpx_integer(lp)* para resolver o problema.

- Utilizar a função *lpx_mip_obj_val(LPX *lp)* para obter o valor da função objetivo.

- Utilizar a função *lpx_mip_col_val(LPX *lp, int col)* para obter o valor de cada variável.

- Utilizar a função *lpx_delete_prob(LPX *lp)* para apagar o problema.

5. Dados Históricos

Os dados históricos sobre as ações brasileiras podem ser obtidos utilizando o pacote de economia ECONOMÁTICA. No entanto, este pacote é proprietário e como um dos objetivos deste trabalho é utilizar apenas softwares não-proprietários, buscamos tais informações no site da BOVESPA.

Os dados histórico de todas as ações negociadas pela BOVESPA estão disponíveis no site da empresa. As informações são apresentadas em arquivos que seguem um padrão específico, que é definido no documento *SeriesHistoricas_Layout.pdf*[7], também disponível no site. Estes arquivos contêm, além dos preços de abertura e fechamento das ações, muitas outras informações sobre as ações. No entanto, para o problema estudado, são necessárias apenas as seguintes informações sobre as ações: a data do pregão, o código da ação e o valor de fechamento. A seguir é apresentado um exemplo de uma linha desse arquivo, vale destacar que os espaços são considerados em sua formatação,

"012007010202ABNB3 010ABNOTE ON NM R$ 000000000172000000000017200000000001700000000000170900000000017000000000001700000000001710004000000000000009840000000000168251400000000000000009999123100000010000000000000BRABNBACNOR4102"

Definição da linha segundo o documento da BOVESPA:

- Da posição 3 até a posição 6 está escrito o ano do pregão, nesse caso 2007

- Da posição 7 até a posição 8 está escrito o mês do pregão, nesse caso 01

- Da posição 9 até a posição 10 está escrito o dia do pregão, nesse caso 02

- Da posição 13 até a posição 24 está escrito o código da ação, nesse caso ABNB3.

- Da posição 109 até a posição 121 está escrito o valor de fechamento da ação neste dia, nesse caso 0000000001700, sendo que os dois últimos números são as casas decimais, portanto o valor é R$ 17,00.

Como se pode observar, extrair manualmente os dados das ações não é uma tarefa simples e rápida, portanto, neste trabalho é proposto um *parser* capaz de extrair dos arquivos de dados históricos as informações necessárias para o modelo estudado.

[7] http://www.bovespa.com.br/Pdf/SeriesHistoricas_Layout.pdf 01/10/2008.

5.1 *Parser* proposto

O *parser* é composto de dois programas, um que recebe o arquivo de cotação histórica da BOVESPA e retorna o preço da ação em cada período e o outro programa que recebe o arquivo do primeiro programa e calcula os retornos das ações.

O primeiro programa precisa receber como dados de entrada:

- A quantidade de períodos que se deseja calcular (12, quando se deseja os retornos mensais e 52 quando se deseja os retornos semanais).
- A data fim de cada período que se deseja calcular (no formato D M, onde D representa o dia e M representa o mês, no caso de retorno mensal deve-se colocar o último dia do mês no caso de retorno semanal, deve-se colocar sempre o mesmo dia da semana).
- O arquivo de cotação histórica.

O primeiro programa então percorre cada linha do arquivo de cotação histórica e guarda o preço de fechamento da ação no último dia de cada período (no caso da ação não ser negociada nesse dia, ele armazena o preço do último dia negociado antes da data requerida). O programa percorre uma única vez o arquivo de entrada e consegue obter todos os dados desejados. Em seguida, ele ordena as ações em ordem alfabética.

O segundo programa, a partir do arquivo gerado, calcula o retorno de cada período utilizando a fórmula:

retorno = (período_atual – período_anterior) / período_anterior.

5.2 Testes computacionais do *Parser*

O *parser* foi testado comparando os retornos obtidos pelo *software* ECONOMÁTICA com os retornos fornecidos pelo *parser*. Nas Tabelas 1 e 2, a título de exemplo, comparamos os resultados obtidos pelo *parser* e extraídos do ECONOMÁTICA para duas ações PETR3 e ARCZ3. Na primeira coluna é definido o período, na segunda e terceiras colunas são dados os retornos obtidos utilizando, respectivamente, o ECONOMÁTICA e o *parser.* Na quarta coluna, é reportada a diferença entre os resultados.

Como se pode observar, o desvio foi significativo apenas para: Set/2005. Este desvio ocorre, pois no mês de setembro de 2005 ocorreu um agrupamento das ações da Petrobrás, ou seja, as ações da empresa foram agrupadas para formar uma nova ação. Nos arquivos de dados da BOVESPA, as operações de agrupamento e divisão não são reportadas, o que impede que o *parser* detecte automaticamente tal operação. Essa falha pode ser corrigida manualmente pelo usuário do programa. Note que para a ação ARCZ3 não houve agrupamento.

Os demais desvios nos retornos foram causados pela imprecisão dos dados fornecidos pela BOVESPA. Os dados fornecidos pela BOVESPA nos arquivos de cotações históricas possuem precisão de duas casas decimais apenas, enquanto os dados do ECONOMÁTICA contam com seis casas decimais. Note, no entanto, que na maioria das situações não ocorre uma diferença significativa entre os retornos.

PETR3	**Economática**	**Parser**	**Parser - Economática**
fev/05	0,183597	0,183597	0,000000
mar/05	-0,074803	-0,074803	0,000000
abr/05	-0,071931	-0,085106	-0,013175
mai/05	0,042884	0,042884	0,000000
jun/05	0,082062	0,082062	0,000000
jul/05	0,050205	0,032891	-0,017314
ago/05	0,186752	0,186752	0,000000
set/05	0,082717	-0,729321	**-0,812038**
out/05	-0,114286	-0,114286	0,000000
nov/05	0,054698	0,054698	0,000000
dez/05	0,098404	0,098404	0,000000
jan/06	0,266912	0,251574	-0,015338
fev/06	-0,059779	-0,059779	0,000000
mar/06	-0,041152	-0,041152	0,000000
abr/06	0,109006	0,094635	-0,014371
mai/06	-0,030974	-0,030974	0,000000
jun/06	-0,020838	-0,020838	0,000000
jul/06	0,032025	0,032025	0,000000
ago/06	-0,048649	-0,048649	0,000000
set/06	-0,045665	-0,045665	0,000000
out/06	0,045204	0,045204	0,000000
nov/06	0,099138	0,075949	-0,023189
dez/06	0,068431	0,068431	0,000000

Tabela 1 – Retorno da ação PETR3

ARCZ3	Economática	Parser	Parser - economática
fev/05	0,026950	0,026950	0,000000
mar/05	0,077348	0,077348	0,000000
abr/05	-0,061069	-0,070513	-0,009444
mai/05	-0,031246	-0,055172	-0,023926
jun/05	0,054950	0,051095	-0,003855
jul/05	0,041667	0,041667	0,000000
ago/05	-0,026667	-0,026667	0,000000
set/05	0,094521	0,094521	0,000000
out/05	0,001252	0,001252	0,000000
nov/05	-0,043750	-0,043750	0,000000
dez/05	-0,000867	-0,019608	-0,018741
jan/06	0,073333	0,073333	0,000000
fev/06	0,093168	0,093168	0,000000
mar/06	0,157144	0,147727	-0,009417
abr/06	0,178218	0,178218	0,000000
mai/06	0,003218	-0,008403	-0,011621
jun/06	0,014479	0,008475	-0,006004
jul/06	-0,052941	-0,052941	0,000000
ago/06	-0,007098	-0,007098	0,000000
set/06	0,013629	0,007149	-0,006480
out/06	0,091393	0,091393	0,000000
nov/06	0,101626	0,101626	0,000000
dez/06	-0,035481	-0,040590	-0,005109

Tabela 2 – Retorno da ação ARCZ3

5.3. Dificuldades encontradas

Uma ação pode não ter sido negociada durante um dia, nesse caso ela não aparece no arquivo da BOVESPA. Quando isso ocorre é necessário resgatar o valor do fechamento anterior para que este seja usado para o dia em que a ação não foi negociada, portanto caso uma ação não seja negociada no último dia do período desejado é retornado o valor do fechamento do último dia que a ação foi negociada.

Nem todos os arquivos de cotações históricas da BOVESPA estavam ordenados por ordem cronológica o que dificultou o trabalho, porém como já existia o tratamento para

armazenar os valores do fechamento anteriores da ação, caso ela não tenha sido negociada, foi necessário apenas uma pequena alteração para guardar esses valores de fechamento por período, anteriormente era armazenado em uma mesma variável.

6. Testes computacionais do modelo e dificuldades encontradas

O programa foi testado com os dados gerados pelo *parser* para as ações que fazem parte do IBOVESPA para janeiro de 2007, 58 ações, porém a ação da empresa Eletropaulo (ELPL6) começou a ser negociada no dia 31/08/2006, e resolvemos não incluí-la no conjunto de ações a serem analisadas, pois para as demais ações seriam utilizados doze cenários.

Para cada uma dessas ações foi calculado o retorno mensal ou semanal do ano anterior (entre janeiro de 2006 e dezembro de 2006).

Foram geradas 8 carteiras, para todas, foi considerado o capital inicial de R$1.000.000,00, e imposto de renda de 15%. Foi utilizado o valor de fechamento do último dia útil de dezembro de 2006 para o preço de compra de cada ação. As 8 carteiras geradas são definidas pelos parâmetros descritos na Tabela 1.

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**
Retorno mínimo	1,43%	1,72%	1,43%	1,72%	1,43%	1,72%	1,43%	1,72%
Custo fixo de transação	R$ 1,00	R$ 1,00	R$ 2,00	R$ 2,00	R$ 1,00	R$ 1,00	R$ 2,00	R$ 2,00
Base	mensal	mensal	mensal	mensal	semanal	semanal	semanal	semanal

Tabela 3 – Carteiras geradas

Após o programa escolher quais ações fariam parte de cada carteira de ação, foi calculado o retorno anual obtido, levando em consideração o ano seguinte (janeiro de 2007 até dezembro de 2007). Para os parâmetros definidos para as carteiras 1, 2, 3 e 4 a carteira escolhida pelo modelo foi a mesma, a qual é descrita na Tabela 4. Ao considerarmos as características das carteiras 5, 6, 7 e 8, também foi obtida uma mesma carteira, cuja composição é apresentada na Tabela 5.

Ação	**Quantidade**
ACES4	3070
ALLL11	9021
BRTP3	1835
LIGT3	2382
SBSP3	682
TCSL3	9034
VCPA4	4811

Tabela 4 – Ações da carteira C1, C2, C3 e C4.

Ação	Quantidade
ACES4	118
ALLL11	9021
BRTP3	1207
CSNA3	2789
LIGT3	8689
SBSP3	636
SUBA3	2123
TCSL3	7327

Tabela 5 – Ações da carteira C5, C6, C7e C8.

A Figura 3 mostra uma comparação dos retornos acumulados no ano de 2007 para as carteiras C1 e C5 e pelo IBovespa. O IBovespa é um índice que é composto por ações mais negociadas no mercado a vista. A porcentagem de cada ação na carteira é definida pelo número de negócios e volume financeiro no mercado a vista.

As carteiras C1 e C5 foram calculadas com base nas ações que faziam parte do IBovespa para janeiro de 2007, porém a cada quatro meses esse índice é recalculado, o que não foi feito para as carteiras. Apesar de não ter sido recalculada a carteira C5 esteve durante 10 meses com um retorno acumulado superior ao retorno do IBovespa, o que mostra que a utilização de um modelo para definição da carteira de ações é interessante.

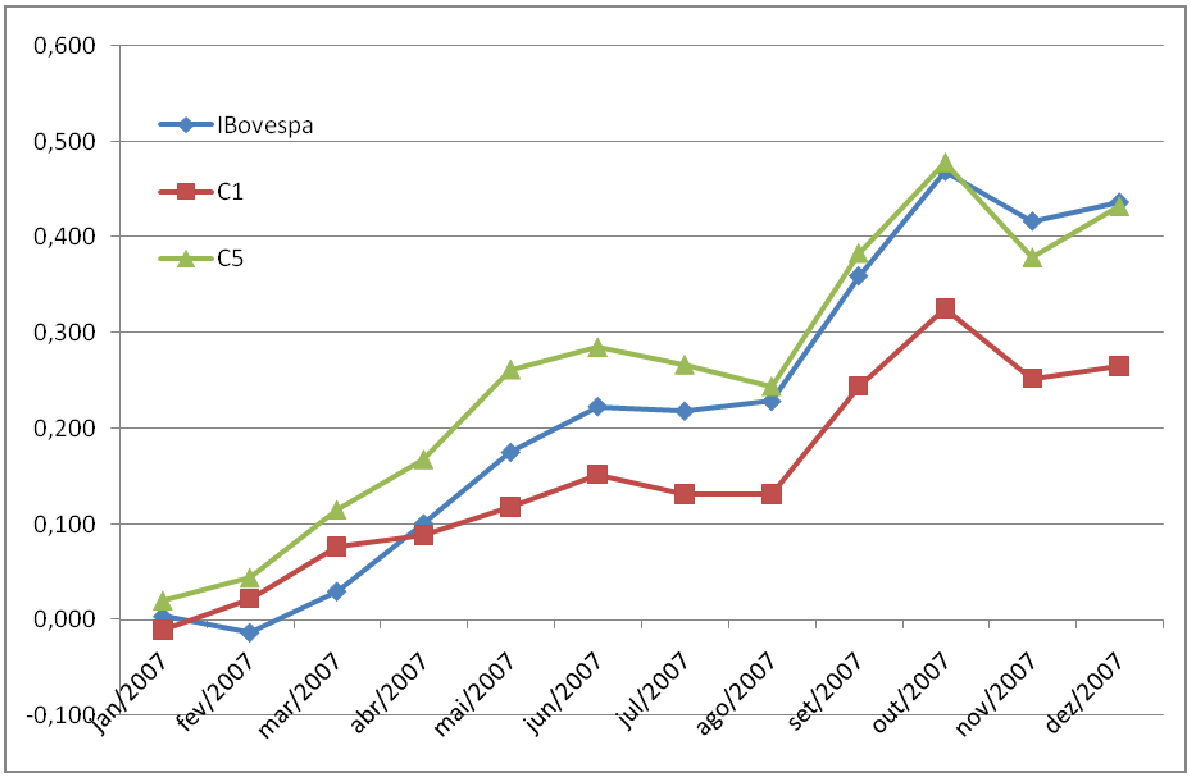

Figura 3. Gráfico comparando os retornos acumulados no período.

A maior dificuldade encontrada foi com relação a documentação do GLPK, pois não explica o uso correto de algumas funções necessárias para o desenvolvimento do projeto. Mesmo assim, foi possível, através de buscas em *sites* achar todas as funções necessárias para o desenvolvimento do projeto.

7. Conclusões e trabalhos futuros

O modelo proposto foi aplicado ao mercado brasileiro e os resultados comparados ao índice IBovespa. Pelos testes realizados conseguimos observar que as carteiras obtiveram pelo menos o retorno mínimo esperado. No caso da carteira gerada com os retornos semanais em alguns meses foi possível superar não só esse retorno, mas também o retorno obtido pelo IBovespa no período.

A aplicabilidade do modelo ao mercado brasileiro foi comprovada e a utilização do *software* livre GLPK e o desenvolvimento do *parser* permitem que esse modelo seja aplicado ao mercado sem que haja a necessidade de um *software* proprietário. O GLPK conseguiu calcular os dados para as 57 ações rapidamente, demorando em média 2 segundos.

Durante o desenvolvimento do projeto constatamos a possibilidade de incluir uma variável que representasse a liquidez de uma ação. A liquidez indica o quão rápido é realizada a compra e venda de uma ação, pois apesar de uma determinada ação ter sido escolhida para fazer parte da carteira, pode ocorrer de não existirem ofertas de venda da ação, e posteriormente pode não ocorrer a compra da mesma. Porém a liquidez não foi utilizada no trabalho, pois o tempo de desenvolvimento do trabalho não permitia.

Referências Bibliográficas

[1] Konno, H. e Kobayashi, K. (1997) An integrated Stock-Bond Portfolio Optimization Model, *Journal of Economic Dynamics and Control*, 21, 1427-1444.

[2] Markowitz H. (1952) Portfolio selection. *Journal of Finance* 7, 77-91.

[3] Benati, S. (2003) The optimal portfolio problem with coherent risk measure constraints. *European Journal of Operational Research* 150, 572-584.

[4] Konno, H. (1991) Mean-Absolute Deviation Portfolio Optimization Model and its Applications to Tokyo Stock Market*, Management Science 37,* 519-531.

[5] Speranza, M.G. (1996), "A heuristic algorithm for a portfolio optimization model applied to the Milan stock market", *Computers and Operations Research* 23, 5 p. 433-441.

[6] Papahristodoulou, C. e Dotzauer, E. (2004), "Optimal portfolios using linear programming models", *Journal of the Operational Research Society* 55, p. 1169-1177.

[7] Mansini, R. e Speranza, M. G. (2005), "An exact approach for portfolio selection with transaction costs and rounds", *IIE Transaction* 37, p. 919-929.

[8] Dantzig, G. B. "Linear Programming and Extensions". Princeton, NJ: Princeton University Press, 1963.

[9] Klee, V.; Minty, G. J.; and Shisha, O. (Eds.) (1972) "How Good is the Simplex Algorithm?" In "Inequalities 3". New York: Academic Press, p. 159-175.