# UM ESTUDO DO PLANEJAMENTO DE ROTAS AÉREAS COM MÚLTIPLOS DESTINOS

**Kátia Yoshime Nakamura e Mariá Cristina Vasconcelos Nascimento**

Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
Rua Talim, 330 – Jd. Aeroporto – São José dos Campos – São Paulo – Brasil
E-mails: {katia.nakamura, mcv.nascimento}@unifesp.br

## RESUMO

Atualmente, o transporte aéreo tem sido utilizado por milhões de pessoas em todo o mundo por ter se tornado mais acessível a diversas classes sociais. Adicionalmente, a existência de várias companhias aéreas aumenta a gama de opções de passagens aéreas. Consequentemente, para uma melhor escolha de compra de passagens, é necessário um planejamento prévio das rotas de viagem. Dessa maneira, realizar um planejamento em que se define a rota com o menor custo de passagens é de considerável relevância prática. Este artigo apresenta uma formulação matemática para a determinação de rotas aéreas dado um conjunto de cidades a serem visitadas, sem a definição prévia da ordem de viagem, para que a decisão da rota de viagem fique a critério da minimização de custos das passagens e tempos de estadia estabelecidos. Para resolver esse problema, uma metaheurística GRASP é desenvolvida e, segundo experimentos, a mesma forneceu resultados satisfatórios em casos de testes com dados reais e comparando com o desempenho do pacote CPLEX.
**PALAVRAS CHAVE**. **Planejamento de rotas aéreas, Múltiplos destinos,** *GRASP*

## ABSTRACT

Nowadays, air transportation has been used by millions of people over the world albeit due its accessibility to more social classes. Additionally, the existence of several airline companies increases the number of options of available tickets. Consequently, for a better choice for purchasing air tickets, it is necessary a previous planning of the travel route. Thus, the definition of the minimum cost route regarding the ticket costs is very relevant in practice. This paper presents a study of a mathematical formulation that aims to determine the best travel route according to the minimum total costs of air tickets and of the time to visit each place. To solve this problem, a metaheuristic GRASP is developed and, according to the experiments, it provided satisfactory results regarging the real data case tests and comparing with the performance of the solver CPLEX.
**KEYWORDS. Air route planning, Multiple travelling, GRASP**

## 1. Introdução

Segundo dados da Agência Nacional de Aviação Civil, a última década foi marcada por um grande aumento no número de pessoas que utilizam transporte aéreo. No ano de 1998, estima-se que 22,8 milhões de passagens aéreas foram vendidas no Brasil, enquanto que, em 2008, esse número chegou a 56,2 milhões. Apesar desses números estarem estáveis nos últimos cinco anos, eles indicam um grande interesse por parte da população nesse meio de transporte.

Muitos dos interessados no transporte aéreo são turistas que desejam conhecer pontos turísticos localizados em diversas localidades (Vansteenwegen et al., 2011). Para auxiliá-los, as próprias empresas, muitas vezes, disponibilizam em suas páginas eletrônicas a opção de viagem em múltiplos destinos. Entretanto, o usuário deve informar a ordem exata das cidades em que deseja viajar, sendo que muitas vezes a escolha da ordem de viagem por parte do viajante é influenciada pelo preço, que ele não conhece previamente.

Este artigo aborda o problema de decidir a melhor rota aérea de viagem de um passageiro, aqui denominado de Problema de Planejamento de Rotas Aéreas com Múltiplos Destinos (*Planning Air Routes with Multiple Destinations Problem* ou *PARMDP*), tendo como principal critério de decisão os custos totais de passagens. Um trabalho correlato é o de Ambite et al. (2002), que é direcionado para auxiliar o usuário no monitoramento de passagens disponíveis. Entretanto, alguns parâmetros não cobertos pela literatura foram considerados neste trabalho de iniciação científica, como a busca pela sequência de viagens, sem definir sua ordem exata para realizar o percurso levando em consideração o custo e a distância das rotas, e o tempo que o indivíduo ficará em cada destino e a imposição de um limite de tempo total da rota. Além disso, uma metaheurística *Greedy Randomized Search Procedure* (*GRASP*) (Feo and Resende, 1989) foi proposta e testada em alguns casos de teste baseados em dados reais coletados de um banco de dados. Como resultado, uma das versões do *GRASP* proposto apresentou bons resultados, sendo que, para um dos casos de teste, todas as versões do *GRASP* encontraram a sua solução ótima em frações de segundo.

## 2. Formulação Matemática

Para o *PARMDP*, considere a existência de $n$ localidades a serem visitadas, sendo $n-1$ delas distintas entre si, e que cada um desses lugares esteja indexado com um valor natural entre o intervalo $[1, n]$. Desse conjunto de $n$ lugares, o índice 1 será sempre a origem da viagem, e o índice $n$ será sempre o destino final da viagem, que aqui será sempre considerado como sendo o mesmo que a origem.

É necessário definir os horários de viagem para a consideração correta da rota de viagem, lembrando-se que é necessário também considerar os tempos de estadia nas diferentes localidades. Para facilitar essa questão, dividiu-se o dia em seis períodos distribuídos de acordo com a Tabela 1.

Inicialmente, os parâmetros a serem considerados para a modelagem do nosso problema são $t_{ijt}$ e $c_{ijt}$. O parâmetro $t_{ijt}$ representa o número total de períodos de estadia a partir do momento em que se desloca da localidade $i$ para a $j$. Nesse caso, tal parâmetro será o número de dias de estadia em $j$, que é definido pelo usuário, vezes o número de períodos considerado por dia, mais o número de períodos de viagem para se chegar ao destino $j$ considerando que o passageiro saia de $i$ no tempo $t$. Já o parâmetro $c_{ijt}$ é o custo da passagem partindo de um lugar $i$ com destino ao lugar $j$ num dado tempo $t$.

As variáveis consideradas para o nosso problema são:

- $x_{ijt}$: recebe 1, caso o indivíduo saia da cidade $i$ para a $j$ no tempo $t$; e 0, caso contrário;

Tabela 1: Divisão do dia em períodos.

| | JANELA DE TEMPO | |
|---|---|---|
| Período | Horário inicial | Horário final |
| $1^o$ | 0:01h | 4:00h |
| $2^o$ | 4:01h | 8:00h |
| $3^o$ | 8:01h | 12:00h |
| $4^o$ | 12:01h | 16:00h |
| $5^o$ | 16:01h | 20:00h |
| $6^o$ | 20:01h | 0:00h |

- $t_i$: valor discreto que representa o tempo de saída de um lugar $i$.

A opção estudada para definir a rota foi a de minimizar o custo total de passagens, fixando-se um tempo máximo de viagem. Como resultado, foi desenvolvido o seguinte modelo matemático de otimização inteira:

$$\min \sum_{i} \sum_{j} \sum_{t} c_{ijt} x_{ijt}$$

sujeito a:

$$\sum_{t=1}^{T} \sum_{i=1}^{N-1} x_{ijt} = 1 \qquad \forall j = 2, \ldots, N \qquad (1)$$

$$\sum_{t=1}^{T} \sum_{j=2}^{N} x_{ijt} = 1 \qquad \forall i = 1, \ldots, N-1 \qquad (2)$$

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} f x_{ijt} = \sum_{i=1}^{N-1} t_i \qquad \forall i = 1, \ldots, N-1; \forall j = 2, \ldots, N; \forall f = 1, \ldots, T \qquad (3)$$

$$t_j \geq t_i + t_{ijt} - (1 - x_{ijt})M \qquad \forall i = 1, \ldots, N-1; \forall j = 2, \ldots, N; \forall t = 1, \ldots, T \qquad (4)$$

$$t_i \geq f x_{ijt} \qquad \forall i = 1, \ldots, N-1; \forall j = 2, \ldots, N; \forall t, f = 1, \ldots, T \qquad (5)$$

$$t_i < t_N \qquad \forall i = 1, \ldots, N-1 \qquad (6)$$

$$t_1 \leq pd \qquad (7)$$

$$x_{ijt} \in \{0, 1\} \qquad \forall i = 1, \ldots, N-1; \forall j = 2, \ldots, N; \forall t = 1, \ldots, T \qquad (8)$$

$$t_i \in Z \qquad \forall i = 1, \ldots, N-1; \qquad (9)$$

Em que:
1: é o índice do lugar de origem da viagem;
$N$: é o número total de cidades a serem visitadas, que coincide com o índice do destino final, que é o mesmo local de origem;
$p$: é o número de períodos considerados por dia;
$d$: é o limite máximo de dias a mais em cada lugar;
$f$: representa o período na linha do tempo;
$T$: é o tempo máximo de viagem em número de períodos no horizonte de planejamento;

$M$: é um valor real suficientemente grande.

As desigualdades (1) e (2) restringem, respectivamente, que a cidade $i$ seja visitada por exatamente uma vez, sendo o ponto de partida para um outro destino apenas uma vez, e que a cidade $j$ seja visitada exatamente uma vez e, portanto, seja uma única vez o ponto de destino de um trajeto. As restrições (3) garantem que o tempo de chegada no último destino seja igual ao tempo de saída do penúltimo destino do trajeto. As restrições (4) garantem que o momento de saída de uma localidade $i$ para uma outra, $j$, na linha do tempo, seja superior ou igual ao momento em que se chegou em $i$ mais o número de períodos de estadia nessa localidade. As restrições (5) controlam, no horizonte de planejamento, o momento de saída $f$ da cidade $i$. As restrições (6) forçam que o valor de $t_N$ seja sempre maior que qualquer $t_i$, para garantir que $t_N$ seja o destino final, enquanto a restrição (7) faz com que a variável $t_1$ seja sempre menor que o limite máximo de dias de espera para cada lugar multiplicado pelo número total de períodos considerados por dia. E, por fim, as restrições (8) e (9) estabelecem, respectivamente, o domínio das variáveis $x_{ijt}$, que são sempre binárias, e o de $t_i$ que serão sempre valores naturais.

## 3. Métodos Propostos

Para a solução do *PARMDP*, foram propostos métodos baseados na metaheurística *Greedy Randomized Adaptive Search Procedure* (*GRASP*) (Feo and Resende, 1989). *GRASP* é uma metaheurística que tem sido bastante utilizada ao longo dos anos para resolver problemas de otimização combinatória (Vansteenwegen et al., 2011; Souffriau et al., 2010).

O *GRASP* é um método iterativo, no qual cada iteração consiste de duas fases: a primeira, construtiva, em que é construída uma solução inicial, e a segunda, que a partir da solução obtida na primeira fase, efetua uma busca local para melhorar a sua qualidade. O pseudocódigo do *GRASP* é apresentado no Algoritmo 1.

**Algoritmo 1:** *GRASP*

```
1  MelhorSoluçãoLista ← ∅;
2  CustoMínimo ← ∞;
3  it ← 0;
4  enquanto it < número de iterações faça
5      SoluçãoLista ← FaseConstrutiva();
6      se SoluçãoLista é factível então
7          SoluçãoLista ← BuscaLocal(SoluçãoLista);
8          se custoTotal(SoluçãoLista) < custoMínimo então
9              MelhorSoluçãoLista ← SoluçãoLista;
10             custoMínimo ← custoTotal(MelhorSoluçãoLista);
11     it++;
12 retorna MelhorSoluçãoLista;
```

### 3.1. *GRASP I*

As fases construtiva e de busca local da primeira versão do *GRASP*, aqui denominada de *GRASP I*, são apresentadas, respectivamente, nos Algoritmos 2 e 3.

**Algoritmo 2:** Fase Construtiva *GRASP I*

```
1 enquanto DestinosLista ≠ ∅ faça
2     enquanto t ∈ períodoVálido faça
3         L ← passagens com saída no período t;
4     ordenaCustoCrescente(L);
5     RCL ← trunca(L, α);
6     elemento ← sorteiaAleatório(RCL);
7     adiciona(elemento, SoluçãoLista);
8     remove(destinoElemento, DestinosLista);
9 retorna SoluçãoLista;
```

**Algoritmo 3:** Busca Local *GRASP I*

```
Entrada: SoluçãoLista
1 MelhorSoluçãoLocal ← SoluçãoLista;
2 enquanto existe 2-vizinhança ainda não feita na SoluçãoLista faça
3     SoluçãoTemporária ← BuscaLocal(SoluçãoLista);
4     se SoluçãoTemporária é factível então
5         se custoTotal(SoluçãoTemporária) < CustoMínimo então
6             MelhorSoluçãoLista ← SoluçãoTemporária;
7             CustoMínimo ← custoTotal(SoluçãoLista);
Saída: MelhorSoluçãoLocal
```

Pode-se observar no Algoritmo 1 que, por um determinado número de iterações, uma solução é gerada pela fase construtiva, à qual aplica-se uma busca local. Na fase construtiva, inicialmente, a solução encontra-se apenas com a origem, de onde se inicia a sequência de viagem (equivale a uma solução vazia). Então, essa solução é alterada ao se escolher um próximo destino válido (que não foi visitado) para aonde existe passagem a partir daquela localidade em um período válido, esperando-se, no máximo, por um dia a partir da data de quando o passageiro pode sair dessa localidade (considerando que ele ficou o tempo de estadia devido para aquela localidade). Todos os destinos e períodos válidos são armazenados em uma lista $L$, ordenada de acordo com os custos das passagens correspondentes. A partir dessa lista $L$, deve-se truncar seu tamanho de acordo com um $\alpha \in [0, 1]$ definido, gerando uma lista restrita de candidatos a entrar na solução, aqui representada pela $RCL$. Então, um dos $|RCL|$ destinos da $RCL$ é escolhido aleatoriamente e adicionado à solução parcial da iteração do *GRASP* até a mesma ser factível ou até que não seja mais possível encontrar um próximo destino no intervalo de tempo estabelecido. No primeiro caso, a solução encontrada passa para a fase de busca local. No segundo caso, ela é descartada.

Para a busca local, a solução é alterada de acordo com a sequência de destinos encontrada. Consideram-se soluções vizinhas, aquelas que se diferenciam em no máximo duas posições do vetor de sequências da solução original. Essa vizinhança é conhecida na literatura por 2-vizinhança. Vale observar que, ao trocar duas cidades de posição, é necessário rearranjar toda a sequência de passagens para garantir que os tempos de estadia sejam devidamente cumpridos. Entretanto, nem sempre, a sequência produzida pelo movimento da vizinhança gera soluções factíveis, mesmo após os rearranjos na sequência.

Após um experimento utilizando um grafo pequeno, foi observada uma certa dificuldade do *GRASP I* em encontrar soluções factíveis, tanto em sua fase construtiva quanto na fase de busca local. Por esse motivo, houve a necessidade de desenvolver uma segunda versão do algoritmo *GRASP*, aqui chamada de *GRASP II*.

### 3.2. *GRASP II*

Na tentativa de melhorar as soluções geradas pelo *GRASP I*, na segunda versão do algoritmo, chamado de *GRASP II*, foi gerado um grafo de acordo com a frequência de cada origem para cada destino das passagens em relação ao número total de passagens disponíveis nos períodos considerados para a busca. Esse grafo foi gerado para encontrar os possíveis *clusters* de cidades que possuem passagens com mais frequência (Fortunato and Castellano, 2008).

A fase construtiva do *GRASP II* é a mesma do *GRASP I*. O que diferencia um algoritmo do outro é a fase de busca local, já que no *GRASP II* uma troca de posição entre duas cidades é permitida se pelo menos uma das duas cidades se aproxima de cidades do mesmo *cluster*, caso contrário, essa troca é ignorada e a busca local continua. Apesar de uma melhoria na qualidade das soluções encontradas em relação ao *GRASP I*, uma outra versão do *GRASP* foi proposta, pois, a fase construtiva dessas duas versões ainda gerava muitas soluções inviáveis.

### 3.3. *GRASP III*

Para a terceira versão do algoritmo, aqui chamada de *GRASP III*, a fase de construção da solução é feita considerando os grupos gerados pelo grafo de frequências de saída dos destinos. Para produzir a *RCL*, considera-se apenas passagens para os destinos que pertençam ao mesmo grupo da cidade de saída, caso nem todos os destinos do grupo já tiverem sido visitados. A busca local para esse método se mantém como a busca local do *GRASP II*.

## 4. Experimentos

Esta seção apresenta alguns experimentos computacionais envolvendo dados reais de empresas aéreas de baixo custo da Europa. Todos os algoritmos foram implementados em linguagem Java utilizando conexão ao banco de dados *MySQL*. Os dados foram coletados a partir da ferramenta *Skypicker* (Dlouhy, 2012). Para todos os experimentos, foi utilizado um computador com as seguintes configurações: Mac OS X - Versão 10.6.8; Processador: 2.66GHz - Intel Core 2 Duo; Memória: 8GB - 1067Mhz DDR3. Os *GRASP I*, *II* e *III* foram testados de acordo com o número de iterações, que foram respectivamente, 1.000, 10.000 e 100.000. Em todos eles, o valor do parâmetro $\alpha$ do *GRASP* foi ajustado experimentalmente e, em todos os casos, foram consideradas 10 execuções independentes de cada *GRASP*. Para uma melhor avaliação da heurística, os testes também foram executados no pacote de otimização *CPLEX* (IBM, 2011).

### 4.1. Estudo de Caso: Origem e 6 Destinos

Esse estudo de caso possui origem em Milão, na Itália e mais 6 destinos finais, com 13 dias de estadia, no mínimo, divididos em cada um dos destinos, totalizando 78 períodos mais 42 períodos de folga (um dia a mais em cada destino), com um total de 120 períodos.

Os resultados desse experimento estão reportados na Tabela 2, cujas colunas 2, 3 e 4 representam, respectivamente, o menor tempo, o maior tempo e o tempo médio de execução do método indicado

na coluna 1; já as colunas 5, 6 e 7 indicam, respectivamente, a menor solução, a maior solução e a solução média encontrada pelo método indicado na coluna 1. É importante observar que o *CPLEX* não foi capaz de encontrar uma solução utilizando a modelagem apresentada na Seção 2, em máquina já descrita. Não foi encontrada uma solução factível para esse problema, havendo estouro de memória na construção da árvore de busca antes de atingir o limite de 3 horas de execução.

Tabela 2: Teste para estudo de caso com 7 destinos.

| | Tempo (s) | | | Preço (€) | | |
|---|---|---|---|---|---|---|
| Método | Menor | Maior | Média | Menor | Maior | Média |
| 1.000 iterações e alfa=0,9 | | | | | | |
| GRASP I | 0,04 | 0,72 | 0,15 | 308,68 | 401,27 | 355,17 |
| GRASP II | 0,04 | 0,87 | 0,18 | 308,68 | 393,83 | 334,54 |
| GRASP III | 0,09 | 0,94 | 0,26 | 303,68 | 308,68 | 308,18 |
| 10.000 iterações e alfa=0,9 | | | | | | |
| GRASP I | 0,31 | 1,13 | 0,48 | 308,68 | 325,41 | 310,35 |
| GRASP II | 0,25 | 1,86 | 0,66 | 308,68 | 308,68 | 308,68 |
| GRASP III | 0,57 | 3,27 | 0,87 | 303,68 | 303,68 | 303,68 |
| 100.000 iterações e alfa=0,9 | | | | | | |
| GRASP I | 18,2 | 22,2 | 19,9 | 303,68 | 326,5 | 318,41 |
| GRASP II | 13,8 | 17,6 | 14,8 | 308,68 | 327,33 | 319,46 |
| GRASP III | 14,1 | 18,0 | 14,7 | 303,68 | 303,68 | 303,68 |

### 4.2. Estudo de Caso: Origem e 13 Destinos

Esse estudo de caso possui origem em Milão, na Itália e mais 13 destinos finais, com 35 dias de estadia, no mínimo, divididos em cada um dos destinos, totalizando 210 períodos mais 84 períodos de folga (um dia a mais em cada destino), com um total de 294 períodos. Os resultados desse experimento estão reportados na Tabela 3.

Tabela 3: Teste para estudo de caso com 14 destinos.

| | Tempo (s) | | | Preço (€) | | | Infactibilidade |
|---|---|---|---|---|---|---|---|
| Método | Menor | Maior | Média | Menor | Maior | Média | Porcentagem |
| 1.000 iterações e alfa=0,9 | | | | | | | |
| GRASP I e II | - | - | - | - | - | - | 100% |
| GRASP III | 0,5 | 2,1 | 1,1 | 760,63 | 791,86 | 779,84 | 70% |
| 10.000 iterações e alfa=0,9 | | | | | | | |
| GRASP I e II | - | - | - | - | - | - | 100% |
| GRASP III | 0,21 | 1,50 | 0,39 | 715,45 | 896,23 | 776,83 | 0 |
| 100.000 iterações e alfa=0,9 | | | | | | | |
| GRASP I | 9,61 | 11,57 | 9,84 | 861,05 | 1068,94 | 966,71 | 70% |
| GRASP II | 9,46 | 11,51 | 9,70 | 774,76 | 930,83 | 823,72 | 60% |
| GRASP III | 2,00 | 3,24 | 2,17 | 674,04 | 761,77 | 714,63 | 0 |

Esse estudo de caso não chegou a ser executado no *CPLEX* por inviabilidade. É possível perceber que para 13 destinos, a execução de 1.000 iterações não teve um desempenho tão bom, encontrando resultados em apenas 30% dos testes realizados para o *GRASP III*. Com 10.000 iterações, são obtidos resultados em todos os testes feitos para o *GRASP III*, porém, ainda com 100% de infactibilidade para os métodos *GRASP I* e *GRASP II*, o que demonstra que a separação dos *clusters* na fase construtiva e na busca local influenciam na busca por soluções factíveis. Com 100.000 iterações, obtém-se alguns poucos resultados para o *GRASP I* e o *GRASP II*, respectivamente, 30% e 40% de soluções factíveis para os métodos. Com esses resultados, pode-se notar que o algoritmo *GRASP III* foi o que encontrou melhores resultados nesse caso de teste, inclusive, com o menor custo computacional.

## 5. Conclusões

Este artigo apresentou um estudo do Problema de Planejamento de Rotas Aéreas com Múltiplos Destinos (*PARMDP*). Como parte do trabalho, foi desenvolvida uma modelagem matemática e, devido à dificuldade do *CPLEX* em encontrar soluções factíveis para o problema segundo essa modelagem, foram propostas três versões da metaheurística *GRASP* para resolver o *PARMDP*, aqui chamadas de *GRASP I*, *GRASP II* e *GRASP III*. Essas versões surgiram devido ao conhecimento adquirido após observar os estudos de caso. Essas observações foram sobre uma base de dados reais, na qual diversas características inerentes ao problema como a frequência de voos entre certos trechos e preços de passagens entre os diferentes destinos foram observadas. Como consequência dessas observações, grupos bem conectados de frequências de voos foram encontrados por meio de um algoritmo de agrupamento em grafos e tais grupos foram usados como informação de entrada nas versões II e III da metaheurística. No experimento computacional, destaca-se o *GRASP III* que apresentou melhor qualidade de soluções em um tempo bem inferior às outras versões.

## Referências

Ambite, J. L., Muslea, G. B. C. A. K. M., Oh, J., and Minton, S., editors (2002). *Getting from here to there: Interactive planning and agent execution for optimizing travel*, /AAAI-02.

**Dlouhy, O.** (2012). Skypicker.

**Feo, T. A. and Resende, M. G.** (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, /8:67–71.

**Fortunato, S. and Castellano, C.** (2008). Community structure in graphs. chapter of Springer's, Encyclopedia of Complexity and System Science.

**IBM** (2011). *User's Manual for CPLEX*.

**Souffriau, W., Vansteenwegen, P., Vanden Berghe, G., and Van Oudheusden, D.** (2010). A path relinking approach for the team orienteering problem. *Computers & Operations Research*, /37:1853–1859.

**Vansteenwegen, P., Souffriau, W., Berghe, G. V., and Oudheusden, D. V.** (2011). The city trip planner: An expert system for tourists. *Expert Systems with Applications*, /38:6540–6546.