**UNIVERSIDADE FEDERAL DA PARAÍBA PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA**

Alexander de Almeida Pinto

# UTILIZAÇÃO DAS METAHEURÍSTICAS GRASP E ILS COM BUSCA LOCAL EXATA PARA RESOLUÇÃO DO PROBLEMA DE CONSTRUÇÃO DE TRILHOS DE AERONAVES

Dissertação submetida ao Programa de Pós-Graduação em Informática da Universidade Federal da Paraíba para a obtenção do grau de Mestre em Informática.
Orientador: Prof. Dr. Lucídio dos Anjos Formiga Cabral

João Pessoa

2012

Alexander de Almeida Pinto

**UTILIZAÇÃO DAS METAHEURÍSTICAS GRASP E ILS COM BUSCA LOCAL EXATA PARA RESOLUÇÃO DO PROBLEMA DE CONSTRUÇÃO DE TRILHOS DE AERONAVES**

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Informática e aprovada em sua forma final pelo Programa de Pós-Graduação em Informática da Universidade Federal da Paraíba.
João Pessoa, 23/03/2012.

---

Tatiana Aires Tavares, Dra.
Coordenadora do Curso

**Banca Examinadora:**

---

Lucídio dos Anjos Formiga Cabral, Dr.
Orientador

---

Roberto Quirino do Nascimento, Dr., UFPB

---

Lúcia Maria de Assumpção Drummond, Dra., UFF

*Dedico este trabalho a minha família que me ajudou em todos os momentos que precisei.*

*Se você continua vivo é porque ainda não chegou aonde devia.*

Albert Einstein

## AGRADECIMENTOS

Fazer um agradecimento que inclua todos que me ajudaram a construir essa dissertação não é uma tarefa fácil, principalmente por se correr o risco de esquecer de mencionar alguém.

No âmbito acadêmico devo agradecer principalmente a excelência profissional do Dr. Lucídio dos Anjos Formiga Cabral que conferiu prestígio e valor a meu trabalho de mestrado além de ser um amigo de grande valia. Também gostaria de fazer um agradecimento especial ao Roberto Pontes que teve uma grande participação nesse projeto com idéias que foram resgatadas do seu trabalho relizado na COPPE/RioSul. E também aos demais professores que me acompanharam durante a graduação e o mestrado, me ensinando conceitos importantíssimos.

Agradeço também a meus amigos que, de uma forma ou de outra, contribuíram com sua amizade e com sugestões efetivas para a realização deste trabalho, principalmente a Daniel Gonçalves, Gilberto Farias e Rennan Toscano. Gostaria de expressar minha profunda gratidão.

Não poderia esquecer de meus pais, que me deram contínuo apoio em todos esses anos, ensinando-me os valores que irei levar durante toda minha vida e que acreditaram na minha capacidade de realização tornando-se assim os elementos propulsores desta dissertação.

Agradeço também ao Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, pela bolsa concedida durante os anos do curso.

## RESUMO

Os problemas operacionais cresceram muito em complexidade nos últimos tempos, o que tem acentuado a necessidade do desenvolvimento de técnicas que possam agilizar os processos de tomada de decisão.

Este trabalho trata da etapa de geração dos trilhos de aeronaves, ou seja, o sequenciamento de voos de cada aeronave. O objetivo aqui é minimizar o número de aeronaves necessárias para operar uma dada malha de voos.

Esse problema possui é combinatoriamente explosivo e a sua resolução fica mais difícil à medida que a quantidade de voos envolvidos cresce. Pequenas modificações nos horários de partida desses voos, ou o acréscimo de algum voo de resposicionamento entre dois aeroportos próximos podem gerar soluções de melhor qualidade.

Nós apresentamos um algoritmo híbrido baseado na metaheurística GRASP, com a utilização do ILS e de programação inteira na busca local. Os resultados tem mostrado que essa abordagem é capaz de gerar boas soluções.



**Palavras-chave**: Transporte, PCTA, Metaheurística, Método Exato,GRASP, Rotas e Aeronaves.

## ABSTRACT

Operational research problems has growing in complexity in the last years, this has accentuated the necessity to develop of techniques witch can accelerate the process of decision making.

This work covers the step of aircraft rotation problem, i.e., the sequencing of flights for each aircraft. The goal here is minimize the number of aircraft required to operate a given network of flights.

This problem is combinatorial and it resolution is more difficult when the number of involved flights grows. However small changes in departure time, or the addition of a repositioning flight between two nearby airports can reduce the cost of solutions.

We present a hybrid algorithm based on the metaheuristic GRASP, using the ILS and integer programing in the local search.

The results has shown which this approach can generate good solutions.



**Keywords**: Transportation, ARP, metaheuristic, Exact Method, GRASP,Aircraft Routing.

## LISTA DE FIGURAS


1.1 Rotas de voo da companhia aérea TAM
Fonte: (http://www.airlineroutemaps.com) . . . . . 19

1.2 Malha hub-and-spoke.
Fonte: (Própria) . . . . . 20

2.1 Pseudocódigo da heurística de construção gulosa de uma solução inicial.
Fonte: (SOUZA, 2009) . . . . . 24

2.2 Pseudocódigo da heurística de construção aleatória de uma solução inicial.
Fonte: (SOUZA, 2009) . . . . . 25

2.3 Pseudocódigo do procedimento GRASP.
Fonte: (FEO; RESENDE, 1995) . . . . . 27

2.4 Pseudocódigo do procedimento de construção do GRASP para um problema de minimização.
Fonte: (FEO; RESENDE, 1995) . . . . . 28

2.5 Pseudocódigo do procedimento de busca local do GRASP.
Fonte: (FEO; RESENDE, 1995) . . . . . 30

2.6 Pseudocódigo do procedimento Iterated Local Search.
Fonte: (SOUZA, 2009) . . . . . 31

2.7 Representação esquemática do funcionamento do ILS.
Fonte: (SOUZA, 2009) . . . . . 32

4.1 Representação esquemática do arco do tipo 1.
Fonte: (Própria) . . . . . 40

4.2 Representação esquemática do arco do tipo 2.

Fonte: (Própria) . . . . . 40

4.3 Representação esquemática do arco do tipo 3.

Fonte: (Própria) . . . . . 41

4.4 Representação esquemática do arco do tipo 4.

Fonte: (Própria) . . . . . 41

4.5 Construção de Trilhos de Aeronaves.

Fonte: (Própria) . . . . . 42

5.1 Arcos necessários para ligar dois voos.

Fonte: (Própria) . . . . . 45

5.2 Conversão de um voo para ser utilizado no solver.

Fonte: (Própria) . . . . . 49

5.3 Pseudocódigo do procedimento de seleção de um voo inicial.

Fonte: Própria . . . . . 51

5.4 Pseudocódigo do procedimento de formação sequencial dos trilhos.

Fonte: Própria . . . . . 52

5.5 Pseudocódigo do procedimento de formação paralela dos trilhos.

Fonte: Própria . . . . . 53

5.6 Pseudocódigo de calculo do proximo voo de um trilho

Fonte: Própria . . . . . 56

5.7 Estrutura de vizinhança Swap-1.

Fonte: (Própria) . . . . . 57

5.8 Estrutura de vizinhança CrossOver.

Fonte: (Própria) . . . . . 58

5.9 Estrutura de vizinhança Compactação.
Fonte: (Própria) . . . . . 58
5.10 Pseudocódigo do método de comunicação paralela usado na construção dos trilhos.
Fonte: Própria . . . . . 62

## LISTA DE TABELAS

6.1 Parametrização dos cenários . . . . . 66
6.2 Resultados do otimizador no cenário 1 . . . . . 67
6.3 Resultados do otimizador no cenário 2 . . . . . 67
6.4 Resultados da metaheurística pura no cenário 1 . . . . . 68
6.5 Resultados da metaheurística pura no cenário 2 . . . . . 68
6.6 Resultados da metaheurística híbrida no cenário 1 . . . . . 69
6.7 Resultados da metaheurística híbrida no cenário 2 . . . . . 69

## SUMÁRIO

**1 INTRODUÇÃO** ..... 14
1.1 OBJETIVOS DO TRABALHO ..... 21
**1.1.1 Geral** ..... 21
**1.1.2 Específicos** ..... 21
1.2 ORGANIZAÇÃO DA DISSERTAÇÃO ..... 22
**2 FUNDAMENTAÇÃO TEÓRICA** ..... 23
2.1 HEURÍSTICAS CONSTRUTIVAS ..... 23
2.2 METAHEURÍSTICA ..... 24
**2.2.1 GRASP** ..... 26
**2.2.2 ILS** ..... 31
2.3 PROGRAMAÇÃO LINEAR ..... 33
**3 REVISÃO DA LITERATURA** ..... 35
**4 DESCRIÇÃO DO PROBLEMA** ..... 38
**5 MÉTODO PROPOSTO** ..... 44
5.1 MODELO MATEMÁTICO ..... 45
**5.1.1 Definição** ..... 46
5.2 FUNÇÃO OBJETIVO ..... 46
5.3 RESTRIÇÕES ..... 47
5.4 FASE DE CONSTRUÇÃO DO GRASP ..... 49
**5.4.1 Formação dos trilhos de forma sequencial** ..... 50
**5.4.2 Formação dos trilhos de forma paralela** ..... 51
**5.4.3 Escolha dos voos de um trilho** ..... 53

5.5 FASE DE BUSCA LOCAL . . . . . 55
**5.5.1 Estruturas de vizinhança** . . . . . 56
Swap-X . . . . . 57
Cross-Over . . . . . 57
Compactação . . . . . 58
**5.5.2 Perturbação usando o método exato** . . . . . 58
**5.5.3 Método de construção paralela** . . . . . 60
**6 RESULTADOS** . . . . . 64
6.1 AMBIENTE DE TESTES . . . . . 64
6.2 RESULTADOS DO OTIMIZADOR . . . . . 67
6.3 RESULTADOS DA METAHEURÍSTICA PURA . . . . . 68
6.4 RESULTADOS DA METAHEURÍSTICA HÍBRIDA . . . . . 69
6.5 CONSIDERAÇÕES FINAIS . . . . . 69
**Referências bibliográficas** . . . . . 72
**Anexo A – Rede de voos da Rio-Sul** . . . . . 73
**Anexo B – Tempo dos voos da Rio-Sul** . . . . . 75
**Anexo C – Resultado ótimo da instância Rio-Sul** . . . . . 76
**Anexo D – Rede de voos da TAM** . . . . . 78
**Anexo E – Tempo dos voos da TAM** . . . . . 81
**Anexo F – Resultado ótimo da instância TAM** . . . . . 82

# 1 INTRODUÇÃO

A aviação é o principal meio de transporte de pessoas e de mercadorias capaz de atravessar grandes distâncias. Ela é fundamental para o crescimento da economia e do turismo mundial, contribui para a melhora da qualidade de vida das pessoas proporcionando lazer e experiências com outras culturas. O uso da aviação comercial tem crescido significativamente nas últimas décadas e a previsão é que esse aumento seja cada vez maior, pelo menos no Brasil, que está fazendo grandes investimento para ampliar e melhorar a infraestrutura aeroportuária com a finalidade de atender a grande demanda que é esperada para a Copa do Mundo de 2014 e para as Olimpíadas de 2016. Além disso, a economia do país está crescendo e provocando um aumento da renda e da qualidade de vida, incentivando as pessoas a viajar e explorar novas oportunidades.

Esse ambiente favorável tem provocado o aumento do número de companhias aéreas, fazendo com que ocorra uma maior competição, isso faz com que se acentue a necessidade de otimizar a utilização dos recursos disponíveis, reduzindo os custos para poder continuar a crescer e oferecer tarifas mais competitivas. Porém, os problemas presentes na indústria aeronáutica são complexos envolvendo múltiplas decisões conflitantes que precisam ser otimizadas em conjunto. Diversas técnicas são desenvolvidas e utilizadas para tentar melhorar o planejamento e a operação das empresas aéreas. Muitas dessas técnicas estão disponíveis na literatura científica, nos campos da pesquisa operacional e da matemática. Normalmente, essas técnicas são modeladas para funcionar em sistemas computadorizados de alta capacidade, com a finalidade de automatizar ou, pelo menos, auxiliar na tomada de decisões. Essas técnicas se tornam necessárias à medida que a empresa aérea cresce e o processo de tomada de decisão, baseada nos julgamentos

individuais e nas experiências se torna mais difícil (ABDELGHANY; ABDELGHANY, 2009).

Os principais problemas relacionados dizem respeito ao planejamento, envolvendo a criação de linhas de trabalho (sequência de voos) tanto para as aeronaves quanto para a tripulação. O objetivo costuma ser a minimização dos custos operacionais ou a maximização dos rendimentos. Custos operacionais consistem, por exemplo, nos custos envolvidos com combustíveis, óleo e taxas de aterrissagem. Também pode ser levado em consideração a perda de rendimentos com a utilização de aeronaves com menos assentos do que a demanda de passageiros, provocado pelo mal dimensionamento da demanda. Fatores como o bem estar dos passageiros também pode ser levado em consideração, provocando por exemplo uma redução na quantidade de conexões e de escalas.

Para que seja possível ter uma visão geral do contexto é necessário descrever os problemas que estão associados com a construção de trilhos de aeronaves. O primeiro dos problemas, que será mencionado aqui, é a modelagem de mercado que tem como objetivo fazer o levantamento da quantidade de passageiros que tem interesse em viajar entre as localidades levando em consideração o espaço de tempo desejado. Adicionalmente pode-se criar demandas através da criação de novas rotas com o auxilio de propagandas e promoções. Essa demanda normalmente apresenta uma grande variedade dependendo da época do ano e o levantamento desses dados de forma errada pode causar grandes prejuízos. É importante categorizar o tipo de passageiro para poder fazer escolhas futuras que não levem em consideração apenas fatores quantitativos.

Posterior a modelagem de mercado, tem-se a necessidade de definir qual equipamento (frota) irá operar cada trecho definido pela demanda (PIMENTEL, 2005), esse problema é conhecido como a atribuição de frota (*Fleet Assigment Problem*) . A definição

de um equipamento errado pode provocar a subutilização da aeronave o que provocaria prejuízos ou a superutilização que provocaria a perda de clientes para a concorrência. Ao final dessa etapa irão ser formados conjuntos de voos que deverão ser operados por tipos de aeronaves específicas. É a partir desses conjuntos que deverá ser feita a construção dos trilhos das aeronaves.

Um trilho de aeronave é a sequência de voos que uma aeronave deve operar em um determinado período, e o problema de construção desses trilhos (*Aircraft Rotation Problem*) tem como principal objetivo a redução do número de aeronaves necessárias para operar todos os voos e como objetivo secundário fazer o menor número de modificações no planejamento inicial desses voos.

O problema seguinte utiliza-se das sequências de voos das aeronaves para construir o melhor conjunto de *pairings*[1] de forma que cada voo seja coberto por pelo menos um pairing. Gastos com alojamentos, alimentação, transporte em terra e *deadheads*[2] devem ser levados em consideração. A partir desse conjunto é feita a escala dos tripulantes, (*Crew Scheduling Problem*) que atribui os pairings à tripulação acrescentando as atividades de solo, tais como *Call Time*[3], *Stand-by duties*[4] e os dias de descanso. O objetivo dessa última etapa é fazer uma distribuição da forma mais justa possível, tentando balancear a quantidade de trabalho (horas a serem voadas) entre os tripulantes e também tentar cumprir suas preferências sem violar nenhuma restrição da legislação trabalhista

[1] Pairing é o conjunto de voos que pode ser operados por uma tripulação sem que seja violada qualquer regra da legislação vigente e que ao final do último voo o tripulante esteja de volta a sua cidade base.

[2] Deadhead é o voo que o tripulante viaja sem trabalhar, com a finalidade de transporte para outra localidade normalmente para sua base ou para suprir uma nova demanda, o deadhead pode ser operado por aviões de outras companhias, nesse caso o custo da viagem é maior.

[3] Call time é o tempo que a tripulação tem para se apresentar a companhia aérea antes de iniciar de fato seu turno de trabalho.

[4] Stand-by duties são turnos em que o tripulante fica a disposição da companhia aérea a fim de suprir possíveis eventualidades.

em vigor.

Esse trabalho mostra uma forma eficiente de resolver o Problema de Construção de Trilhos de Aeronaves (PCTA) que também é conhecido na literatura como Aircraft Rotation Problem (ARP) . O PCTA é um dos principais problemas presentes na industria da aviação e seu objetivo é fazer o sequenciamento dos voos de cada frota da companhia de forma que seja possível operá-las com o menor número de aeronaves possíveis (LUCENA; PONTES, 2007) bem como efetuar a menor modificação possível no planejamento inicial dos voos. Cada sequência de voos recebe o nome de trilho de aeronave e o conjunto desses trilhos é denominado de malha aérea.

Para resolver o PCTA foram aplicadas metaheurísticas combinadas com método exato de *branch-and-bound* através da resolução de um modelo matemático de programação linear inteira simplificado. Essa estratégia foi escolhida pois com os computadores atuais não é possível obter resultados apenas com a aplicação do modelo exato, que retorna o resultado ótimo, pois o tempo necessário para resolver instâncias semanais do PCTA já leva um tempo inviável. A utilização apenas de metaheurística foi levada em consideração no início, porém nos resultados práticos a qualidade da solução se mostrou muito aquém da desejada. Dessa forma surgiu a idéia de juntar as duas técnicas, com a finalidade de obter a convergência do método exato e a velocidade da metaheurística.

Na literatura tem-se observado um crescimento no número de trabalhos que se utilizam de metaheurísticas híbridas como método de resolução de problemas complexos de otimização combinatória apresentando soluções de alta qualidade . Esse fato também aconteceu na resolução do PCTA.

Nos dias de hoje, com o avanço da tecnologia e o aumento da competitividade,

desenvolver soluções com melhor qualidade acaba se tornando um fator decisivo para a permanência no mercado, tornando-se então necessária a obtenção de soluções de forma mais rápida, mais barata e utilizando menos recursos. Por causa da dificuldade que é inerente a essa classe de problemas, a qualidade das soluções obtidas manualmente são muito inferiores em relação à melhor solução possível. Outra característica que reforça a necessidade da obtenção de melhores soluções é o aumento do tamanho e da complexidade das instâncias trabalhadas. A partir desse cenário pode-se perceber a necessidade de utilização de técnicas de otimização.

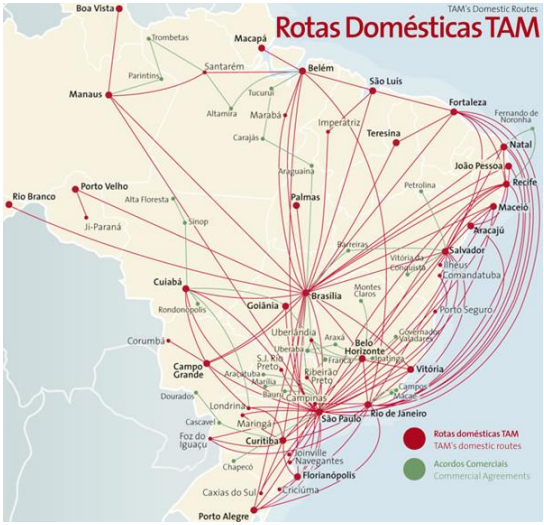
Em relação ao PCTA a literatura pesquisada (ABDELGHANY; ABDELGHANY, 2009) (ARGUELLO; BARD, 1997) (CORDEAU *et al.*, 2001) (HAOUARI *et al.*, 2011) (LUCENA; PONTES, 2007) mostra uma grande quantidade de tentativas de resolver o problema utilizando modelagens matemáticas, que apesar de garantir a solução ótima não se mostra viável para resolver grandes instâncias. Alguns trabalhos mostram a similaridade desse problema com o problema do caixeiro viajante assimétrico (CLARKEA *et al.*, 1997). E outros resolvem uma parte do problema utilizando metaheurísticas (ARGUELLO; BARD, 1997). Além disso, a pesquisa constatou que a literatura acerca do PCTA não disponibiliza as instâncias que foram trabalhadas dificultando assim a comparação dos resultados obtidos com esses trabalhos. Logo um dos objetivos desse trabalho é disponibilizar um conjunto de instâncias.

Atualmente existem diversas fontes na qual se podem obter instâncias para problemas de otimização combinatória sendo uma das mais conhecidas a OR-Library [5] que foi descrito inicialmente em J.E.Beasley (BEASLEY, 1990) permitindo o acesso a centenas de conjuntos de instâncias a partir da Internet.

[5] A OR-Library pode ser acessado em http://people.brunel.ac.uk/ mastjjb/jeb/info.html

Apesar da existência dessa entidade apenas uma instância, a da Rio Sul, foi encontrada em um relatório técnico da UFPB. Com a finalidade de ter outra instância, foi feito na web a coleta dos dados dos voos operados pela compahia denominada TAM (http://www.tam.com.br). Esse levantamento foi feito através de uma pesquisa observativa fazendo-se o comparativo da malha operada pela companhia, que pode ser visto na Figura 1.1, e as possibilidades de voos diretos do equipamento AirBus Industrie A310, que formava a maior frota da empresa.

**Figura 1.1: Rotas de voo da companhia aérea TAM**
**Fonte: (http://www.airlineroutemaps.com)**



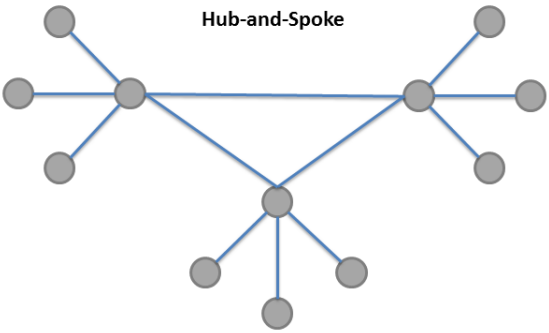

A malha da Rio Sul se refere a uma instância com voos de um dia de operação da empresa contendo 107 voos, onde na prática os voos eram operados com 20 trilhos obtidos pela montagem manual de um experiente operador (PONTES *et al.*, 2002). A malha da TAM, que foi obtida manualmente, é formada por 241 voos possuindo uma grande quantidade de ligações entre os 31 aeroportos envolvidos, dessa forma essa instância

obteve um maior grau de complexidade.

Essas instâncias foram estendidas para o período de uma semana para testar o comportamento do algoritmo e do solver, onde a instância da Rio Sul estendida apresentou 749 voos e a da Tam estendida ficou com 1687 voos.

Levou-se em consideração apenas instâncias de companhias aéreas brasileiras pois as grandes empresas globais apresentam uma malha com característica *hub-and-spoke*, como pode ser visto na Figura 1.2. Uma malha é caracterizada como sendo do tipo *hub-and-spoke* se ela apresentar uma grande concentração de voos em poucos aeroportos e adicionalmente se um voo tem ligação com um *hub* ele não poderá ter ligação com outros *hubs* a não ser que ele também seja um *hub*. Instâncias com essa caractéristicas são mais fáceis de serem resolvidas pois não apresentam a caractéristica explosiva no nível presente nas malhas aéreas brasileiras.

**Figura 1.2: Malha hub-and-spoke.**
**Fonte: (Própria)**




Não foi possível obter mais instâncias reais pois a comunicação entre essas empresas e a acadêmia é fraca. Grande parte dessa falta de comunicação provavelmente se

deve ao receio de revelar dados que podem vir a lhes prejudicar junto a concorrência.

As instâncias podem ser vistas nos anexos ao final desse trabalho.

## 1.1 OBJETIVOS DO TRABALHO

### 1.1.1 Geral

Tendo em vista os aspectos apresentados, o objetivo principal dessa dissertação consiste no desenvolvimento de um método híbrido baseado nas metaheurísticas GRASP e ILS e em programação linear inteira para a resolução do problema construção de trilhos de aeronaves (PCTA) cobrindo todos os voos planejados com o menor número de aeronaves bem como efetuando a menor mudança possível no planejamento inicial desses voos.

### 1.1.2 Específicos

Esse trabalho também visa a disponibilização de um conjunto de instâncias com os resultados que foram obtidos. Isso irá permitir a sua comparação com outros trabalhos no futuro.

O método proposto consegue acelerar a convergência da fase de busca local através de uma busca exata em uma vizinhança restrita usando programação linear inteira. Adicionalmente esse método permite escapar de minimos locais.

Para conseguir melhores resultados foi permitida, em alguns cenários, a utilização de pequenas alterações no horário de partida sugerido dos voos bem como a criação de voos de reposicionamento.

## 1.2 ORGANIZAÇÃO DA DISSERTAÇÃO

O restante desse trabalho está estruturado da seguinte forma:

- Capítulo 2: Apresenta a fundamentação sobre otimização, metaheurísticas e programação linear inteira. A seção referente às metaheurísticas inicia com uma descrição, seguida do detalhamento das metaheurísticas utilizadas no trabalho, o GRASP e o ILS.
- Capítulo 3: Descreve o problema, explicando os conceitos que são utilizados no trabalho.
- Capítulo 4: Introduz o modelo matemático que foi desenvolvido.
- Capítulo 5: Descreve o método proposto nesse trabalho, mostrando como foi feita a integração das metaheurísticas e da programação linear inteira e também descreve os parâmetros e as restrições que foram utilizadas.
- Capítulo 6: Apresenta os resultados obtidos com o algoritmo e dá diretrizes de como utilizar o método proposto.
- No final são apresentados a bibliográfia e os anexos que contém um maior detalhamento das instâncias e dos resultados obtidos.

# 2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo é feita a fundamentação dos principais assuntos presentes nesse trabalho: a heurística construtiva, as metaheurísticas e a programação linear inteira. Nas seções seguintes são descritos os aspectos teóricos e os principais métodos relacionados a esse trabalho.

## 2.1 HEURÍSTICAS CONSTRUTIVAS

As técnicas de resolução heurísticas se utilizam de processos intuitivos com a finalidade de obter uma boa solução, a um custo computacional aceitável, ou seja, não garante a otimalidade da solução de um problema. O objetivo é obter, em um tempo reduzido, uma solução tão próxima quanto possível do ótimo global.

Uma heurística é dita construtiva quando a construção da solução se dá elemento por elemento. A forma de escolha dos elementos varia de acordo com a estratégia e a função de avaliação adotada, essa escolha deve levar em consideração o benefício da inserção de cada elemento para a qualidade da solução final, escolhendo sempre o *melhor* elemento em cada passo.

O Algoritmo 2.1 mostra o pseudocódigo para a construção de uma solução inicial para um problema de otimização que utiliza uma função gulosa $g(.)$. Nesta figura, $t_{melhor}$ indica o membro do conjunto de elementos candidatos com o valor mais favorável da função de avaliação $g$, isto é, aquele que possui o menor valor de $g$ no caso de o problema ser de minimização ou o maior valor de $g$ no caso de o problema ser de maximização.

Uma outra forma de obter uma solução inicial é escolher os elementos candidatos

**Figura 2.1: Pseudocódigo da heurística de construção gulosa de uma solução inicial. Fonte: (SOUZA, 2009)**

**Algoritmo** *ConstrucãoGulosa*$(g(.), s)$
1 s $\leftarrow \emptyset$;
2 Inicialize o conjunto $C$ de candidatos;
3 **enquanto** $C \neq \emptyset$ **faça**
4 $\quad g(t_{melhor}) = melhor\{g(t) \mid t \in C\}$;
5 $\quad s \leftarrow s \cup \{t_{melhor}\}$;
6 $\quad$ Atualize o conjunto $C$ de elementos candidatos;
7 **fim enquanto**
8 **devolva** $s$;
**fim**

aleatoriamente. Isto é, a cada passo, o elemento a ser inserido na solução é aleatoriamente selecionado dentre os elementos candidatos ainda não selecionados. A grande vantagem desta metodologia reside na simplicidade de implementação. Segundo testes empíricos a desvantagem é a baixa qualidade, em média, da solução final. Essa técnica é recomendada quando a característica do problema torna mais fácil o refinamento do que a construção de uma solução (SOUZA, 2009).

O Algoritmo 2.2 mostra o pseudocódigo para a construção de uma solução inicial aleatória para um problema de otimização.

Para melhores resultados, essa etapa deve ser seguida de um refinamento, pois a solução, quando gerada aleatoriamente, não costuma ser de boa qualidade.

## 2.2 METAHEURÍSTICA

A utilização de métodos exatos para a resolução de problemas reais envolvendo otimização combinatória é restrito. Isso acontece pois com o aumento da complexidade das instâncias envolvidas, o número de soluções possíveis cresce exponencialmente, fa-

**Figura 2.2: Pseudocódigo da heurística de construção aleatória de uma solução inicial. Fonte: (SOUZA, 2009)**

**Algoritmo** *ConstrucãoAleatória*$(g(.), s)$
1 s $\leftarrow \emptyset$;
2 Inicialize o conjunto $C$ de candidatos;
3 **enquanto** $C \neq \emptyset$ **faça**
4 Escolha aleatoriamente $t_{escolhido} \in C$;
5 $s \leftarrow s \cup \{t_{escolhido}\}$;
6 Atualize o conjunto $C$ de elementos candidatos;
7 **fim enquanto**
8 **devolva** $s$;
**fim**

zendo com que as operações necessárias para a sua resolução não possa ser feita com os computadores atuais em tempo viável.

Para contornar essa limitação e obter soluções para esses tipos de problemas, os pesquisadores desenvolveram técnicas que são capazes de guiar o procedimento de busca e assim encontrar boas soluções (FEO; RESENDE, 1995). Esses algoritmos, denominados heurísticas, encontram essas soluções utilizando pouco recursos computacionais, porém não garantem a solução ótima do problema (DIAS, 2006). Na prática, geralmente, uma boa solução é suficiente, já que a tomada de decisão tem que acontecer em um curto espaço de tempo.

As metaheurísticas são aplicadas a uma maior gama de problemas e surgiu para suprir algumas deficiências das heurísticas. Para contornar essas deficiências, foram desenvolvidas técnicas mais generalistas que foram denominadas de metaheurísticas. A metaheurística pode ser definida como um método heurístico para resolver de forma genérica problemas de otimização com a capacidade de escapar de ótimos locais. A idéia utilizada, normalmente, é obtida de algum evento natural como sistemas biológicos, da

física, da inteligência artificial entre outros.

As metaheurísticas podem explorar o espaço de soluções basicamente de duas formas: as de busca local e as de busca populacional. Nas metaheurísticas de busca local, o procedimento de busca utiliza uma solução como ponto de partida em cada iteração. As metaheurísticas GRASP, recozimento simulado (*Simulated Annealing*), busca tabu e ILS podem ser citadas como exemplos de metaheurísticas ponto-a-ponto. Nas metaheurísticas de busca populacionais, soluções de boa qualidade são combinadas com o intuito de produzir soluções melhores. Podemos citar como exemplo de métodos populacionais, os algoritmos genéticos, colônia de formigas (*Ant Colony System*), núvem de particulas (*Particle Swarm Optimization*) e etc (ARAUJO, 2009).

Nesse trabalho foram utilizadas as metaheurísticas de busca local GRASP e ILS de forma híbrida. As próximas seções descrevem essas metaheurísticas.

### 2.2.1 GRASP

Essa seção descreve a metaheurística GRASP (*Greedy Randomized Adaptive Search Procedure* - Procedimento de busca adaptativa gulosa e randômica), que foi proposta por (FEO; RESENDE, 1995), e cujos conceitos serão utilizados na metodologia proposta para resolução do PCTA. A metaheurística GRASP é um método iterativo do tipo *multi-start* formado por duas fases: uma fase de construção de uma solução e outra de busca local. A fase de construção objetiva gerar uma solução viável para o problema proposto. A fase de busca local tem como meta encontrar um ótimo local na vizinhança da solução construída. A melhor solução encontrada, ao longo de todas as iterações GRASP realizadas, é retornada.

O pseudo-código descrito no Algoritmo 2.3 ilustra um procedimento GRASP para um problema de minimização. Na linha 1 o custo da função objetivo da melhor solução encontrada é inicializada com ∞(infinito). A linha 2 repete o procedimento de construção e refinamento *GRASPMax* vezes. Por causa dessa etapa que o GRASP é considerado *multi-start*.

Nas linhas 3 e 4 são feitas, respectivamente, a construção e a busca local, que são apresentadas nos Algoritmos 2.4 e 2.5 e serão detalhadas mais adiante.

Nas linhas 5 a 8, se a solução obtida na busca local for melhor que a melhor solução obtida até o momento ($f(s) < f*$), então são atualizados respectivamente a solução e o custo relativo a função objetivo da melhor solução corrente. A linha 9 encerra as iterações do GRASP e a linha 10 retorna a melhor solução obtida na execução do algoritmo.

**Figura 2.3: Pseudocódigo do procedimento GRASP.**
**Fonte: (FEO; RESENDE, 1995)**

**Algoritmo** *GRASP*$(f(.), g(.), N(.), GRASPMax, s)$
1 f* $\leftarrow\infty$;
2 **para** $1, 2, ..., GRASPMax$ **faça**
3 Construção$(g(.), \alpha, s)$;
4 BuscaLocal$(f(.), N(.), s)$;
5 **se** $f(s) < f*$ **então**
6 $s* \leftarrow s$;
7 $f* \leftarrow f(s)$;
8 **fim se**
9 **fim para**
10 **devolva** $s*$;
**fim**

Na fase de construção, uma solução é iterativamente construída, elemento por elemento. A parte gulosa da função visa gerar uma solução factível. O componente

aleatório é incluído para explorar regiões diversas do espaço de soluções e é uma das chaves da efetividade do GRASP.

A fase de construção do GRASP é baseada na construção de uma lista restrita de candidatos (LCR). Essa lista contem os melhores candidatos que podem ser adicionados a solução em um dado momento. A quantidade de elementos dessa lista é regulada pelo $\alpha$ que é um dos parâmetros do GRASP. O $\alpha$ é definido como sendo o nível de aleatoriedade da solução.

**Figura 2.4: Pseudocódigo do procedimento de construção do GRASP para um problema de minimização.**
**Fonte: (FEO; RESENDE, 1995)**

**Algoritmo** *Construção*$(g(.), \alpha, s)$
1 s $\leftarrow \emptyset$;
2 Inicialize o conjunto $C$ de candidatos;
3 **enquanto** $C \neq \emptyset$ **faça**
4 $g(t_{min}) \leftarrow min\{g(t) \mid t \in C\}$;
5 $g(t_{max}) \leftarrow max\{g(t) \mid t \in C\}$;
6 $LCR \leftarrow \{t \in C \mid g(t) \leq g(t_{min}) + \alpha(g(t_{max}) - g(t_{min}))\}$;
7 Selecione aleatoriamente um elemento $t \in LCR$;
8 $s \leftarrow s \cup \{t\}$;
9 Atualize conjunto de candidatos;
10 **fim enquanto**
11 **devolva** $s$;
**fim**

Em cada iteração são selecionados todos os elementos que podem ser inseridos na solução e então é formada uma lista de candidatos que é ordenada segundo algum critério pré-determinado. No caso de um problema de minimização, a lista normalmente é ordenada de acordo com o acréscimo na função objetivo que esse elemento acarretaria se fosse escolhido. A heurística é dita adaptativa porque os benefícios associados com a escolha de cada elemento são atualizados em cada iteração da fase de construção para re-

fletir as mudanças oriundas da seleção do elemento anterior. A componente probabilística do procedimento reside no fato de que cada elemento é selecionado de forma aleatória a partir de um subconjunto restrito formado pelos melhores elementos que compõem a lista de candidatos. Este subconjunto recebe o nome de lista de candidatos restrita (LCR). Esta técnica de escolha permite que diferentes soluções sejam geradas em cada iteração GRASP (SOUZA, 2009). O valor do grau de aleatoriedade $\alpha$ se encontra entre [0,1].

Um valor de $\alpha = 0$ faz com que o algoritmo gere soluções puramente gulosas enquanto a escolha de um $\alpha = 1$ faz com que o algoritmo gere soluções puramente aleatórias.

A construção do GRASP difere do Algoritmo 2.1 por causa das linhas 4 a 7. A linha 4 obtém o valor mínimo que será acrescentado à solução final, dentre os candidatos possíveis, e a linha 5 obtém o valor máximo. A linha 6 forma a LCR com os elementos que tiverem o valor entre $g(t_{min}) + \alpha(g(t_{max}) - g(t_{min}))$. Por fim, a linha 7 seleciona aleatoriamente um elemento da LCR.

Com isso a quantidade de soluções possíveis é ampliada, porém somente soluções promissoras são geradas.

As soluções geradas pela fase de construção do GRASP normalmente não são localmente ótimas com relação à definição de vizinhança adotada. Surge então a necessidade de complementar o método com a adição de uma busca local, que tem o objetivo de melhorar a solução construída na fase de construção. O Algoritmo 2.5 descreve um procedimento básico de busca local relativo a uma vizinhança $N(.)$ de $s$ para um problema de minimização. A qualidade da construção gerada causa um impacto direto na busca local, uma vez que essa solução inicial pode se mostrar como um ponto de partida promissor para a busca local, permitindo assim agilizá-la.

**Figura 2.5: Pseudocódigo do procedimento de busca local do GRASP.**
**Fonte: (FEO; RESENDE, 1995)**

**Algoritmo** *BuscaLocal*$(f(.),N(.),s)$
1 $V \leftarrow \{s' \in N(s) \mid f(s') < f(s)\}$;
2 **enquanto** $\mid V \mid > 0$ **faça**
3 Selecione $s'$ de $V$;
4 $s \leftarrow s'$;
5 $V \leftarrow \{s' \in N(s) \mid f(s') < f(s)\}$;
6 **fim enquanto**
7 **devolva** $s$;
**fim**

O algoritmo de busca local define nos passos 1 e 5 o conjunto de vizinhos da solução $s'$ que melhoram o valor de sua função objetivo. Do passo 2 a 6, a solução corrente é atualizada enquanto houver uma solução melhor na vizinhança.

O GRASP apresenta basicamente o parâmetro $\alpha$ que pode ser ajustado. Valores de $\alpha$ que levem a uma LCR com tamanho bastante limitado implicam soluções próximas as da solução gulosa, obtidas com um baixo esforço computacional, provocamdo assim uma baixa variedade de soluções construídas, que normalmente não é interessante para a busca local já que as soluções geradas são muito próximas. Por outro lado a escolha de valores de $\alpha$ muito elevado implica na geração de uma grande diversidade de soluções mas, por outro lado, muitas das soluções construídas são de baixa qualidade.

Procedimentos GRASP mais sofisticados levam em consideração a mudança do valor de $\alpha$ ao longo das iterações de acordo com os resultados obtidos em iterações anteriores. Estudos feitos em (PRAIS; RIBEIRO, 2000) indicam que essa adaptação do valor de $\alpha$ produz soluções melhores do que aquelas obtidas com um valor fixo.

### 2.2.2 ILS

Essa seção descreve a metaheurística ILS (*Iterated Local Search* - Busca Local Iterativa) que se baseia na idéia de que um procedimento de busca local consegue melhores resultados quando inicializa a busca em diferentes bacias de atração. Esses locais diferentes são obtidos a partir de pertubações em cima da solução ótima local corrente.

O Algoritmo 2.6 ilustra o pseudo-código do ILS. Nele pode-se perceber a necessidade da definição de quatro procedimentos: (a) $GeraSoluçãoInicial()$ que obtém o ponto de partida $s_0$ para o problema; $BuscaLocal(s)$, que retorna o mínimo local da solução $s$, tendo como base as estruturas de vizinhança definidas; (c) $Pertubação(histórico, s)$, que altera a solução $s$ para outra solução, e se utiliza do histórico para evitar repetir soluções bem como para inferir o grau de pertubação necessário para escapar do mínimo local; E o (d) $CritérioDeAceitação(s, s'', histórico)$, que decide em qual solução a próxima pertubação será aplicada.

**Figura 2.6: Pseudocódigo do procedimento Iterated Local Search.**
**Fonte: (SOUZA, 2009)**

**Algoritmo** *ILS*
1 $s_0 \leftarrow$GeraSoluçãoInicial;
2 $s \leftarrow$BuscaLocal($s_0$);
3 **enquanto** os critérios de parada não estiverem satisfeito **faça**
4 $s' \leftarrow$Pertubação($histórico, s$);
5 $s'' \leftarrow$BuscaLocal($s'$);
6 $s \leftarrow$CritérioAceitação($s, s'', histórico$);
7 **fim enquanto**
8 **devolva** $s$;
**fim**

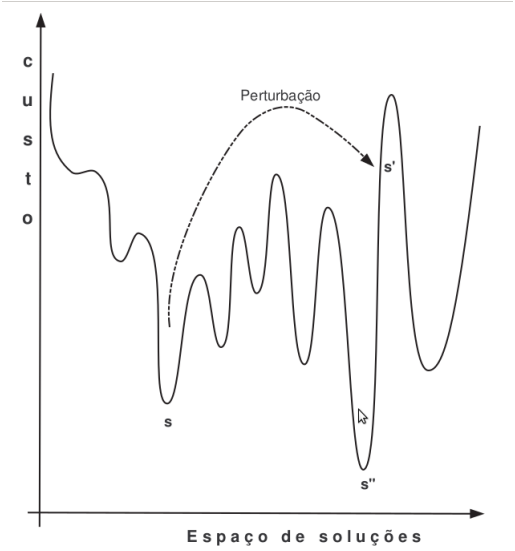
O ILS é dependente da escolha do método de busca local, das pertubações e do

critério de aceitação. Normalmente um método de descida é utilizado, mas também é possível aplicar algoritmos mais sofisticados como Busca Tabu ou outras metaheurísticas.

A intensidade da perturbação deve ser forte o suficiente para permitir escapar do ótimo local corrente e permitir explorar diferentes regiões. Ao mesmo tempo, ela precisa ser fraca o suficiente para guardar características do ótimo local corrente (SOUZA, 2009).

Um aspecto importante do critério de aceitação e da pertubação é que eles induzem aos procedimentos de intensificação e diversificação. A intensificação consiste em procurar melhores soluções nas área de busca corrente, isso acontece reduzindo a força da pertubação que faz com que as novas soluções de partida se encontrem nas proximidades da anterior. A diversificação acontece com a aplicação de grandes pertubações.

**Figura 2.7: Representação esquemática do funcionamento do ILS.**
**Fonte: (SOUZA, 2009)**




A Figura 2.7 demonstra o funcionamento do método ILS em um problema de

minimização. Dado um ótimo local $s$, é realizada uma pertubação que lhe direciona para $s'$. Depois da aplicação da busca local, o novo mínimo $s''$, melhor que a anterior, é encontrada. Ou seja $f(s'') < f(s)$.

Uma exemplo de pertubação seria a aplicação sucessiva de estruturas de vizinhança a solução corrente.

## 2.3 PROGRAMAÇÃO LINEAR

A programação linear é provavelmente a mais conhecida e utilizada técnica de otimização em todo o mundo, e geralmente é utilizada para tomada de decisões gerenciais sobre a alocação de recursos para produção. Os custos dos recursos e as receitas geradas pelos produtos são usados para determinar a melhor solução. Qualquer problema que possa ser formulado com variáveis de decisão reais, tendo uma função objetivo linear, e funções de restrição lineares, em princípio pode ser solucionado através da programação linear. Tais programas originariamente utilizavam o método *Simplex*, porém, após a década de 80, temos também métodos de "*pontos interiores*".

Embora a programação linear seja muito eficiente para a resolução de problemas lineares, sua aplicação a problemas que apresentem objetivos ou restrições não-lineares tem levado a problemas e falhas de modelagem. Em alguns casos, funções não-lineares podem ser aproximadas por algumas funções lineares conjugadas, e a programação linear ainda pode ser utilizada. Contudo, isso leva a uma representação ineficiente do problema, podendo causar matrizes de decisão explosivamente grandes que demandam um tempo excessivo para resolução. Esta é uma dificuldade comum em problemas que envolvem, por exemplo, "*scheduling*"e "*sequenciamento*"de processos como é o caso do PCTA.

De forma equivalente, outros tipos de variáveis não podem ser tratadas diretamente com o uso de programação linear. Programação inteira usa programação linear para resolver problemas sobre variáveis inteiras, mas ainda com funções objetivo e restrições puramente lineares. As variáveis inteiras são representadas como variáveis reais no algoritmo de resolução do problema. Então um processo repetitivo é usado para delimitar o valor destas variáveis em valores inteiros, através da adição de restrições e reprocessamento da solução. Esse método, conhecido como *branch & bound*, finaliza quando todas as variáveis assumem valores inteiros. Quando o número de variáveis inteiras é pequeno, a programação inteira soluciona o problema rapidamente. Infelizmente esse procedimento pode consumir muito tempo com um número grande de variáveis inteiras, podendo, em alguns casos, necessitar de milhões de iterações para serem resolvidos.

Essa técnica foi muito utilizada na segunda guerra mundial para otimizar as perdas inimigas e reduzir o custo das operações e também é utilizado no planejamento de algumas empresas.

# 3 REVISÃO DA LITERATURA

(ARGUELLO; BARD, 1997) apresenta um algoritmo, baseado no GRASP, para reconstruir trilhos de aeronaves que tenham sofrido modificações durante o decorrer do dia.

Para isso ele usa uma função de avaliação gulosa e um método de seleção aleatório que de forma iterativa tenta construir uma solução viável, a solução inicial é utilizada como entrada e então é viabilizada através desses movimentos. Note que se a solução inicial ainda for válida, a fase de construção se torna desnecessária.

Em cada iteração os movimentos possíveis são gerados e avaliados. Os mais promissores são armazenados na lista restrita de candidatos (LRC) na qual uma é selecionada aleatoriamente. A LCR pode ser restringida por quantidade ou por qualidade. Por exemplo, os melhores $X$ movimentos ou todos os movimentos com $Y\%$ do melhor movimento podem ser armazenados na LRC. Esse processo é repetido até que uma solução válida seja obtida. Isto representa uma iteração. A vantagem de usar a solução inicial é que ela é próxima a malha aérea planejada inicialmente pela companhia.

A fase de busca local usa a solução da fase de construção para encontrar um mínimo local. Essa fase usa procedimentos de trocas determinísticas para atingir o mínimo local.

A vizinhança definida inclui a criação de um novo conjunto de trilhos em que os voos que forem alocados a eles serão cancelados. Três são as operações usadas para gerar novos vizinhos: *flight route augmentation*, *partial route exchange*, e *simple circuit cancellation*. Os dois primeiros são aplicados em um par de trilhos e o terceiro é aplicado em um único trilho.

O *flight route augmentation* opera removendo um voo ou uma sequência de voos de um dos trilhos e adicionando esses voos no outro. O trilho de destino é aumentado dos voos removidos do trilho de origem. O trilho de destino pode ser aumentado de três formas distintas. Primeiro, um circuito pode ser colocado no seu início. Um circuito é uma sequência de voos que inicia e termina no mesmo aeroporto. Uma segunda forma é inserir o circuito em algum lugar entre o primeiro e o último voo do trilho de destino. A terceira forma permite alterar o aeroporto de destino, ou seja, insere uma sequência de voos no trilho de destino.

O *partial route exchange* opera simplesmente trocando um par de sequências de voos entre dois trilhos e o *simple circuit cancelation* remove um circuito de um trilho e cria um trilho de cancelamento.

Os testes realizados no artigo foram feitos em uma instância cedida pela Continental Airlines, referente a frota 757, com 42 voos operados por 16 aeronaves em uma rede de 13 aeroportos espalhados por 3 continentes. A malha não foi disponibilizada no artigo.

(MERCIER; SOUMIS, 2007) desenvolveram um modelo matemático que permite resolver de forma integrada a construção de trilhos de aeronaves e a escala de tripulantes permitindo mudanças nos horários dos voos. A ligação entre as restrições desses dois problemas garante que o mesmo planejamento seja feito para os trilhos das aeronaves e para a escala dos tripulantes, permitindo assim a redução de custos pela antecipação de problemas das etapas futuras. Por outro lado essa abordagem aumenta a complexidade na resolução desses problemas. Com a agregação desses dois problemas a resolução se tornou pesada e viável apenas para instâncias diárias. Os testes do algoritmo foram baseados em instâncias contendo no máximo 500 voos que foram fornecidas por duas

grandes companhias aereas, os dados dessas instâncias não se encontram disponíveis no artigo.

(CORDEAU *et al.*, 2001), (KLABJAN *et al.*, 2002) e (COHN; BARNHART, 2003) mostraram em seus trabalhos que a resolução desses problemas de forma integrada podem gerar soluções que são significativamente melhores que as obtidas pela resolução sequencial.

Pontes R., et al (PONTES *et al.*, 2002) utilizaram a fase de construção do GRASP para resolver o PCTA, também propuseram um modelo matemático que foi adaptado para auxiliar na geração da nossa solução. Além disso, uma instância da Rio-Sul foi disponibilizada para a realização de testes. Com o solver eles conseguiram obter a solução ótima dessa instância mas o autor informou que essa resolução demorou dias para finalizar. Com a utilização da heurística eles conseguiram apenas se aproximar dessa solução porém com um tempo de 384 segundos.

Em (HAOUARI *et al.*, 2011) Mohamed et al. resolveu de forma integrada o problema de atribuição de frota e o problema de construção de trilhos de aeronaves, para uma pequena empresa de aviação a TunisAir. Além disso as restrições de manutenção não foram levadas em consideração pelo fato dela poder ser feita em todos os aeroportos em que as aeronaves passam a noite.

# 4 DESCRIÇÃO DO PROBLEMA

Com um conjunto de voos atribuídos a uma mesma frota tem-se a necessidade de sequenciá-los de forma que cada aeronave possa ficar responsável por uma sequência válida de voos que são chamada de trilho. Um conjunto de voos está bem sequenciado se é possível atendê-los com o menor número de aeronaves e com a menor alteração do planejamento inicial dos horários desses voos.

Esse problema apresenta uma grande quantidade de restrições que devem ser atendidas. Essas restrições foram obtidas a partir da literatura e também da legislação vigente. Ainda podem existir outras restrições, porém atualmente existe uma lacuna causada pela falta de comunicação entre a academia e as empresas que, em sua grande maioria, são fechadas a esse tipo de comunicação com a finalidade de proteger seus ativos.

As principais restrições que envolvem o PCTA são as temporais e as geográficas. Essas duas restrições são triviais e representam respectivamente que um voo não pode partir antes da chegada de seu antecessor e que um voo não pode partir de um local diferente do local de chegada do voo que lhe antecede, a não ser que haja tempo para a criação de um voo de reposicionamento.

Deve-se levar em consideração também o tempo de solo que representa o tempo que o avião tem que permanecer no solo a cada pouso. Esse tempo é requerido para efetuar o reabastecimento da aeronave e a troca de passageiros que chegaram ao seu destino ou que irão fazer conexões no aeroporto que a aeronave pousou. Esse tempo é variável dependendo do aeroporto e do tipo da aeronave que está sendo operada. Adicionalmente esse tempo pode ser acrescido, se houver necessidade de efetuar a trocar da tripulação.

Existe também a possibilidade de modificar o tempo de partida dos voos planeja-

dos, no entanto essas mudanças são prejudiciais para a companhia, por alterar o horário que foi inicialmente definido pela demanda, e por isso são penalizadas se adicionadas a solução. Também é possível efetuar a ligação de voos que estão em aeroportos diferentes a partir da criação de um voo de reposicionamento, ou seja, de um voo que não estava no planejamento inicial.

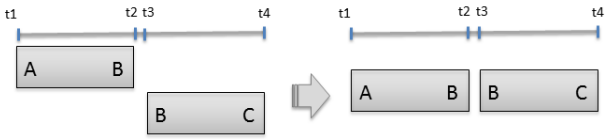
Essas restrições induzem a formação de uma rede de possíveis conexões entre voos, essas conexões são denominadas de tipos de arcos. Porém para conseguir melhores resultados deve-se permitir uma maior flexibilidade na construção dos trilhos possibilitando mudanças no tempo de partida sugerido dos voos e também a criação de voos de reposicionamento.

A ligação dos voos pode ocorrer de 6 formas distintas. Os arcos do tipo 1 permitem a ligação de voos sem a utilização de atrasos e/ou reposicionamento. Os arcos do tipo 2 utilizam atrasos mas não o reposicionamento. Os arcos do tipo 3 permitem o sequenciamento com a utilização de um voo de reposicionamento mas sem inserir atraso em nenhum dos voos envolvidos. Os arcos do tipo 4 utilizam-se de atrasos e de um voo de reposicionamento para fazer a ligação entre dois voos. Os arcos do tipo 5 e 6 são usados na modelagem matemática, que foi formulado como um problema de minimização de fluxos em rede, para representar respectivamente o nó origem(*source*) e o destino(*sink*). Abaixo um maior detalhamento desses arcos:

- Conexão natural (Arco do tipo 1) - Os arcos desse tipo conectam dois voos respeitando o tempo de partida sugerido e a restrição geográfica. Eles são associados com as ligações que não requerem mudanças no tempo de partida e nem precisam de voos de reposicionamento. O arco do tipo 1 não apresenta custo para ser

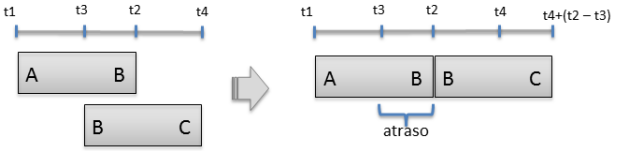
adicionado a solução.

**Figura 4.1: Representação esquemática do arco do tipo 1.**
**Fonte: (Própria)**




- Mudança no tempo (Arco do tipo 2) - Apesar de ter os voos incidentes no mesmo aeroporto, os arcos desse tipo não permitem a ligação de forma direta pois o tempo de solo disponível não é suficiente para permitir a ligação. No entanto, a escolha desse tipo de arco implica em uma mudança no tempo de partida sugerido para quaisquer um dos voos envolvidos. O custo de um arco desse tipo é igual a soma dos valores absolutos dos atrasos (em minutos) dos horários de partida envolvidos.

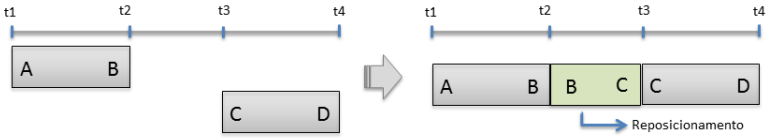
**Figura 4.2: Representação esquemática do arco do tipo 2.**
**Fonte: (Própria)**




- Voos de reposicionamento (Arco do tipo 3) - Esses arcos representam conexões entre dois voos onde o primeiro chega em um aeroporto diferente do aeroporto de partida do voo seguinte, no entanto, existe tempo suficiente para um voo de reposicionamento, entre os dois locais, sem violar as restrições de tempo de solo. O

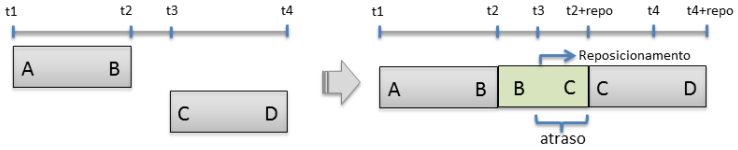
custo de um arco do tipo 3 é igual a duração do voo de reposicionamento, incluindo o seu tempo de solo.

**Figura 4.3: Representação esquemática do arco do tipo 3.**
**Fonte: (Própria)**




- Voos de reposicionamento mais mudança de tempo (Arco do tipo 4) - Esses arcos representam conexões que precisam de um voo de reposicionamento mais mudança no tempo de partida sugerido. O arco do tipo 4 tem custo igual tempo do voo de reposicionamento, incluindo o tempo de solo, mais a soma dos atrasos dos horários de partida envolvidos em valor absoluto.

**Figura 4.4: Representação esquemática do arco do tipo 4.**
**Fonte: (Própria)**

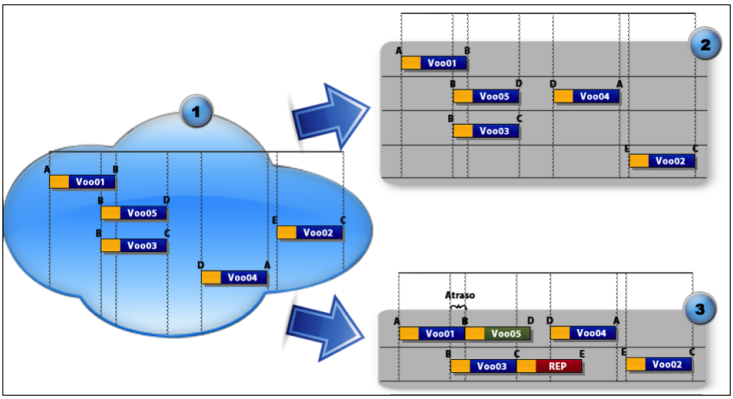



- Arcos do nó fonte ou *source* (Arco do tipo 5) - Esses, arcos são criados para identificar o inicio de um trilho e é com ele também que se sabe a quantidade de trilhos necessários para resolver o problema. Cada arco do tipo 5 tem o custo igual a 1000.

- Arcos do nó final ou *sink* (Arco do tipo 6) - Esses arcos tem como destino o nó fictício que é usado para finalizar um trilho no modelo. Os arcos do tipo 6 não tem custo.

Na Figura 4.5 temos um exemplo visual de como o processo de montagem dos trilhos das aeronaves podem influenciar de forma significativa na utilização dos recursos da companhia aérea. Cada retângulo representa um voo onde a parte laranja representa o tempo de solo que cada voo deve obedecer e a azul o tempo de voo esperado da cidade de origem para a cidade de destino. As letras representam aeroportos diferentes e a linha pontilhada indica o tempo de início e de término de cada voo, ela é importante para identificação dos atrasos.

**Figura 4.5: Construção de Trilhos de Aeronaves.**
**Fonte: (Própria)**




A parte 1 da Figura 4.5 representa os voos da companhia que ainda não foram cobertos por nenhuma aeronave e nas partes 2 e 3 são demonstradas duas formas de organizar esses voos em trilhos.

Na parte 2 temos a melhor forma possível de se organizar os voos disponíveis

utilizando apenas os arcos do tipo 1, ou seja sem a utilização de atrasos ou de voos de reposicionamento. Dessa forma se consegue uma formação com 4 trilhos.

Na parte 3 temos a melhor forma de organizar os voos com a possibilidade de utilizar qualquer tipo de ligação e com um atraso máximo igual a um tempo de solo. Pode-se verificar que atrasando um voo e criando outro de reposicionamento consegue-se economizar duas aeronaves, gerando uma solução com apenas 2 trilhos.

Pode-se verificar que a utilização de diferentes tipos de arcos pode proporcionar uma melhora significativa na qualidade da solução. Porém essa abordagem faz com que a complexidade e a quantidade das soluções cresça muito e por isso a escolha dos tipos de ligações entre os voos deve ser feita de forma cuidadosa.

# 5 MÉTODO PROPOSTO

O método proposto se utiliza do GRASP, do ILS e da abordagem exata através da programação linear inteira, pretendendo tirar proveito das vantagens de cada uma dessas técnicas. Ou seja, combinando a agilidade dos métodos heurísticos com a optimalidade do método exato.

Da mesma forma que em outras abordagens heurísticas, esse novo algoritmo consome pouco tempo computacional, em relação ao método exato e tem a capacidade de escapar de mínimos locais.

O GRASP foi utilizado como a base do algoritmo, onde a parte da construção seguiu a sua definição padrão, com a geração de uma lista restrita de candidatos (LRC) e a posterior escolha aleatória entre esses elementos. A parte da busca local foi adaptada para executar em conjunto com o ILS modificado. Para o ILS foram definidos algumas estruturas de vizinhança que foram utilizadas na busca local, e a perturbação foi feita com a utilização de um *solver* em uma parte do problema. Essa abordagem permite que o algoritmo gere boas soluções e escape de mínimos locais além de promover uma aceleração na obtenção de boas soluções, pois quando o *solver* encontra uma melhor solução ele consegue mudar o espaço de soluções em que a busca era efetuada.

O *solver* é utilizado para resolver um modelo matemático que foi desenvolvido baseado na proposta de (PONTES *et al.*, 2002) que é aplicado a uma parte do problema cada vez que se deseja fazer uma perturbação. Enquanto a busca local usa o método de descida, variando entre três estruturas de vizinhança, o *swap-x*, o *crossover* e a *compactação*. Mais adiante serão dados mais detalhes sobre o modelo matemático, a forma de escolha do sub problema, da fase de construção que foi implementada, da busca

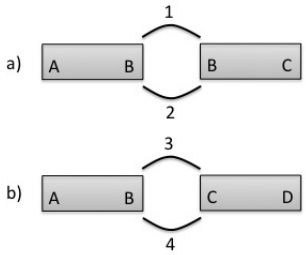
local e das implementações que não tiveram êxito.

## 5.1 MODELO MATEMÁTICO

A modelagem proposta por (PONTES *et al.*, 2002) aborda todas as restrições do problema fazendo com que a quantidade de restrições geradas seja muito elevada. A idéia utilizada na nossa formulação é a de tentar reduzir ao máximo a quantidade de restrições necessárias. Isso é feito com a modelagem de apenas algumas restrições, aquelas que são possíveis no mundo real.

Primeiro se percebeu que não há necessidade de modelar os 4 tipos de arcos para cada voo, uma vez que dados dois voos só pode vir a ocorrer dois tipos de arcos possíveis entre eles. Essa situação é ilustrada na Figura 5.1.

**Figura 5.1: Arcos necessários para ligar dois voos.**
**Fonte: (Própria)**




Em a) os voos respeitam a restrição geográfica, dessa forma apenas os arcos de tipo 1 e 2 precisam ser modelados uma vez que não teria sentido fazer um voo de reposicionamento nessa situação. Em b) os aeroportos em questão são diferentes, sendo necessário apenas a modelagem dos arcos do tipo 3 e 4, perceba que não teria outra alternativa se não fazer um voo de reposicionamento.

### 5.1.1 Definição

Seja $D = (V,A)$ um grafo representando uma instância do PCTA, onde o conjunto de vértice $V = v_i : i \in I$ de D é indexado por $I = 1,2,...,n+1,n+2$ onde $v_{n+1}$ e $v_{n+2}$, identificam, respectivamente, os nós fonte e destino. E os nós restantes referem-se ao conjunto de nós originais, com $n$ elementos. Sejam os custos $c_{ij} : (i,j) \in A$ introduzidos acima, estando associados com cada arco do grafo.

Seja $x_{ij} : (i,j) \in A$ um conjunto binário 0-1 de variáveis usada para controlar a inclusão $(x_{ij} = 1)$ ou a exclusão $(x_{ij} = 0)$ de um arco (possível conexão) entre vértices (voos) $v_i$ e $v_j$. O conjunto $\bar{I}$ identifica o conjunto de nós excluindo o nó fonte $(v_{n+1})$ e o nó de destino $(v_{n+2})$. A função objetivo foi dividida para facilitar o entendimento de como é feito o custo de adicionar um trilho. Variáveis reais $\delta_i$ e $\theta_i$, $i \in \bar{I}$ são usados para representar, respectivamente, o desvio do tempo de partida sugerido e a norma desse desvio para $v_i$. Essas variáveis devem no entanto obedecer $-\gamma_i \geq \delta_i \geq \gamma_i$ e $0 \geq \theta_i \geq \gamma_i$, onde $\gamma_i$ é o valor máximo de desvio permitido (em cada direção) do tempo de partida sugerido para o voo. Finalmente o tempo de partida sugerido que é dado por $s_i : i \in \bar{I}$.

## 5.2 FUNÇÃO OBJETIVO

$$Minimizar \quad \sum_{j \in \bar{I}} x_{v_{n+1}j}(CUSTO\_TRILHO) + \sum_{i \in \bar{I}} \sum_{j \in I} x_{ij} c_{ij} + \sum_{i \in \bar{I}} \theta_i \tag{1}$$

## 5.3 RESTRIÇÕES

a) Garantia de recobrimento dos voos

$$\sum_{i \in I} x_{ij} = 1 \ \ \forall_j \in \bar{I} \tag{2}$$

$$\sum_{j \in I} x_{ij} = 1 \ \ \forall_i \in \bar{I} \tag{3}$$

b) Viabilidade das conexões

$$s_i + t_i x_{ij} - M(1 - x_{ij}) + \delta_i \leq s_j + \delta_j \ \ \forall_{i,j} \in \bar{I} \tag{4}$$

c) Modulo do desvio do tempo de partida sugerido

$$\theta_i \geq \delta_i \ \ \forall_i \in \bar{I} \tag{5}$$

$$\theta_i \geq -\delta_i \ \ \forall_i \in \bar{I} \tag{6}$$

d) Limites das variáveis

$$0 \geq \theta_i \geq \gamma_i \ \ \forall i \in \bar{I} \tag{7}$$

$$-\gamma_i \geq \delta_i \geq \gamma_i \ \ \forall_i \in \bar{I} \tag{8}$$

$$x_{ij} \in \{0, 1\} \tag{9}$$

A função 1 do modelo representa a função objetivo onde a primeira parte representa o custo de inicializar um trilho, ou seja, sempre que um arco partindo de $n+1$, que é o nó origem, for ativado acrescenta-se o custo de um trilho a solução. A segunda parte representa o custo de adicionar os demais arcos e a terceira parte representa a adição do módulo dos atrasos que foram utilizados. Perceba que a primeira e a segunda parte podem ser modeladas juntas, a sua separação foi feita para facilitar o entendimento do modelo.

As restrições 2 e 3 representam respectivamente que todos os nós deverão ter apenas um antecessor e um sucessor, as únicas exceções são o nó origem que pode incidir em vários outros nós e o nó de destino que pode ser incidido por vários nós.

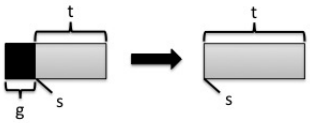
A restrição 4 é utilizada para manter a viabilidade das conexões acrescentando atrasos a elas se for necessário. Se $x_{ij} = 1$ então tem-se que o tempo de partida do voo i ($s_i$) mais a duração do voo do seu voo ($t_i$) mais um possível atraso nesse voo ($\delta_i$) deve ser menor ou igual ao tempo de partida do voo j ($sj$) mais um atraso que lhe seja dado ($\delta_j$). Se $x_{ij} = 0$ então M irá validar automáticamente essa restrição. M tem valor igual a soma da duração dos voos. As restrições 5, 6 e 7 modelam o valor absoluto do atraso em $\theta$.

A equação 8 restringe o atraso/adiantamento máximo que pode ser utilizado e a equação 9 modela a variável de decisão $x$ como sendo booleana.

Pode-se perceber que o modelo matemático não faz menção ao tempo de solo ($g$). Isso ocorre pois esse tempo é incorporado ao voo como demonstrado na Figura 5.2, ou seja o tempo de partida sugerido $s$ passa a ter o valor $s-g$ e a duração $t$ do voo passa a ter o valor $t+g$. Uma vantagem de usar essa abordagem que integra o tempo de solo ao voo é a redução da quantidade de restrições do problema.

Além disso, o conjunto $A$ contém apenas um tipo de arco, o arco do tipo 1 se os

**Figura 5.2: Conversão de um voo para ser utilizado no solver.**
**Fonte: (Própria)**




voos satisfazem a restrição geográfica e o arco do tipo 3 caso não satisfaçam. Os arcos dos tipos 2 e 4 são modelados a partir da variável $\delta$ que tem seu custo acrescentado na função objetivo.

Essa estratégia permite a redução de 3 arcos para cada voo, o que deixa o modelo mais leve.

O cálculo dos custos são feitos através de um pré-processamento, onde os arcos viáveis recebem os valores referentes ao seu tipo, por exemplo, no caso de um arco originário do nó origem, arco do tipo 6, um custo 1000 é atribuído. No caso de arcos que deverão ser evitados um custo elevado é atribuído.

## 5.4 FASE DE CONSTRUÇÃO DO GRASP

A construção da solução é feita elemento a elemento utilizando o GRASP. Inicialmente é feita a ordenação do conjunto de voos a partir do seu tempo de partida sugerido. O algoritmo só termina quando todos os voos já foram alocados em algum trilho.

Existem duas formas de fazer a montagem da solução, uma seria a montagem de trilhos de forma sequencial, onde um novo trilho só é iniciado quando o anterior se encontra saturado. A outra forma é a montagem de trilhos de forma paralela, que, a priori, provocaria uma melhor distribuição dos voos. Na prática a primeira abordagem foi adotada, pois, nas instâncias disponíveis ela apresentou, sempre, soluções de melhor

qualidade.

Pode ser que para instâncias com alguma característica específica a estratégia de montagem dos trilhos de forma paralela podem apresentar melhores resultados.

### 5.4.1 Formação dos trilhos de forma sequencial

Quando se pensa na escolha do primeiro voo do trilho, a decisão imediata é a escolha do voo que contenha o menor horário de partida sugerido, ou seja, o voo mais próximo. Porém essa escolha reduz a quantidade de soluções que podem ser geradas, isso ocorre pois o primeiro voo tem uma grande influência nas possíveis soluções que um trilho pode assumir. Para evitar isso a escolha do primeiro voo de um trilho é feita baseando-se nos 5 voos com menor horário de partida que ainda não estejam alocados a nenhum outro trilho.

Esses voos são adicionados a lista de candidatos iniciais (LCI) em seguida é feita a escolha do elemento que irá iniciar o novo trilho levando em consideração apenas os elementos que possuam o horário de partida distante de até $\alpha\%$ do voo de menor horário de partida, esses elementos são então inseridos em uma lista restrita inicial (LCI) na qual é feita a seleção aleatório de um dos voos. Isso é feito para evitar a escolha de um voo muito distante do menor voo.

Os pseudocódigos 5.3 e 5.4 apresentam esses procedimentos. Primeiro deve-se entender o pseudocódigo 5.3 onde é feita a escolha de um voo que deverá iniciar um trilho, esse pseudocódigo serve tando para montar trilhos de forma sequencial como de forma paralela. Em 1 é formada a LCI que é composta pelos 5 primeiros voos ainda não alocados, de 2 a 4 é feita a formação da LRI tendo como base o menor horário de partida

$h(v_{min})$, o maior horário de partida $h(v_{max})$ e o $\alpha$ e em 5 é feita a escolha aleatória de um desses voos. Já no pseudocódigo 5.4 é feita a montagem dos trilhos de forma sequencial. Em 1 os voos disponíveis são ordenados de acordo com seu horário de partida, em 2 a malha, que será o resultado, é inicializada com vazio. De 3 a 9 é feita a montagem dos trilhos que irão compor a malha, em 4 e 5 é obtido o primeiro voo do trilho, em 6 é feita a montagem completa desse trilho, em 7 os voos pertencentes ao trilho corrente são removidos do conjunto de voos disponíveis e em 8 o trilho corrente é adicionado a malha resultante.

**Figura 5.3: Pseudocódigo do procedimento de seleção de um voo inicial.**
**Fonte: Própria**

**Algoritmo** *selecionaVooInicial(V)*
1 $LCI \leftarrow$cincoPrimeirosVoos($V$);
2 $h(v_{min}) \leftarrow min\{h(v) \mid v \in LCI\}$;
3 $h(v_{max}) \leftarrow max\{h(v) \mid v \in LCI\}$;
4 $LRI \leftarrow \{v \in LCI \mid h(v) \leq h(v_{min}) + \alpha(h(v_{max}) - h(v_{min}))\}$;
5 Selecione aleatoriamente um elemento $v \in LRI$;
6 **devolva** $v$;
**fim**

### 5.4.2 Formação dos trilhos de forma paralela

Nessa estratégia um trilho é iniciado sempre que existe um voo que não pode ser inserido em nenhum dos trilhos que estejam sendo montados, mantendo-se assim um Conjunto de Trilhos Disponíveis (CTD).

Em cada iteração o Trilho Corrente (TC) é escolhido a partir do CTD de forma aleatória. Feito isso, adiciona-se um voo a esse trilho. Caso não existam voos candidatos para adição no TC este é removido da CTD e uma nova iteração é iniciada.

**Figura 5.4: Pseudocódigo do procedimento de formação sequencial dos trilhos.**
**Fonte: Própria**

**Algoritmo** *formaçãoSequencialDosTrilhos(V)*
1 Ordene o conjunto de voos não alocados $V$;
2 $M \leftarrow \emptyset$;
3 **enquanto** $V \neq \emptyset$ **faça**
4 $v \leftarrow selecionaVooInicial(V)$
5 $T \leftarrow \{v\}$;
6 $T \leftarrow completaTrilho(T)$;
7 $V \leftarrow V - \{v \in T\}$;
8 $M \leftarrow M \cup T$;
9 **fim enquanto**
10 **devolva** $M$;
**fim**

O pseudocódigo 5.5 ajudam no entendimento do algoritmo. Em 1 os voos disponíveis são ordenados de acordo com seu horário de partida, em 2 e 3 respectivamente a malha resultante e o CDT são inicializados com vazio. De 4 a 19 é feita a montagem paralela dos trilhos que irão formar a malha. Em 5 é feita a seleção de um voo inicial. Em 6 verifica-se se existe algum trilho corrente que pode incorporar esse voo, se existir então em 7 é feita a seleção de um trilho corrente (TC) a partir dos elementos da CDT e em 8 seleciona-se um voo que possa ser inserido em TC e insere o voo em 13. Se não existir nenhum voo que possa ser inserido em TC então este é removido da CDT como pode ser visto de 9 a 12.

De 15 a 18 é criado um novo trilho sempre que não existe nenhum trilho da CDT que possa comportar o voo selecionado em 5. Em 20 se insere os trilhos pertencentes a CDT na malha resultante. Isso ocorre porque todos os voos já foram alocados em algum trilho.

**Figura 5.5: Pseudocódigo do procedimento de formação paralela dos trilhos.**
**Fonte: Própria**

**Algoritmo** *formaçãoParalelaDosTrilhos(V)*
1 Ordene o conjunto de voos não alocados $V$;
2 $M \leftarrow \emptyset$;
3 $CTD \leftarrow \emptyset$;
4 **enquanto** $V \neq \emptyset$ **faça**
5 $v \leftarrow selecionaVooInicial(V)$
6 **se** $v$ pode ser inserido em um trilho do CTD **então**
7 $TC \leftarrow escolheTrilhoAleatório(CTD)$;
8 $nv \leftarrow selecionaVooCandidato(TC)$;
9 **se** $nv = \emptyset$ **então**
10 $CDT \leftarrow CDT - \{TC\}$
11 $M \leftarrow M \cup \{TC\}$
12 **senão**
13 $TC \leftarrow TC \cup \{nv\}$
14 **fim se**
15 **senão**
16 $T \leftarrow \{v\}$;
17 $CTD \leftarrow CTD \cup T$
18 **fim se**
19 **fim enquanto**
20 $M = M \cup \{t \in CTD\}$;
21 **devolva** $M$;
**fim**

### 5.4.3 Escolha dos voos de um trilho

A escolha do primeiro voo de um trilho é feita como explicado nas seções anteriores enquanto os demais voos são escolhidos tendo como base um tipo de arco e uma lista restrita de candidatos (LRC).

Os tipos de arcos foram definidos no Capítulo 4, porém nessa etapa apenas 4 tipos são considerados, o $a_1, a_2, a_3, a_4$ que representam formas de ligações entre os voos. Os arcos do tipo 5 e 6 só são utilizados na modelagem matemática. Os arcos do tipo 1

permitem a ligação de voos sem a utilização de atrasos e/ou reposicionamentos. Os arcos do tipo 2 utilizam atrasos mas não o reposicionamento. Os arcos do tipo 3 permitem o sequenciamento com a utilização de um voo de reposicionamento mas sem inserir atraso em nenhum dos voos envolvidos. Os arcos do tipo 4 utilizam-se de atrasos e de um voo de reposicionamento para fazer a ligação entre dois voos. Os arcos do tipo 5 partem do nó *source* e servem para modelar o início de um trilho. Os arcos do tipo 6 representam o nó de destino (*sink*), nele incidem todos os últimos voos de cada trilho.

Primeiramente é feita a escolha do tipo de arco que será utilizado para efetuar a ligação do último voo do trilho corrente. Essa escolha é feita tendo como base as probabilidades de cada um desses arcos acontecer. Essa probabilidade foi definida como sendo $P(a_1) = 0.79, P(a_2) = 0.16, P(a_3) = 0.04, P(a_4) = 0.01$ pois a solução ótima do problema real da Rio Sul apresentava essas características. Esses valores são empíricos e se alterados podem vir a melhorar, ou piorar, a qualidade da solução, dependendo das características da instância.

De posse do tipo de arco, é feita então a formação da lista de candidatos. Essa lista é ordenada de acordo com o seu horário de partida sugerido, caso o arco seja do tipo $A_1$, ou pelo custo associado a sua escolha para os demais tipos de arco. No caso da lista de candidatos não possuir nenhum voo, então outro tipo de arco é sorteado, até que não seja possível acrescentar voos ao trilho de nenhuma forma, quando isso ocorrer a construção desse trilho é finalizada.

Caso seja possível a obtenção de uma lista de candidatos então ela é reduzida tendo como base o passo 2 a 4 do algoritmo alg:selectinit formando assim a Lista de Candidatos Restrita (LCR), essa redução remove os candidatos que estão muito afastados do melhor candidato da lista. Como está lista se encontra ordenada, então, o elemento

de menor impacto ($v_{menor}$) na solução é o primeiro e o de maior impacto ($v_{maior}$) é o último. A LCR contém os candidatos que tenha o valor de impacto na solução entre $v_{menor}$ e $v_{menor} + \alpha * (v_{maior} + v_{menor})$, onde $\alpha$ é o grau de gulosidade do GRASP, que nesse trabalho foi definido como sendo 0.5. O candidato deve ser escolhido de forma aleatória entre os elementos da LCR.

O pseudocódigo 5.6 explica como isso acontece. Em 1 inicializa o conjunto de arcos com os 4 tipos possíveis em 2 realiza-se um sorteio do tipo de arco que irá ser inicialmente avaliado levando em consideração a probabilidade de cada um deles acontecer ($P(a_1), P(a_2), P(a_3), P(a_4)$). Em 3 é feita a remoção do tipo de arco selecionado do conjunto inicial e em 4 é selecionado o voo corrente que necessita de um sucessor, nesse caso o último voo do trilho.

De 5 a 13 procura-se um candidato para o voo corrente levando em consideração todos os tipos de arcos, iniciando pelo que foi selecionado no passo 2.

## 5.5 FASE DE BUSCA LOCAL

Com a finalização da etapa anterior tem-se uma solução do problema. A fase de busca local efetua modificações nessa solução com a finalidade de obter outras melhores que estejam próximos a ela, isso é feito através da aplicação das estruturas de vizinhanças. No método proposto essa fase foi implementada através da utilização da metaheurística ILS que alterna busca local com pertubações conseguindo assim escapar de mínimos locais, ou seja, primeiro são aplicados as estruturas de vizinhança, visando obter o valor ótimo local da solução, depois é feita uma perturbação que diversifica a solução. Isso é feito através da aplicação do modelo matemático, definido no início

**Figura 5.6: Pseudocódigo de calculo do proximo voo de um trilho**
**Fonte: Própria**

**Algoritmo** *obtemProximoVoo(T,V)*
1 $A \leftarrow \{1,2,3,4\}$;
2 $a \leftarrow sorteaTipoDeArco(A, P(a_1), P(a_2), P(a_3), P(a_4))$;
3 $A \leftarrow A - \{a\}$;
4 $v \leftarrow ultimoVoo(T)$;
5 **para** $i$ **de** 1 **para** 4 **faça**
6 $c \leftarrow proximoCandidato(v,V,a)$;
7 **se** $c = \emptyset$ **então**
8 $a \leftarrow proximoArco(A)$;
9 $A \leftarrow A - \{a\}$;
10 **senão**
11 **devolva** $c$;
12 **fim se**
13 **fim para**
14 **devolva** $\emptyset$;
**fim**

desse capítulo, em uma parte do problema. Quando nenhuma das duas estratégias consegue melhorar a solução então a busca local se encerra e uma nova iteração do GRASP pode ser iniciada.

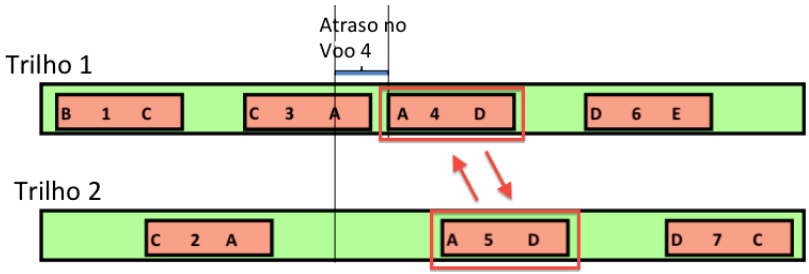
### 5.5.1 Estruturas de vizinhança

Foram definidas três estruturas de vizinhança para serem utilizadas na busca local, o Swap-X e o Cross-Over, que tem o objetivo de remover modificações nos horários de partida sugeridos dos voos, e a Compactação, que promove a redução do número de trilhos. Abaixo essa estruturas são explicadas.

Swap-X

Esse operador efetua a troca de X voos de um trilho por um conjunto de voos de outro trilho. Dessa forma pode-se conseguir remover os atrasos que foram criados na etapa de construção. No método proposto apenas os movimentos do tipo Swap-1 e Swap-2 são utilizados, pois foram as que obtiveram mais êxito em melhorar a solução. A Figura 5.7 mostra um caso em que a aplicação dessa estrutura permite remover um atraso.

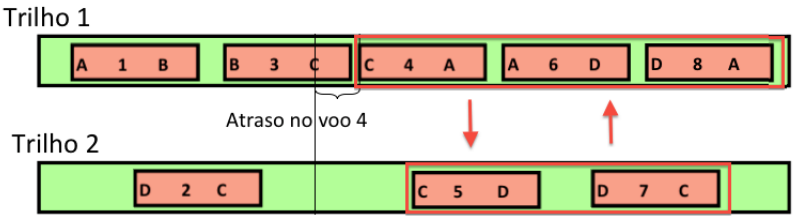
**Figura 5.7: Estrutura de vizinhança Swap-1.**
**Fonte: (Própria)**




Cross-Over

A ideia do operador *crossover* é a de efetuar troca entre dois segmentos de trilhos com a finalidade de gerar novos trilhos que não apresentem voos atrasados. A Figura 5.8 mostra um caso em que a aplicação dessa estrutura melhora a solução.

**Figura 5.8: Estrutura de vizinhança CrossOver.**
**Fonte: (Própria)**

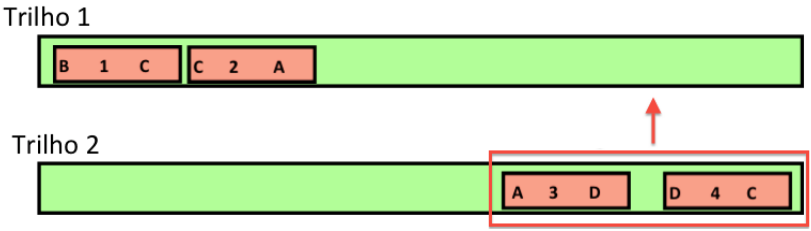



Compactação

A compactação é a única estrutura de vizinhança utilizada que é capaz de reduzir a quantidade de trilhos da solução final. Isso ocorre porque ela consegue inserir um trilho em outro de forma direta ou com a utilização de um voo de reposicionamento.

A figura 5.9 mostra a redução de um trilho com a utilização desse movimento.

**Figura 5.9: Estrutura de vizinhança Compactação.**
**Fonte: (Própria)**




### 5.5.2 Perturbação usando o método exato

A perturbação é utilizada quando as estruturas de vizinhança não conseguem melhorar a solução, quando isso ocorre, pode-se dizer que a solução corrente é a ótima local

com relação a vizinhança que foi definida.

Para tentar encontrar outros mínimos locais aplica-se uma modificação na estrutura da solução, mesmo que isso provoque uma piora na sua qualidade. Isso se mostra interessante para o algoritmo pois ele irá efetuar busca de melhorias em outros locais no espaço de soluções através da sua busca local.

O método de perturbação utilizado aqui difere do que normalmente é aplicado pois a solução tem a sua estrutura modificada e ainda consegue melhorar a sua qualidade. Isso é feito através da resolução do modelo matemático em uma parte do problema. A sua utilização ocorre com a seleção de um conjunto de trilhos, e sua posterior aplicação no *solver* configurado para o conjunto de voos da seleção.

O método exato retorna a configuração ótima desses voos, que são agrupados novamente a solução antiga. O solver tem um tempo máximo estabelecido e se não retornar nenhuma solução nesse tempo considera-se que não houve melhora e uma nova iteração é iniciada.

A seleção dos trilhos é feita com base no seu *grau de compactação* que é definido como sendo porcentagem de utilização efetiva de um trilho com relação ao tempo de partida do primeiro voo e o tempo de chegada do último voo da instância, ou seja, quanto maior o tempo que a aeronave, que opera um determinado trilho, permanece voando maior será o seu grau de compactação. O cálculo do grau de compactação não leva em consideração os voos de reposicionamento, pois eles não estão no planejamento inicial e por isso não são passados para o modelo.

Existe três formas de fazer a seleção desses. Pode-se adicionar os trilhos que possuem o maior grau de compactação, ou adicionar os trilhos que possuem o menor grau de compactação ou pode-se alternar entre a escolha de um trilho com o maior grau

de compactação e um de menor grau de compactação.

Na prática adotou-se a segunda abordagem, selecionando os trilhos de menor grau de compactação, pois os resultados foram superiores aos das outras abordagens.

Os trilhos são adicionados a solução até o limite de 80 voos, pois o solver conseguiu em nossos experimentos resolver um problema desse porte de forma imediata. Isso ocorre porque em nossos experimentos o *solver* conseguiu resolver um problema desse porte de forma quase imediata.

### 5.5.3 Método de construção paralela

A estratégia paralela foi desenvolvida para aumentar a escalabilidade na resolução de grandes problemas e também na tentativa de obter melhores soluções em um menor espaço de tempo. O seu funcionamento parte da divisão da instância em períodos de tempos iguais que são distribuídas para cada processo. Em cada período os processos realizam uma construção do algoritmo descrito na Seção 5.4. Ao final dessa etapa cada processo possui um um conjunto de trilhos que devem ser ligados uns com os outros para a geração de uma malha de voo completa.

O algoritmo é dividido em passos onde em cada passo um nível diferente de vizinho é visitado na tentativa de efetuar a ligação dos seus trilhos. No nível 1 é feita a comunicação seguindo a topologia de uma rede em anel, onde cada processo fala com seu vizinho sucessor a uma distância de 1, ou seja, o processo 0 fala com o processo 1, o 1 com o 2 e assim por diante até que o último processo fale com o processo 0. Quando todos as tentativas de ligação de trilhos entre esses processo são esgotados então o passo do algoritmo é incrementado para que o processo consiga conversar com o processo su-

cessor que possua uma distância de 2 dele, ou seja, o processo 0 se comunica com o processo 2, o 1 com o 3 e assim por diante, isso ocorre até que todos os processo tenham se comunicado com os outros ou até que todos os trilhos estejam ligados.

Essa estratégia é interessante pois normalmente as empresas trabalham com uma malha circular que costuma se repetir toda a semana.

O pseudocódigo 5.10 simplifica o entendimento do algoritmo. Perceba que a complexidade de execução e de mensagens do algoritmo distribuído 5.10 no pior caso é de $O(s.c)$, onde $s$ é o número de processos e $c$ é a maior quantidade de trilhos de um processo.

A linha 1 é utilizada na inicialização dos voos, onde cada processo obtém o conjunto de voos que irá trabalhar, esse conjunto de voos é definido pela instância $M$, pelo número de processos que estão executando $s$ e pelo número do processo corrente $r$. O laço iniciado na linha 2 modela a quantidade de iterações do algoritmo, que é baseado no número de processos. A variável *cn* é o contador de nível e informa para qual vizinho o processo corrente irá tentar efetuar ligações dos seus trilhos disponíveis.

O cálculo do índice do processo que irá receber a comunicação é feita na linha 3. A linha 4 inicia o contador de mensagens enviadas, que é usado como auxiliar de condição de parada no laço da linha 13.

É feita então uma tentativa de conectar os trilhos que ainda não foram conectados com os trilhos dos vizinhos. Nas linhas 6 a 10 é feito o envio de solicitações de conexões de todos os últimos voos ainda não conectados de cada um dos seus trilhos para o vizinho corrente que é definido pela variável *dest*. Na linha 12 o processo sinaliza para o seu vizinho corrente que não tem mais nenhum trilho para tentar efetuar uma ligação.

Depois o processo passa a agir como uma máquina de estados, como pode ser

**Figura 5.10: Pseudocódigo do método de comunicação paralela usado na construção dos trilhos.**
**Fonte: Própria**

```
Algoritmo inicioComunicacaoParalela(M, s, r)
 1  inicializaVoos(M,s,r);
 2  para cn de 1 para (s − 1) faça
 3      dest ← (r + cn) MOD s;
 4      c ← 0;
 5      para cada t em M faça
 6          f ← ultimoVoo(t);
 7          se f ainda não foi selecionado então
 8              c ← c + 1;
 9              solicitaConexão(f, dest);
10          fim se
11      fim para
12      EnviaFinalizaMsg(dest);
13      enquanto Não tiver respondido todas as requisições faça
14          P ← RecebeMSG();
15          se P é uma solicitação de conexão então
16              f ←selecionaUmVoo(M, P.voo);
17              se f ≠ ∅ então
18                  EfetuaConexão(P, f);
19              senão
20                  RejeitaConexão(P, f);
21              fim se
22          senão se P é uma aceitação de conexão então
23              ConfiguraLigação(P);
24          senão se P é uma finalização de mensagens então
25              F ← verdadeiro;
26          fim se
27      fim enquanto
28  fim para
fim
```

visto nas linhas 13 a 27, respondendo as requisições que forem recebendo. Isso acontece até ele receber uma mensagem de finalização do seu vizinho e todas as mensagens que ele tenha enviado anteriormente tenham sido respondidas. Na linha 14 o processo recebe

uma mensagem que vai lhe informar qual a ação que deverá ser executada.

Quando a mensagem é uma solicitação de conexão, linhas 15 a 22, então o processo tenta selecionar um voo, do inicio de um de seus trilhos, que ainda não tenha se ligado com nenhum outro trilho para se ligar com o voo candidato da solicitação. A escolha do voo que vai se ligar é feita usando o GRASP com uma LRC. Caso exista algum voo que possa se conectar, linha 17, então uma mensagem de aceitação de conexão é enviada para o solicitante e o voo é marcado como já selecionado, isso é feito no procedimento *EfetuaConexão(P, f)*. Caso não exista nenhum voo que possa se candidatar então uma mensagem de rejeição de conexão é enviada pelo procedimento *RejeitaConexão(P, f)*.

As linhas 22 a 24 são executadas quando o processo recebe uma mensagem de aceitação ou rejeição de conexão e é utilizada para configurar o voo que teve a solicitação aceita. Já as linhas 24 a 26 marca a flag de finalização do vizinho $F$ como verdadeiro.

Na prática, essa estratégia conseguiu obter resultados aproximadamente 10% melhores que a estratégia sequencial, ou seja, de um processo com a fatia completa do tempo, levando em consideração apenas a fase de construção. Por questões de tempo esse experimento não foi concluído, com a inserção da fase de refinamento que utiliza o *solver*.

Para justificar o uso de uma estratégia mais eficiente, como essa, seria necessário obter primeiro uma instância que não obtivesse resultados satisfatórios com a estratégia híbrida sequencial. Aparentemente essa nova estratégia iria ocasionar melhoras nos resultados finais já que a qualidade da solução gerada na construção tem influencia direta na velocidade que o algoritmo encontra uma melhor solução.

# 6 RESULTADOS

Neste capítulo apresenta-se os resultados que foram encontrados através da aplicação do solver, da metaheurística pura e da metaheurística híbrida em um conjunto de quatro instâncias obtidas ou adaptadas de duas grandes companhias aéreas, a Rio-Sul e a TAM. Os resultados obtidos com a programação linear inteira, foram utilizados como limite inferior - *lower bound* das soluções. Serão mostrados os resultados da metaheurística pura para mostrar as vantagens que foram obtidas com a utilização de uma abordagem híbrida.

Na seção 6.1 são apresentados o ambiente de teste e as características dos problemas de testes que foram utilizados nos experimentos. Nas seções 6.2, 6.3, 6.4 serão mostrados, respectivamente, os resultados obtidos com o otimizador, metaheurística pura e metaheurística híbrida. Na seção 6.5 são dadas as considerações finais.

## 6.1 AMBIENTE DE TESTES

Todos os algoritmos descritos foram desenvolvidas na linguagem C++ usando o solver CPLEX Academic 12 da IBM que implementa técnicas de resolução de programação inteira, os experimentos computacionais foram feitos em um notebook com processador Pentium T4500 2.3 Ghz, 2 GBytes de memória RAM (2x1 GB) e com o sistema operacional operacional Linux Ubuntu 11.04 de 32 bits.

As instâncias obtidas tem a duração de um dia, a da Rio-Sul tem 107 voos e a da TAM tem 241 voos. A instância da Rio-Sul foi obtida a partir do trabalho de (PONTES *et al.*, 2002) e a da TAM foi obtida através da seleção manual dos voos pelo *site*

da companhia. Com o desenvolvimento do trabalho, essas instâncias passaram a ser resolvidas facilmente pelo algoritmo. Com a finalidade de gerar instâncias mais difíceis foi proposta a extensão da frequência dos voos da instância Rio-Sul e da TAM para uma semana, dessa forma foi gerada uma instância com 749 voos e outra de 1687. A instância diária da Rio-Sul era operada por 20 aeronaves, com uma malha construída manualmente por um funcionário com larga experiência nesse tipo de serviço.

Para manter a compatibilidade dos resultados com o trabalho de (PONTES *et al.*, 2002) foi utilizado o tempo de solo de 20 minutos para todos os aeroportos. Vale lembrar que esse tempo pode variar de acordo com o aeroporto e o tipo da frota. Houve dificudades na obtenção de dados mais detalhados junto às companhias aéreas.

O GRASP foi configurado com 1000 iterações na sua aplicação pura e com 10 na sua aplicação híbrida, também foi utilizado um $\alpha$ de 0.4. Esses valores foram obtidos de forma empírica a partir dos testes que foram feitos no decorrer do trabalho. A busca local é exaustiva e finaliza quando não consegue melhorar o valor objetivo da solução. O tempo máximo de execução dos algoritmos foram de 32 horas (115200 segundos).

No método híbrido a maior utilização do poder de processamento do CPU é gasto executando a etapa de busca local, cerca de 98% do tempo.

A demonstração da eficiência do algoritimo foi feita baseando-se na solução ótima dos problemas que foi obtida a partir da utilização do modelo matemático descrito no Capítulo 5. A única instância em que isso não foi possível foi na TAM estendida, onde, no período máximo estabelecido, não foi possível obter nenhuma solução inteira para o problema. Essa foi a maior instância utilizada no trabalho.

Nesse capítulo $s*$ indica o valor ótimo de uma solução, $s_m$ indica a média dos valores obtidos com todas as execuções do algoritmo, $t_{s*}$ é o tempo de execução do solver

**Tabela 6.1: Parametrização dos cenários**

| | Cenário 1 | Cenário 2 |
|---|---|---|
| Atraso Maximo | 0 | 10 |
| Prob. Arc. Tipo 1 | 0.92 | 0.69 |
| Prob. Arc. Tipo 2 | 0 | 0.16 |
| Prob. Arc. Tipo 3 | 0.08 | 0.04 |
| Prob. Arc. Tipo 4 | 0 | 0.01 |

e $t_s$ representa o tempo médio de execução do algoritmo. Por fim $\Delta$ (GAP) representa a diferencia da média dos valores das soluções obtidas em relação ao valor ótimo da instância. O seu cálculo é feito com a fórmula abaixo:

$$\Delta(GAP) = (s - s*)/s*$$

A resolução dessas instâncias foram parametrizadas levando-se em consideração dois cenários. O cenário 1 faz o sequenciamento dos voos sem a permissão de utilizar nenhum atraso, essa representação é comum nas companhias que não aceitam a modificação do planejamento inicial. O cenário 2 se utiliza de atrasos, de no máximo 10 minutos, permitindo assim uma maior liberdade na hora da montagem dos trilhos e obtendo, assim, um melhor aproveitamento da utilização das aeronaves. Os parâmetros utilizados são detalhados na Tabela 6.1.

Os resultados que utilizaram a metaheurística foram obtidos a partir de 100 execuções e somente os valores das médias foram levados em consideração.

**Tabela 6.2: Resultados do otimizador no cenário 1**

| Instância | $s*$ (trilhos) | $t_{s*}(s)$ |
|---|---|---|
| Rio Sul | 17.138 (17) | 4 s |
| TAM | 35.334 (34) | 26 s |
| Rio Sul Estendida | 18.392 (17) | 24192 s |
| TAM Estendida | - | 115200 s |

**Tabela 6.3: Resultados do otimizador no cenário 2**

| Instância | $s*$ (trilhos) | $t_{s*}(s)$ |
|---|---|---|
| Rio Sul | 16.158 (16) | 4 s |
| TAM | 35.015 (34) | 27 s |
| Rio Sul Est. | 17.433 (16) | 33001 s |
| TAM Estendida | - | 115200 s |

## 6.2 RESULTADOS DO OTIMIZADOR

Após muitas melhorias na modelagem matemática que foi utilizada no *solver*, consegiu-se resultados bastante expressivos na resolução das instâncias. Percebe-se que em um curto espaço de tempo o *solver* conseguiu resolver de forma eficiente problemas diários de até 241 voos (TAM). Porém percebeu-se também que esse desempenho diminui bastante com o aumento do tamanho das instâncias, demorando 6 a 9 horas para resolver uma instância de 749 voos (Rio-Sul estendida) e não conseguindo resolver em 32 horas uma instância com 1687 voos (TAM estendida) nos cenários que foram modelados.

Percebeu-se também que existe um sequenciamento possível nos voos que permite economizar até 4 aeronaves para atender a instância da Rio-Sul. Na prática essa instância era operada com 20 aeronaves.

## 6.3 RESULTADOS DA METAHEURÍSTICA PURA

**Tabela 6.4: Resultados da metaheurística pura no cenário 1**

| Instância | $s_m$ (trilhos) | $t_s(s)$ | $\Delta$ |
|---|---|---|---|
| Rio Sul | 19.049 (19) | 247 s | 0.11 |
| TAM | 42.328 (42) | 1710 s | 0.20 |
| Rio Sul Estendida | 21.516 (20) | 1687 s | 0.17 |
| TAM Estendida | 51.903 (44) | 11722 s | - |

**Tabela 6.5: Resultados da metaheurística pura no cenário 2**

| Instância | $s_m$ (trilhos) | $t_s(s)$ | $\Delta$ |
|---|---|---|---|
| Rio Sul | 18.256 (18) | 224 s | 0.13 |
| TAM | 41.719 (41) | 2066 s | 0.19 |
| Rio Sul Est. | 20.080 (19) | 1592 s | 0.15 |
| TAM Estendida | 51.032 (44) | 11902 s | - |

O algoritmo usando a metaheurística pura teve dificuldades para resolver as instâncias, inclusive não obteve o valor ótimo de nenhuma delas ficando entre 11% e 20% da solução ótima obtida pelo *solver*. Apesar dos resultados não terem sido o esperado, nota-se que ele ainda é melhor que o que era utilizado em produção pela Rio-Sul. O GRASP foi configurado para fazer 1000 repetições, pois a quantidade de soluções geradas nesse método é muito diversificado sendo necessário mais tempo para obter soluções melhores.

Esse comportamento nos levou a acrescentar uma etapa de programação linear inteira na parte de busca local para obter melhores resultados.

## 6.4 RESULTADOS DA METAHEURÍSTICA HÍBRIDA

**Tabela 6.6: Resultados da metaheurística híbrida no cenário 1**

| Instância | $s_m$ (trilhos) | $t_s(s)$ | $\Delta$ |
|---|---|---|---|
| Rio Sul | 17.138 (17) | 7 s | 0 |
| TAM | 35.334 (34) | 36 s | 0 |
| Rio Sul Estendida | 18.392 (17) | 64 s | 0 |
| TAM Estendida | 49.857 (35) | 154 s | - |

**Tabela 6.7: Resultados da metaheurística híbrida no cenário 2**

| Instância | $s_m$ (trilhos) | $t_s(s)$ | $\Delta$ |
|---|---|---|---|
| Rio Sul | 16.158 (16) | 7 s | 0 |
| TAM | 35.015 (34) | 36 s | 0 |
| Rio Sul Est. | 17.532 (16) | 65 s | $<$0.01 |
| TAM Estendida | 48.803 (35) | 159 s | - |

O algoritmo híbrido conseguiu ser consistente na obtenção de bons resultados. Foram utilizadas 10 iterações do algoritmo, sendo possível obter uma convergência de 100% nos resultados mostrados nas tabelas 6.6 e 6.7.

## 6.5 CONSIDERAÇÕES FINAIS

Para instâncias pequenas a utilização do *solver* se mostrou suficiente, porém percebeu-se que com o aumento do tamanho da instância o solver leva muito tempo para resolver e uma estratégia híbrida pode ser o caminho para obter boas soluções em um curto espaço de tempo.

Pode-se observar que nos dois cenários a solução ótima foi obtida para as

instâncias da Rio-Sul e no cenário 1 para versão diária da TAM. A solução obtida na TAM Estendida pode ser considerada boa, pois levando em consideração o resultado ótimo da sua versão diária o resultado ótimo não poderia ter menos que 34 trilhos.

Alguns ajustes ainda podem melhorar o modelo híbrido para que ele possa se aproximar mais da solução ótima. A modificação da estrutura a ser otimizada na busca local pode ser um ponto que ajude a melhorar os resultados, pois a literatura mostra que esse é um dos pontos mais importantes de uma heurística híbrida.

Uma das grandes dificuldades encontradas no trabalho foi a falta de instâncias tornando difícil a comparação de resultados com outras algoritmos. Esse trabalho disponibiliza as instâncias trabalhadas nos anexos. A instância da Rio-Sul se encontra nos Anexos A e B com o melhor resultado obtido detalhado no Anexo C. A instância da TAM se encontra nos Anexos D e E com o melhor resultado obtido detalhado no Anexo F.

Experimentos realizados com a implementação paralela discutida no Capítulo 5 demonstrou potencial de melhorar os resultados obtidos pois as soluções iniciais geradas por ela se mostraram melhores que as obtidas de forma sequencial.

# REFERÊNCIAS BIBLIOGRÁFICAS


ABDELGHANY, A.; ABDELGHANY, K. **Modeling Applications in the Airline Industry**. USA: Ashgate Publishing Limited, 2009.

ARAUJO, T. M. U. de. **Métodos híbridos baseados em continuous-GRASP aplicados à otimização global contínua**. Dissertação (Mestrado) — Universidade Federal da Paraíba, 2009.

ARGUELLO, M. F.; BARD, J. F. A grasp for aircraft routing in response to groundings and delays. **Journal of Combinatorial Optimization**, v. 5, p. 211–228, 1997.

BEASLEY, J. E. Or-library: distributing test problems by electronic mail. **Journal of the Operational Research Society**, v. 41, n. 11, p. 1069–1072, 1990.

CLARKEA, L.; JOHNSONA, E.; NEMHAUSERA, G.; ZHUB, Z. The aircraft rotation problem. **Operations Research**, v. 69, p. 33–46, 1997.

COHN, A. M.; BARNHART, C. Improving crew schedulingby incorporating key maintenance routing decisions. **Operations Research**, v. 51, n. 3, p. 387–396, May 2003.

CORDEAU, J. F.; STOJKOVIC, G.; SOUMIS, F.; DESROSIERS, J. Benders decomposition for simultaneous aircraft routing and crew scheduling. **Transportation Transportation Science**, v. 35, p. 375–388, 2001.

DIAS, T. **Algoritmos heurísticos e metaheurísticasa híbridas aplicadas ao planejamento de uma rede de telecomunicações com topologia anel-estrela.** Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, 2006.

FEO, T. A.; RESENDE, M. G. C. Greedy randomized adaptive search procedure. **Journal of Global Optimization**, v. 6, n. 2, p. 109–133, 1995.

HAOUARI, M.; SHERALI, H. D.; MANSOUR, F. Z.; AISSAOUI, N. Exact approaches for integrated aircraft fleeting and routing at tunisair. **Computacional Optimization and Applications**, 2011.

KLABJAN, D.; JOHNSON, E. L.; NEMHAUSER, G. L.; GELMAN, E.; RAMASWAMY, S. Airline crew scheduling with time windows and plane-count constraints. **Transportation Science**, v. 36, n. 3, p. 337–348, 2002.

LUCENA, A.; PONTES, R. **Aviação comercial controlada por máquinas inteligentes**. Rio de Janeiro: e-papers, 2007.

MERCIER, A.; SOUMIS, F. An integrated aircraft routing, crew scheduling and flight retiming model. **Computers and Operations Research**, v. 34, p. 2251–2265, 2007.

PIMENTEL, A. G. **Uma abordagem heurística para a solução de problemas de recobrimento de conjuntos de grande porte, com aplicação à alocação de tripulações para companhias aéreas**. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, 2005.

PONTES, R.; LUCENA, A.; CABRAL, L. **Exact and Heuristic Techniques for Solving the Aircraft Rotation Problem**. [S.l.], 2002.

PRAIS, M.; RIBEIRO, C. Variação de parâmetros em procedimentos grasp. **Investigación Operativa**, 2000.

SOUZA, M. J. F. **Inteligência Computacional para Otimização**. [S.l.], 2009.

## ANEXO A – REDE DE VOOS DA RIO-SUL

A rede de voos abaixo está organizada em 8 colunas que representam respectivamente:

- Número do voo: Sequência que representa a posição do voo na lista.
- Nome do voo: Nome que identifica o voo junto a companhia aérea.
- Dia de partida do voo: Representa em que dia o voo irá partir. O dia 0 (zero) representa o primeiro dia de operação do planejamento. Cada dia acrescentado acrescenta 24 horas ao horário de partida do voo.
- Horário de partida do voo: Identifica em que horário o voo irá partir.
- Dia de chegada do voo: Representa em que dia o voo irá chegar. O dia 0 (zero) representa o primeiro dia de operação do planejamento. Cada dia acrescentado acrescenta 24 horas ao horário de chegada do voo.
- Horário de chegada do voo: Identifica em que horário o voo irá chegar.
- Aeroporto de partida: Identifica o aeroporto de partida do voo.
- Aeroporto de chegada: Identifica o aeroporto de chegada do voo.

001 SL166 0 08:00 0 08:31 GYN BSB
002 SL155 0 08:32 0 10:33 GYN CGH
003 SL330 0 08:34 0 09:59 CGH VIX
004 SL595 0 08:40 0 11:11 POA CGH
005 SL596 0 08:44 0 10:39 CGH POA
006 SL533 0 08:50 0 10:01 BSB PLU
007 SL280 0 08:52 0 09:43 CGH CWB
008 SL587 0 09:00 0 10:01 PLU CGH
009 SL520 0 09:02 0 09:47 SDU PLU
010 JH559 0 09:04 0 10:35 BSB CGH
011 SL406 0 09:06 0 09:55 SDU CGH
012 SL419 0 09:16 0 11:03 FLN CGH
013 SL542 0 09:28 0 10:31 CGH PLU
014 SL408 0 09:30 0 10:21 SDU CGH
015 JH345 0 09:30 0 12:23 SSA CGH
016 SL589 0 09:58 0 11:01 PLU CGH
017 JH502 0 10:02 0 11:33 CGH BSB
018 SL281 0 10:08 0 10:59 CWB CGH
019 SL521 0 10:16 0 11:01 PLU SDU
020 SL532 0 10:20 0 11:31 PLU BSB
021 SL510 0 10:22 0 11:13 CGH CWB
022 SL544 0 10:32 0 11:33 CGH PLU
023 SL331 0 10:40 0 12:11 VIX CGH
055 SL598 0 16:00 0 17:55 CGH POA
056 SL568 0 16:02 0 17:03 CGH PLU
057 SL403 0 16:08 0 17:59 POA CGH
058 SL273 0 16:10 0 17:51 SJP CGH
059 SL391 0 16:12 0 17:55 FLN CGH
060 JH343 0 16:12 0 18:57 SSA CGH
061 SL481 0 17:00 0 17:45 CGH SDU
062 JH506 0 17:02 0 18:33 CGH BSB
063 SL405 0 17:14 0 19:57 CGH CGH
064 SL569 0 17:32 0 18:33 PLU CGH
065 SL152 0 17:44 0 19:35 CGH GYN
066 JH546 0 17:56 0 19:27 CGH BSB
067 SL572 0 17:58 0 18:59 CGH PLU
068 SL482 0 18:30 0 19:21 SDU CGH
069 JH341 0 18:50 1 00:19 JPA CGH
070 SL592 0 18:54 0 20:45 CGH POA
071 JH507 0 19:00 0 20:31 BSB CGH
072 SL528 0 19:02 0 19:47 SDU PLU
073 JH550 0 19:04 0 20:35 CGH BSB
074 SL537 0 19:22 0 20:33 BSB PLU
075 SL576 0 19:26 0 20:27 CGH PLU
076 SL573 0 19:34 0 20:35 PLU CGH
077 SL599 0 19:46 0 21:41 POA CGH

024 SL409 0 11:00 0 11:49 CGH SDU
025 SL543 0 11:00 0 12:03 PLU CGH
026 SL590 0 11:06 0 12:44 CGH POA
027 SL336 0 11:16 0 15:05 CGH REC
028 SL597 0 11:30 0 13:27 POA CGH
029 SL522 0 11:32 0 12:17 SDU PLU
030 SL150 0 11:42 0 13:33 CGH GYN
031 SL511 0 11:50 0 12:41 CWB CGH
032 JH503 0 12:02 0 13:33 BSB CGH
033 SL545 0 12:04 0 13:05 PLU CGH
034 SL470 0 12:28 0 13:19 SDU CGH
035 SL523 0 12:46 0 13:31 PLU SDU
036 JH342 0 13:00 0 15:45 CGH SSA
037 JH500 0 13:02 0 14:33 CGH BSB
038 SL147 0 13:10 0 18:03 BSB BSB
039 JH340 0 13:22 0 18:25 CGH JPA
040 SL591 0 13:30 0 14:57 POA CGH
041 SL402 0 13:38 0 15:33 CGH POA
042 SL471 0 13:54 0 14:39 CGH SDU
043 SL390 0 13:56 0 15:41 CGH FLN
044 SL564 0 13:58 0 14:59 CGH PLU
045 SL151 0 13:58 0 15:59 GYN CGH
046 SL563 0 14:00 0 15:01 PLU CGH
047 SL272 0 14:12 0 15:51 CGH SJP
048 SL282 0 14:22 0 15:13 CGH CWB
049 SL508 0 14:56 0 15:41 SDU PLU
050 JH501 0 14:58 0 16:29 BSB CGH
051 SL480 0 15:30 0 16:21 SDU CGH
052 SL283 0 15:42 0 16:37 CWB CGH
053 SL337 0 15:46 0 19:29 REC CGH
054 SL347 0 16:00 0 16:45 PLU SDU
078 JH547 0 20:00 0 21:31 BSB CGH
079 SL153 0 20:04 0 22:05 GYN CGH
080 SL483 0 20:08 0 20:53 CGH SDU
081 SL412 0 20:10 0 21:29 CGH FLN
082 SL529 0 20:14 0 20:59 PLU SDU
083 SL584 0 20:26 0 21:27 CGH PLU
084 SL514 0 20:30 0 21:21 CGH CWB
085 JH344 0 20:48 0 23:39 CGH SSA
086 SL577 0 21:02 0 22:03 PLU CGH
087 SL536 0 21:04 0 22:13 PLU BSB
088 JH551 0 21:04 0 22:35 BSB CGH
089 SL332 0 21:20 0 22:31 CGH VIX
090 SL492 0 21:30 0 22:21 SDU CGH
091 SL586 0 21:30 0 22:31 CGH PLU
092 SL530 0 21:34 0 22:19 SDU PLU
093 SL515 0 21:45 0 22:37 CWB CGH
094 SL593 0 21:52 0 23:19 POA CGH
095 SL585 0 22:04 0 23:05 PLU CGH
096 SL518 0 22:16 0 23:07 CGH CWB
097 SL413 0 22:22 0 23:31 FLN CGH
098 SL407 0 22:24 0 23:09 CGH SDU
099 SL167 0 22:42 0 23:13 BSB GYN
100 SL531 0 22:48 0 23:33 PLU SDU
101 JH558 0 22:56 1 00:27 CGH BSB
102 SL493 0 23:02 0 23:47 CGH SDU
103 SL333 0 23:04 1 00:15 VIX CGH
104 SL588 0 23:06 1 00:07 CGH PLU
105 SL594 0 23:18 1 01:57 CGH POA
106 SL154 0 23:22 1 01:13 CGH GYN
107 SL418 0 23:32 1 01:19 CGH FLN

## ANEXO B – TEMPO DOS VOOS DA RIO-SUL

A tabela abaixo está organizada em 3 colunas onde as duas primeiras representam dois aeroportos distintos e a terceira representa o tempo de voo em minutos entre esses dois aeroportos.

| | | | |
|---|---|---|---|
| BSB CGH 0091 | CGH SJP 0049 | FLN SSA 0225 | PLU SDU 0045 |
| BSB CWB 0125 | CGH SSA 0171 | FLN VIX 0136 | PLU SJP 0067 |
| BSB FLN 0152 | CGH VIX 0087 | GYN JPA 0216 | PLU SSA 0113 |
| BSB GYN 0031 | CWB FLN 0028 | GYN PLU 0076 | PLU VIX 0044 |
| BSB JPA 0197 | CWB GYN 0114 | GYN POA 0173 | POA REC 0342 |
| BSB PLU 0071 | CWB JPA 0293 | GYN REC 0210 | POA SDU 0129 |
| BSB POA 0185 | CWB PLU 0095 | GYN SDU 0109 | POA SJP 0119 |
| BSB REC 0191 | CWB POA 0062 | GYN SJP 0054 | POA SSA 0267 |
| BSB SDU 0107 | CWB REC 0283 | GYN SSA 0143 | POA VIX 0177 |
| BSB SJP 0066 | CWB SDU 0078 | GYN VIX 0118 | REC SDU 0215 |
| BSB SSA 0125 | CWB SJP 0060 | JPA PLU 0198 | REC SJP 0242 |
| BSB VIX 0109 | CWB SSA 0208 | JPA POA 0352 | REC SSA 0075 |
| CGH CWB 0051 | CWB VIX 0125 | JPA REC 0013 | REC VIX 0169 |
| CGH FLN 0105 | FLN GYN 0142 | JPA SDU 0226 | SDU SJP 0079 |
| CGH GYN 0095 | FLN JPA 0311 | JPA SJP 0251 | SDU SSA 0130 |
| CGH JPA 0255 | FLN PLU 0114 | JPA SSA 0085 | SDU VIX 0048 |
| CGH PLU 0058 | FLN POA 0042 | JPA VIX 0181 | SJP SSA 0169 |
| CGH POA 0097 | FLN REC 0300 | PLU POA 0155 | SJP VIX 0110 |
| CGH REC 0246 | FLN SDU 0087 | PLU REC 0187 | SSA VIX 0097 |
| CGH SDU 0049 | FLN SJP 0088 | | |

# ANEXO C – RESULTADO ÓTIMO DA INSTÂNCIA RIO-SUL

O resultado está organizado em vários blocos que representam as rotas que foram obtidas. Cada bloco inicia com um número que identifica a ordem em que a rota se encontra na lista, seguido de um número que identifica a quantidade de voos presentes nesta rota. Em seguida é descrito cada voo da rota seguindo o mesmo padrão utilizado na descrição da instância. Os voos de reposicionamento se encontram destacados em itálico.

Valor da função objetivo: 16158

Rota[01 - 9]
SL166 0 08:00 0 08:31 GYN BSB
JH559 0 09:04 0 10:35 BSB CGH
SL409 0 11:00 0 11:49 CGH SDU
SL470 0 12:28 0 13:19 SDU CGH
SL272 0 14:12 0 15:51 CGH SJP
SL273 0 16:10 0 17:51 SJP CGH(+1)
SL584 0 20:26 0 21:27 CGH PLU
SL585 0 22:04 0 23:05 PLU CGH
SL418 0 23:32 1 01:19 CGH FLN

Rota[02 - 7]
SL155 0 08:32 0 10:33 GYN CGH
SL590 0 11:06 0 12:44 CGH POA
SL591 0 13:30 0 14:57 POA CGH
SL572 0 17:58 0 18:59 CGH PLU
SL573 0 19:34 0 20:35 PLU CGH
SL332 0 21:20 0 22:31 CGH VIX
SL333 0 23:04 1 00:15 VIX CGH
JH551 0 21:04 0 22:35 BSB CGH
SL154 0 23:22 1 01:13 CGH GYN

Rota[03 - 9]
SL330 0 08:34 0 09:59 CGH VIX
SL331 0 10:40 0 12:11 VIX CGH
*REPO* 0 12:31 0 13:29 CGH PLU
SL563 0 14:00 0 15:01 PLU CGH
SL568 0 16:02 0 17:03 CGH PLU
SL569 0 17:32 0 18:33 PLU CGH
JH550 0 19:04 0 20:35 CGH BSB

Rota[04 - 6]
SL595 0 08:40 0 11:11 POA CGH
JH342 0 13:00 0 15:45 CGH SSA
JH343 0 16:12 0 18:57 SSA CGH
SL576 0 19:26 0 20:27 CGH PLU
SL577 0 21:02 0 22:03 PLU CGH
JH558 0 22:56 1 00:27 CGH BSB

Rota[05 - 5]
SL596 0 08:44 0 10:39 CGH POA
SL597 0 11:30 0 13:27 POA CGH
SL598 0 16:00 0 17:55 CGH POA
SL599 0 19:46 0 21:41 POA CGH
SL407 0 22:24 0 23:09 CGH SDU

Rota[06 - 7]
SL533 0 08:50 0 10:01 BSB PLU (-1)
SL532 0 10:20 0 11:31 PLU BSB
JH503 0 12:02 0 13:33 BSB CGH
SL282 0 14:22 0 15:13 CGH CWB
SL283 0 15:42 0 16:37 CWB CGH
SL405 0 17:14 0 19:57 CGH CGH
JH344 0 20:48 0 23:39 CGH SSA

Rota[07 - 7]
SL280 0 08:52 0 09:43 CGH CWB
SL281 0 10:08 0 10:59 CWB CGH
SL564 0 13:58 0 14:59 CGH PLU
SL347 0 16:00 0 16:45 PLU SDU
SL482 0 18:30 0 19:21 SDU CGH
SL412 0 20:10 0 21:29 CGH FLN
SL413 0 22:22 0 23:31 FLN CGH

Rota[08 - 7]
SL587 0 09:00 0 10:01 PLU CGH
SL510 0 10:22 0 11:13 CGH CWB
SL511 0 11:50 0 12:41 CWB CGH
SL390 0 13:56 0 15:41 CGH FLN
SL391 0 16:12 0 17:55 FLN CGH
SL592 0 18:54 0 20:45 CGH POA
SL593 0 21:52 0 23:19 POA CGH

Rota[09 - 9]
SL520 0 09:02 0 09:47 SDU PLU
SL521 0 10:16 0 11:01 PLU SDU
SL522 0 11:32 0 12:17 SDU PLU
SL523 0 12:46 0 13:31 PLU SDU
SL508 0 14:56 0 15:41 SDU PLU
*REPO* 0 16:01 0 16:59 PLU CGH
SL152 0 17:44 0 19:35 CGH GYN
SL153 0 20:04 0 22:05 GYN CGH
SL588 0 23:06 1 00:07 CGH PLU

Rota[10 - 10]
SL406 0 09:06 0 09:55 SDU CGH
SL544 0 10:32 0 11:33 CGH PLU
SL545 0 12:04 0 13:05 PLU CGH
SL471 0 13:54 0 14:39 CGH SDU
SL480 0 15:30 0 16:21 SDU CGH
SL481 0 17:00 0 17:45 CGH SDU
SL528 0 19:02 0 19:47 SDU PLU
SL529 0 20:14 0 20:59 PLU SDU
SL492 0 21:30 0 22:21 SDU CGH
SL493 0 23:02 0 23:47 CGH SDU

Rota[11 - 6]
SL419 0 09:16 0 11:03 FLN CGH
SL150 0 11:42 0 13:33 CGH GYN
SL151 0 13:58 0 15:59 GYN CGH
JH546 0 17:56 0 19:27 CGH BSB
JH547 0 20:00 0 21:31 BSB CGH
SL518 0 22:16 0 23:07 CGH CWB

Rota[12 - 7]
SL542 0 09:28 0 10:31 CGH PLU
SL543 0 11:00 0 12:03 PLU CGH
SL402 0 13:38 0 15:33 CGH POA
SL403 0 16:08 0 17:59 POA CGH
SL514 0 20:30 0 21:21 CGH CWB
SL515 0 21:45 0 22:37 CWB CGH
SL594 0 23:18 1 01:57 CGH POA

Rota[13 - 6]
SL408 0 09:30 0 10:21 SDU CGH
SL336 0 11:16 0 15:05 CGH REC
SL337 0 15:46 0 19:29 REC CGH
SL483 0 20:08 0 20:53 CGH SDU
SL530 0 21:34 0 22:19 SDU PLU
SL531 0 22:48 0 23:33 PLU SDU

Rota[14 - 6]
JH345 0 09:30 0 12:23 SSA CGH
JH500 0 13:02 0 14:33 CGH BSB
JH501 0 14:58 0 16:29 BSB CGH
JH506 0 17:02 0 18:33 CGH BSB
JH507 0 19:00 0 20:31 BSB CGH
SL586 0 21:30 0 22:31 CGH PLU

Rota[15 - 3]
SL589 0 09:58 0 11:01 PLU CGH
JH340 0 13:22 0 18:25 CGH JPA
JH341 0 18:50 1 00:19 JPA CGH

Rota[16 - 5]
JH502 0 09:42 0 11:33 CGH BSB
SL147 0 12:50 0 18:03 BSB BSB
SL537 0 19:02 0 20:33 BSB PLU
SL536 0 20:44 0 22:13 PLU BSB
SL167 0 22:22 0 23:13 BSB GYN

## ANEXO D – REDE DE VOOS DA TAM

A rede de voos abaixo está organizada em 8 colunas que representam respectivamente:

- Número do voo: Sequência que representa a posição do voo na lista.
- Nome do voo: Nome que identifica o voo junto a companhia aérea.
- Dia de partida do voo: Representa em que dia o voo irá partir. O dia 0 (zero) representa o primeiro dia de operação do planejamento. Cada dia acrescentado acrescenta 24 horas ao horário de partida do voo.
- Horário de partida do voo: Identifica em que horário o voo irá partir.
- Dia de chegada do voo: Representa em que dia o voo irá chegar. O dia 0 (zero) representa o primeiro dia de operação do planejamento. Cada dia acrescentado acrescenta 24 horas ao horário de chegada do voo.
- Horário de chegada do voo: Identifica em que horário o voo irá chegar.
- Aeroporto de partida: Identifica o aeroporto de partida do voo.
- Aeroporto de chegada: Identifica o aeroporto de chegada do voo.

001 JJ3458 0 00:05 0 01:05 MAB BEL
002 JJ3585 0 01:10 0 06:21 RBR BSB
003 JJ3585 0 01:10 0 06:21 RBR BSB
004 JJ3595 0 01:30 0 06:16 PVH BSB
005 JJ3459 0 01:35 0 06:25 BEL CNF
006 JJ3459 0 01:35 0 02:35 BEL MAB
007 JJ3069 0 01:50 0 02:51 CGB CGR
008 JJ3739 0 02:30 0 05:59 FOR BSB
009 JJ2101 0 02:40 0 04:10 NAT SSA
010 JJ3201 0 02:40 0 07:20 NAT CNF
011 JJ3201 0 02:40 0 09:22 NAT CGH
012 JJ3065 0 02:50 0 06:22 AJU SDU
013 JJ3459 0 03:10 0 06:25 MAB CNF
014 JJ3409 0 03:23 0 05:27 BPS CNF
015 JJ3069 0 03:24 0 06:24 CGR SDU
016 JJ3775 0 03:35 0 06:03 CGR CGH
017 JJ3201 0 04:45 0 07:20 SSA CNF
018 JJ3201 0 04:45 0 09:22 SSA CGH
019 JJ3737 0 05:50 0 06:45 SJP CGH
020 JJ4723 0 06:00 0 07:43 BSB CGH
021 JJ3597 0 06:00 0 08:42 CGB BSB
022 JJ3770 0 06:00 0 07:00 GRU IOS
023 JJ3900 0 06:04 0 07:04 CGH SDU
122 JJ3928 0 13:00 0 14:00 CGH SDU
123 JJ3871 0 13:15 0 14:15 BEL MAB
124 JJ3365 0 13:15 0 16:00 IOS SDU
125 JJ3929 0 13:15 0 14:07 SDU CGH
126 JJ3571 0 13:25 0 15:56 CGR BSB
127 JJ3038 0 13:25 0 14:29 JOI CGH
128 JJ3563 0 13:30 0 18:43 RBR BSB
129 JJ3563 0 13:30 0 18:43 RBR BSB
130 JJ3930 0 13:30 0 14:27 CGH SDU
131 JJ3815 0 13:35 0 15:51 PMW BSB
132 JJ3708 0 13:43 0 15:19 CGH BSB
133 JJ3931 0 13:45 0 14:35 SDU CGH
134 JJ3759 0 13:50 0 14:52 CNF SDU
135 JJ3745 0 13:55 0 14:56 SJP CGH
136 JJ3120 0 14:00 0 15:02 NVT CGH
137 JJ3932 0 14:00 0 15:00 CGH SDU
138 JJ3661 0 14:12 0 17:14 IOS CGH
139 JJ3933 0 14:16 0 15:14 SDU CGH
140 JJ3934 0 14:28 0 15:30 CGH SDU
141 JJ3264 0 14:30 0 15:53 CWB SDU
142 JJ3935 0 14:45 0 15:44 SDU CGH
143 JJ3628 0 14:54 0 16:13 CGH SSA
144 JJ4737 0 14:55 0 18:02 CGB CGH

024 JJ3100 0 06:05 0 07:15 FLN CGH
025 JJ3211 0 06:07 0 07:21 CNF CGH
026 JJ3901 0 06:15 0 07:07 SDU CGH
027 JJ3768 0 06:20 0 07:27 LDB CGH
028 JJ3119 0 06:24 0 07:35 CGH NVT
029 JJ3902 0 06:30 0 07:30 CGH SDU
030 JJ3758 0 06:34 0 07:47 SDU CNF
031 JJ3903 0 06:45 0 07:49 SDU CGH
032 JJ3242 0 06:47 0 07:50 CGH UDI
033 JJ3035 0 06:49 0 08:00 CGH JOI
034 JJ3370 0 06:52 0 07:49 CGH GIG
035 JJ3370 0 06:52 0 10:27 CGH REC
036 JJ3732 0 06:52 0 08:12 CGH SSA
037 JJ3753 0 07:00 0 08:04 CNF SDU
038 JJ3666 0 07:00 0 07:45 SDU IOS
039 JJ3904 0 07:00 0 08:02 CGH SDU
040 JJ3022 0 07:06 0 08:56 SDU BSB
041 JJ3855 0 07:10 0 08:15 BSB CNF
042 JJ3023 0 07:12 0 09:02 BSB SDU
043 JJ3905 0 07:16 0 08:13 SDU CGH
044 JJ3906 0 07:30 0 08:29 CGH SDU
045 JJ3907 0 07:45 0 08:51 SDU CGH
046 JJ3740 0 07:55 0 08:55 CGH SJP
047 JJ3740 0 07:55 0 10:05 CGH CGB
048 JJ3201 0 08:00 0 09:22 CNF CGH
049 JJ3908 0 08:00 0 09:00 CGH SDU
050 JJ3130 0 08:12 0 09:35 CGH VIX
051 JJ3667 0 08:15 0 11:00 IOS SDU
052 JJ3118 0 08:15 0 09:09 NVT CGH
053 JJ3909 0 08:15 0 09:15 SDU CGH
054 JJ3411 0 08:15 0 09:22 VIX GIG
055 JJ3269 0 08:18 0 09:48 GIG CWB
056 JJ3385 0 08:30 0 09:38 CNF GIG
057 JJ3910 0 08:30 0 09:30 CGH SDU
058 JJ3239 0 08:30 0 09:41 UDI CGH
059 JJ3032 0 08:40 0 09:52 JOI CGH
060 JJ3261 0 08:42 0 10:27 SDU BSB
061 JJ3911 0 08:44 0 09:55 SDU CGH
062 JJ3911 0 08:44 0 09:55 SDU CGH
063 JJ3370 0 08:45 0 10:27 GIG REC
064 JJ3756 0 08:49 0 09:55 SDU CNF
065 JJ3733 0 08:54 0 12:09 SSA CGH
066 JJ3850 0 08:59 0 10:14 CNF BSB
067 JJ3912 0 09:00 0 10:00 CGH SDU
068 JJ3913 0 09:15 0 10:21 SDU CGH
069 JJ3914 0 09:30 0 10:30 CGH SDU
070 JJ3740 0 09:35 0 10:05 SJP CGB
071 JJ3024 0 09:42 0 11:27 SDU BSB
072 JJ3915 0 09:45 0 10:52 SDU CGH
073 JJ3025 0 09:52 0 11:37 BSB SDU
074 JJ3127 0 10:00 0 11:29 VIX CGH
075 JJ3411 0 10:02 0 12:05 GIG POA
076 JJ3916 0 10:02 0 11:00 CGH SDU
077 JJ3372 0 10:04 0 12:12 CGH SSA
078 JJ3660 0 10:04 0 10:58 CGH IOS
079 JJ3660 0 10:04 0 12:12 CGH SSA
080 JJ3212 0 10:06 0 11:10 CGH CNF
081 JJ3604 0 10:06 0 11:25 CGH SSA
145 JJ3936 0 15:00 0 16:00 CGH SDU
146 JJ3077 0 15:15 0 19:15 REC GIG
147 JJ3077 0 15:15 0 21:11 REC CGH
148 JJ3937 0 15:15 0 16:11 SDU CGH
149 JJ3107 0 15:17 0 16:20 CGH FLN
150 JJ3027 0 15:18 0 17:04 BSB SDU
151 JJ3410 0 15:22 0 16:20 GIG VIX
152 JJ3938 0 15:30 0 16:32 CGH SDU
153 JJ3013 0 15:35 0 16:27 CGH CWB
154 JJ3744 0 15:42 0 16:40 CGH SJP
155 JJ3826 0 15:43 0 17:19 SDU BSB
156 JJ3939 0 15:45 0 16:35 SDU CGH
157 JJ3652 0 15:57 0 16:40 CGH CGR
158 JJ3940 0 16:01 0 17:02 CGH SDU
159 JJ3723 0 16:03 0 17:38 BSB CGH
160 JJ3941 0 16:15 0 17:11 SDU CGH
161 JJ3029 0 16:22 0 17:58 BSB SDU
162 JJ3942 0 16:30 0 17:28 CGH SDU
163 JJ3857 0 16:41 0 17:52 BSB CNF
164 JJ3943 0 16:45 0 17:34 SDU CGH
165 JJ3263 0 16:47 0 18:55 SDU POA
166 JJ3629 0 16:52 0 20:15 SSA CGH
167 JJ3104 0 16:55 0 17:56 FLN CGH
168 JJ3267 0 16:56 0 18:30 SDU CWB
169 JJ3944 0 16:59 0 18:06 CGH SDU
170 JJ3966 0 16:59 0 18:06 CGH SDU
171 JJ3137 0 17:00 0 18:28 VIX CGH
172 JJ3012 0 17:02 0 17:50 CWB CGH
173 JJ3123 0 17:07 0 18:08 CGH NVT
174 JJ3945 0 17:15 0 18:08 SDU CGH
175 JJ3743 0 17:15 0 18:14 SJP CGH
176 JJ3946 0 17:29 0 18:32 CGH SDU
177 JJ3947 0 17:44 0 18:34 SDU CGH
178 JJ3653 0 17:50 0 20:29 CGR CGH
179 JJ3028 0 17:54 0 19:43 SDU BSB
180 JJ3109 0 17:58 0 19:10 CGH FLN
181 JJ3948 0 18:00 0 19:04 CGH SDU
182 JJ3949 0 18:15 0 19:05 SDU CGH
183 JJ3767 0 18:23 0 19:25 CGH LDB
184 JJ3755 0 18:26 0 19:26 CNF SDU
185 JJ3950 0 18:29 0 19:30 CGH SDU
186 JJ3827 0 18:32 0 20:01 BSB GIG
187 JJ3224 0 18:38 0 19:55 CGH CNF
188 JJ3951 0 18:46 0 19:37 SDU CGH
189 JJ3033 0 18:46 0 19:45 CGH JOI
190 JJ3122 0 18:48 0 19:49 NVT CGH
191 JJ3966 0 18:48 0 19:15 SDU CFB
192 JJ3952 0 19:00 0 20:06 CGH SDU
193 JJ3238 0 19:04 0 20:10 CGH UDI
194 JJ3712 0 19:12 0 20:58 CGH BSB
195 JJ3268 0 19:14 0 20:46 CWB GIG
196 JJ3953 0 19:15 0 20:15 SDU CGH
197 JJ3276 0 19:24 0 20:22 CGH RAO
198 JJ3954 0 19:29 0 20:28 CGH SDU
199 JJ3563 0 19:30 0 21:05 BSB GRU
200 JJ3262 0 19:30 0 21:28 POA SDU
201 JJ3955 0 19:44 0 20:38 SDU CGH
202 JJ3967 0 19:45 0 20:12 CFB SDU

082 JJ3917 0 10:15 0 11:12 SDU CGH
083 JJ3053 0 10:16 0 11:37 CGH POA
084 JJ3856 0 10:29 0 11:43 CNF BSB
085 JJ3820 0 10:30 0 12:20 GIG BSB
086 JJ3918 0 10:30 0 11:28 CGH SDU
087 JJ3246 0 10:32 0 11:39 CGH UDI
088 JJ3266 0 10:36 0 11:57 CWB SDU
089 JJ3745 0 10:45 0 13:15 CGB SJP
090 JJ3745 0 10:45 0 14:56 CGB CGH
091 JJ3919 0 10:45 0 11:48 SDU CGH
092 JJ3920 0 11:00 0 12:00 CGH SDU
093 JJ3574 0 11:04 0 12:20 BSB RBR
094 JJ3825 0 11:12 0 12:50 BSB SDU
095 JJ3921 0 11:16 0 12:24 SDU CGH
096 JJ3922 0 11:30 0 12:25 CGH SDU
097 JJ3660 0 11:33 0 12:12 IOS SSA
098 JJ3923 0 11:45 0 12:48 SDU CGH
099 JJ3215 0 11:50 0 12:57 CNF CGH
100 JJ3039 0 11:50 0 12:50 CGH JOI
101 JJ3364 0 12:00 0 12:45 SDU IOS
102 JJ3605 0 12:00 0 15:34 SSA CGH
103 JJ3924 0 12:00 0 13:00 CGH SDU
104 JJ3754 0 12:04 0 13:03 SDU CNF
105 JJ3260 0 12:06 0 13:31 BSB SDU
106 JJ3243 0 12:10 0 13:11 UDI CGH
107 JJ3925 0 12:15 0 13:14 SDU CGH
108 JJ3265 0 12:19 0 13:45 SDU CWB
109 JJ3570 0 12:21 0 12:50 BSB CGR
110 JJ3052 0 12:22 0 13:48 POA CGH
111 JJ3121 0 12:22 0 13:25 CGH NVT
112 JJ3926 0 12:30 0 13:33 CGH SDU
113 JJ3814 0 12:41 0 12:55 BSB PMW
114 JJ3026 0 12:43 0 14:28 SDU BSB
115 JJ3927 0 12:45 0 13:41 SDU CGH
116 JJ3410 0 12:47 0 14:41 POA GIG
117 JJ3373 0 12:52 0 17:14 SSA CGH
118 JJ3661 0 12:52 0 13:37 SSA IOS
119 JJ3661 0 12:52 0 17:14 SSA CGH
120 JJ3963 0 12:58 0 14:01 SDU CGH
121 JJ3709 0 13:00 0 14:38 BSB CGH
203 JJ3110 0 19:50 0 21:05 FLN CGH
204 JJ3752 0 19:54 0 20:54 SDU CNF
205 JJ3956 0 19:59 0 21:03 CGH SDU
206 JJ3764 0 20:05 0 21:11 LDB CGH
207 JJ4722 0 20:05 0 21:44 CGH BSB
208 JJ3957 0 20:15 0 21:08 SDU CGH
209 JJ3226 0 20:19 0 21:36 CGH CNF
210 JJ3226 0 20:19 0 22:59 CGH SSA
211 JJ3226 0 20:19 1 01:05 CGH NAT
212 JJ3034 0 20:20 0 21:17 JOI CGH
213 JJ3757 0 20:33 0 21:34 CNF SDU
214 JJ3030 0 20:37 0 22:23 SDU BSB
215 JJ3959 0 20:45 0 21:37 SDU CGH
216 JJ3967 0 20:45 0 21:37 SDU CGH
217 JJ3245 0 20:45 0 21:55 UDI CGH
218 JJ3031 0 20:46 0 22:05 BSB SDU
219 JJ3105 0 20:46 0 21:53 CGH FLN
220 JJ3958 0 21:00 0 21:55 CGH SDU
221 JJ3275 0 21:02 0 22:04 RAO CGH
222 JJ3961 0 21:16 0 22:15 SDU CGH
223 JJ3854 0 21:28 0 22:44 CNF BSB
224 JJ3960 0 21:29 0 22:19 CGH SDU
225 JJ3769 0 21:42 0 22:40 CGH LDB
226 JJ3736 0 21:56 0 22:57 CGH SJP
227 JJ3077 0 22:02 0 23:11 GIG CGH
228 JJ3064 0 22:07 0 23:40 SDU AJU
229 JJ3774 0 22:07 0 22:47 CGH CGR
230 JJ3068 0 22:09 0 23:20 SDU CGR
231 JJ3226 0 22:20 0 22:59 CNF SSA
232 JJ3226 0 22:20 1 01:05 CNF NAT
233 JJ3177 0 22:20 0 23:35 GRU FLN
234 JJ3458 0 22:25 0 23:35 CNF MAB
235 JJ3458 0 22:25 1 01:05 CNF BEL
236 JJ3584 0 23:11 1 00:20 BSB RBR
237 JJ3594 0 23:26 1 00:25 BSB PVH
238 JJ3226 0 23:35 1 01:05 SSA NAT
239 JJ3738 0 23:47 1 01:20 BSB FOR
240 JJ3408 0 23:50 1 00:59 CNF BPS
241 JJ3068 0 23:55 1 00:55 CGR CGB

## ANEXO E – TEMPO DOS VOOS DA TAM

A tabela abaixo está organizada em 3 colunas onde as duas primeiras representam dois aeroportos distintos e a terceira representa o tempo de voo em minutos entre esses aeroportos.

| | | | |
|---|---|---|---|
| AJU SDU 0212 | BSB SDU 0110 | CGH NVT 0071 | CNF SDU 0073 |
| BEL CNF 0290 | CFB SDU 0027 | CGH POA 0086 | CNF SSA 0155 |
| BEL MAB 0060 | CGB CGH 0251 | CGH RAO 0062 | CWB GIG 0092 |
| BPS CNF 0124 | CGB CGR 0061 | CGH REC 0356 | CWB SDU 0094 |
| BSB CGB 0162 | CGB SJP 0150 | CGH SDU 0071 | FLN GRU 0075 |
| BSB CGH 0106 | CGH CGR 0159 | CGH SJP 0061 | GIG POA 0123 |
| BSB CGR 0151 | CGH CNF 0082 | CGH SSA 0277 | GIG REC 0240 |
| BSB CNF 0076 | CGH CWB 0052 | CGH UDI 0071 | GIG VIX 0067 |
| BSB FOR 0209 | CGH FLN 0075 | CGH VIX 0089 | GRU IOS 0060 |
| BSB GIG 0110 | CGH GIG 0069 | CGR SDU 0180 | IOS SDU 0165 |
| BSB GRU 0095 | CGH IOS 0182 | CNF GIG 0068 | IOS SSA 0045 |
| BSB PMW 0136 | CGH JOI 0072 | CNF MAB 0195 | NAT SSA 0090 |
| BSB PVH 0286 | CGH LDB 0067 | CNF NAT 0280 | POA SDU 0128 |
| BSB RBR 0313 | CGH NAT 0402 | | |

*O tempo de voos entre alguns aeroportos foram omitidos por falta de informação suficiente para inferi-las.

# ANEXO F – RESULTADO ÓTIMO DA INSTÂNCIA TAM

O resultado está organizado em vários blocos que representam as rotas que foram obtidas. Cada bloco inicia com um número que identifica a ordem em que a rota se encontra na lista, seguido de um número que identifica a quantidade de voos presentes nesta rota. Em seguida é descrito cada voo da rota seguindo o mesmo padrão utilizado na descrição da instância. Os voos de reposicionamento se encontram destacados em itálico.

Valor da função objetivo: 35015

Rota[01 - 8]
JJ3458 0 00:05 0 01:05 MAB BEL
JJ3459 0 01:35 0 06:25 BEL CNF
*REPO* 0 06:45 0 07:58 CNF SDU
JJ3909 0 08:15 0 09:15 SDU CGH(+3)
JJ3660 0 10:04 0 12:12 CGH SSA
JJ3661 0 12:52 0 13:37 SSA IOS
JJ3661 0 14:12 0 17:14 IOS CGH
JJ3105 0 20:46 0 21:53 CGH FLN

Rota[02 - 5]
JJ3585 0 01:10 0 06:21 RBR BSB
JJ3855 0 07:10 0 08:15 BSB CNF
JJ3850 0 08:59 0 10:14 CNF BSB
JJ3574 0 11:04 0 12:20 BSB RBR
JJ3563 0 13:30 0 18:43 RBR BSB

Rota[03 - 8]
JJ3585 0 01:10 0 06:21 RBR BSB
JJ3023 0 07:12 0 09:02 BSB SDU
JJ3915 0 09:45 0 10:52 SDU CGH
JJ3922 0 11:30 0 12:25 CGH SDU
JJ3963 0 12:58 0 14:01 SDU CGH
JJ3628 0 14:54 0 16:13 CGH SSA
JJ3629 0 16:52 0 20:15 SSA CGH
JJ3958 0 21:00 0 21:55 CGH SDU

Rota[04 - 4]
JJ3595 0 01:30 0 06:16 PVH BSB
*REPO* 0 06:36 0 08:26 BSB GIG
JJ3370 0 08:45 0 10:27 GIG REC(+1)
JJ3077 0 15:15 0 21:11 REC CGH

Rota[05 - 9]
JJ3459 0 01:35 0 02:35 BEL MAB
JJ3459 0 03:10 0 06:25 MAB CNF
JJ3753 0 07:00 0 08:04 CNF SDU
JJ3261 0 08:42 0 10:27 SDU BSB
JJ3825 0 11:12 0 12:50 BSB SDU
JJ3929 0 13:15 0 14:07 SDU CGH
JJ3744 0 15:42 0 16:40 CGH SJP
JJ3743 0 17:15 0 18:14 SJP CGH
JJ3769 0 21:42 0 22:40 CGH LDB

Rota[06 - 10]
JJ3069 0 01:50 0 02:51 CGB CGR
JJ3069 0 03:24 0 06:24 CGR SDU
JJ3903 0 06:45 0 07:49 SDU CGH
JJ3130 0 08:12 0 09:35 CGH VIX
JJ3127 0 10:00 0 11:29 VIX CGH
JJ3039 0 11:50 0 12:50 CGH JOI
JJ3038 0 13:25 0 14:29 JOI CGH
JJ3942 0 16:30 0 17:28 CGH SDU
JJ3028 0 17:54 0 19:43 SDU BSB
JJ3031 0 20:46 0 22:05 BSB SDU

Rota[07 - 8]
JJ3739 0 02:30 0 05:59 FOR BSB
*REPO* 0 06:19 0 08:09 BSB SDU
JJ3911 0 08:44 0 09:55 SDU CGH
JJ3053 0 10:16 0 11:37 CGH POA
JJ3052 0 12:22 0 13:48 POA CGH
JJ3940 0 16:01 0 17:02 CGH SDU
JJ3752 0 19:54 0 20:54 SDU CNF
JJ3854 0 21:28 0 22:44 CNF BSB

Rota[08 - 6]
JJ2101 0 02:40 0 04:10 NAT SSA
JJ3201 0 04:45 0 09:22 SSA CGH
JJ3604 0 10:06 0 11:25 CGH SSA
JJ3605 0 12:00 0 15:34 SSA CGH
JJ3652 0 15:57 0 16:40 CGH CGR
JJ3653 0 17:50 0 20:29 CGR CGH

Rota[09 - 10]
JJ3201 0 02:40 0 07:20 NAT CNF
JJ3201 0 08:00 0 09:22 CNF CGH
JJ3212 0 10:06 0 11:10 CGH CNF
JJ3215 0 11:50 0 12:57 CNF CGH
JJ3930 0 13:30 0 14:27 CGH SDU
JJ3939 0 15:45 0 16:35 SDU CGH
JJ3123 0 17:07 0 18:08 CGH NVT
JJ3122 0 18:48 0 19:49 NVT CGH
JJ3226 0 20:19 0 21:36 CGH CNF
JJ3226 0 22:20 1 01:05 CNF NAT

Rota[10 - 5]
JJ3201 0 02:40 0 09:22 NAT CGH
JJ3372 0 10:04 0 12:12 CGH SSA
JJ3661 0 12:52 0 17:14 SSA CGH
JJ3948 0 18:00 0 19:04 CGH SDU
JJ3967 0 20:45 0 21:37 SDU CGH

Rota[11 - 8]
JJ3065 0 02:50 0 06:22 AJU SDU
JJ3666 0 07:00 0 07:45 SDU IOS
JJ3667 0 08:15 0 11:00 IOS SDU
JJ3925 0 12:15 0 13:14 SDU CGH
JJ3708 0 13:43 0 15:19 CGH BSB
JJ3723 0 16:03 0 17:38 BSB CGH
JJ3954 0 19:29 0 20:28 CGH SDU
JJ3961 0 21:16 0 22:15 SDU CGH

Rota[12 - 8]
JJ3409 0 03:23 0 05:27 BPS CNF
JJ3211 0 06:07 0 07:21 CNF CGH
JJ3908 0 08:00 0 09:00 CGH SDU
JJ3024 0 09:42 0 11:27 SDU BSB
JJ3814 0 12:41 0 12:55 BSB PMW
JJ3815 0 13:35 0 15:51 PMW BSB
JJ3029 0 16:22 0 17:58 BSB SDU
JJ3955 0 19:44 0 20:38 SDU CGH

Rota[13 - 8]
JJ3775 0 03:35 0 06:03 CGR CGH
JJ3242 0 06:47 0 07:50 CGH UDI
JJ3239 0 08:30 0 09:41 UDI CGH
JJ3660 0 10:04 0 10:58 CGH IOS
JJ3660 0 11:33 0 12:12 IOS SSA
JJ3373 0 12:52 0 17:14 SSA CGH
JJ3238 0 19:04 0 20:10 CGH UDI
JJ3245 0 20:45 0 21:55 UDI CGH

Rota[14 - 8]
JJ3201 0 04:45 0 07:20 SSA CNF
*REPO* 0 07:40 0 08:53 CNF SDU
JJ3913 0 09:15 0 10:21 SDU CGH
JJ3920 0 11:00 0 12:00 CGH SDU
JJ3026 0 12:43 0 14:28 SDU BSB
JJ3027 0 15:18 0 17:04 BSB SDU
JJ3949 0 18:15 0 19:05 SDU CGH
JJ3226 0 20:19 0 22:59 CGH SSA

Rota[15 - 10]
JJ3737 0 05:50 0 06:45 SJP CGH(-5)
JJ3904 0 07:00 0 08:02 CGH SDU
JJ3911 0 08:44 0 09:55 SDU CGH
JJ3918 0 10:30 0 11:28 CGH SDU
JJ3754 0 12:04 0 13:03 SDU CNF
JJ3759 0 13:50 0 14:52 CNF SDU
JJ3826 0 15:43 0 17:19 SDU BSB
JJ3827 0 18:32 0 20:01 BSB GIG
*REPO* 0 20:21 0 21:29 GIG CNF
JJ3458 0 22:25 0 23:35 CNF MAB

Rota[16 - 8]
JJ4723 0 06:00 0 07:43 BSB CGH
JJ3910 0 08:30 0 09:30 CGH SDU
JJ3917 0 10:15 0 11:12 SDU CGH
JJ3924 0 12:00 0 13:00 CGH SDU
JJ3931 0 13:45 0 14:35 SDU CGH
JJ3224 0 18:38 0 19:55 CGH CNF
JJ3757 0 20:33 0 21:34 CNF SDU
JJ3064 0 22:07 0 23:40 SDU AJU

Rota[17 - 6]
JJ3597 0 06:00 0 08:42 CGB BSB
JJ3570 0 12:21 0 12:50 BSB CGR
JJ3571 0 13:25 0 15:56 CGR BSB
JJ3857 0 16:41 0 17:52 BSB CNF
JJ3755 0 18:26 0 19:26 CNF SDU
JJ3957 0 20:15 0 21:08 SDU CGH

Rota[18 - 7]
JJ3770 0 06:00 0 07:00 GRU IOS
*REPO* 0 07:20 0 10:05 IOS SDU
JJ3923 0 11:45 0 12:48 SDU CGH
JJ3934 0 14:28 0 15:30 CGH SDU
JJ3943 0 16:45 0 17:34 SDU CGH
JJ4722 0 20:05 0 21:44 CGH BSB
JJ3584 0 23:11 1 00:20 BSB RBR

Rota[19 - 9]
JJ3900 0 06:04 0 07:04 CGH SDU(-8)
JJ3905 0 07:16 0 08:13 SDU CGH
JJ3912 0 09:00 0 10:00 CGH SDU
JJ3919 0 10:45 0 11:48 SDU CGH
JJ3121 0 12:22 0 13:25 CGH NVT
JJ3120 0 14:00 0 15:02 NVT CGH
JJ3946 0 17:29 0 18:32 CGH SDU
JJ3953 0 19:15 0 20:15 SDU CGH
JJ3960 0 21:29 0 22:19 CGH SDU

Rota[20 - 9]
JJ3100 0 06:05 0 07:15 FLN CGH
JJ3740 0 07:55 0 10:05 CGH CGB
JJ3745 0 10:45 0 13:15 CGB SJP
JJ3745 0 13:55 0 14:56 SJP CGH
JJ3107 0 15:17 0 16:20 CGH FLN
JJ3104 0 16:55 0 17:56 FLN CGH
JJ3950 0 18:29 0 19:30 CGH SDU
JJ3030 0 20:37 0 22:23 SDU BSB
JJ3594 0 23:26 1 00:25 BSB PVH

Rota[21 - 12]
JJ3901 0 06:15 0 07:07 SDU CGH
JJ3906 0 07:30 0 08:29 CGH SDU
JJ3756 0 08:49 0 09:55 SDU CNF
JJ3856 0 10:29 0 11:43 CNF BSB
JJ3260 0 12:06 0 13:31 BSB SDU
JJ3933 0 14:16 0 15:14 SDU CGH
JJ3013 0 15:35 0 16:27 CGH CWB
JJ3012 0 17:02 0 17:50 CWB CGH
JJ3767 0 18:23 0 19:25 CGH LDB
JJ3764 0 20:05 0 21:11 LDB CGH
JJ3774 0 22:07 0 22:47 CGH CGR
JJ3068 0 23:55 1 00:55 CGR CGB

Rota[22 - 7]
JJ3768 0 06:20 0 07:27 LDB CGH
JJ3740 0 07:55 0 08:55 CGH SJP
JJ3740 0 09:35 0 10:05 SJP CGB
JJ4737 0 14:55 0 18:02 CGB CGH
JJ3033 0 18:46 0 19:45 CGH JOI
JJ3034 0 20:20 0 21:17 JOI CGH
JJ3736 0 21:56 0 22:57 CGH SJP

Rota[23 - 8]
JJ3119 0 06:24 0 07:35 CGH NVT
JJ3118 0 08:15 0 09:09 NVT CGH
JJ3914 0 09:30 0 10:30 CGH SDU
JJ3921 0 11:16 0 12:24 SDU CGH
JJ3928 0 13:00 0 14:00 CGH SDU
JJ3937 0 15:15 0 16:11 SDU CGH
JJ3109 0 17:58 0 19:10 CGH FLN
JJ3110 0 19:50 0 21:05 FLN CGH

Rota[24 - 9]
JJ3902 0 06:30 0 07:30 CGH SDU(-5)
JJ3907 0 07:45 0 08:51 SDU CGH
JJ3916 0 10:02 0 11:00 CGH SDU
JJ3364 0 12:00 0 12:45 SDU IOS
JJ3365 0 13:15 0 16:00 IOS SDU
JJ3263 0 16:47 0 18:55 SDU POA
JJ3262 0 19:30 0 21:28 POA SDU
*REPO* 0 21:48 0 23:01 SDU CNF
JJ3408 0 23:50 1 00:59 CNF BPS

Rota[25 - 8]
JJ3758 0 06:34 0 07:47 SDU CNF
JJ3385 0 08:30 0 09:38 CNF GIG
JJ3820 0 10:30 0 12:20 GIG BSB
JJ3709 0 13:00 0 14:38 BSB CGH
JJ3944 0 16:59 0 18:06 CGH SDU
JJ3951 0 18:46 0 19:37 SDU CGH
JJ3956 0 19:59 0 21:03 CGH SDU
JJ3068 0 22:09 0 23:20 SDU CGR

Rota[26 - 8]
JJ3035 0 06:49 0 08:00 CGH JOI
JJ3032 0 08:40 0 09:52 JOI CGH
JJ3246 0 10:32 0 11:39 CGH UDI
JJ3243 0 12:10 0 13:11 UDI CGH
JJ3932 0 14:00 0 15:00 CGH SDU
JJ3947 0 17:44 0 18:34 SDU CGH
JJ3712 0 19:12 0 20:58 CGH BSB
JJ3738 0 23:47 1 01:20 BSB FOR

Rota[27 - 8]
JJ3370 0 06:52 0 07:49 CGH GIG
JJ3269 0 08:18 0 09:48 GIG CWB
JJ3266 0 10:36 0 11:57 CWB SDU
JJ3927 0 12:45 0 13:41 SDU CGH
JJ3936 0 15:00 0 16:00 CGH SDU
JJ3267 0 16:56 0 18:30 SDU CWB
JJ3268 0 19:14 0 20:46 CWB GIG
JJ3077 0 22:02 0 23:11 GIG CGH

Rota[28 - 4]
JJ3370 0 06:52 0 10:27 CGH REC
JJ3077 0 15:15 0 19:15 REC GIG
*REPO* 0 19:35 0 20:43 GIG CNF
JJ3458 0 22:25 1 01:05 CNF BEL

Rota[29 - 10]
JJ3732 0 06:52 0 08:12 CGH SSA
JJ3733 0 08:54 0 12:09 SSA CGH
JJ3926 0 12:30 0 13:33 CGH SDU
JJ3935 0 14:45 0 15:44 SDU CGH
JJ3966 0 16:59 0 18:06 CGH SDU
JJ3966 0 18:48 0 19:15 SDU CFB
JJ3967 0 19:45 0 20:12 CFB SDU
*REPO* 0 20:32 0 21:45 SDU CNF
JJ3226 0 22:20 0 22:59 CNF SSA
JJ3226 0 23:35 1 01:05 SSA NAT

Rota[30 - 7]
JJ3022 0 07:06 0 08:56 SDU BSB
JJ3025 0 09:52 0 11:37 BSB SDU
JJ3265 0 12:19 0 13:45 SDU CWB
JJ3264 0 14:30 0 15:53 CWB SDU
JJ3941 0 16:15 0 17:11 SDU CGH
JJ3952 0 19:00 0 20:06 CGH SDU
JJ3959 0 20:45 0 21:37 SDU CGH

Rota[31 - 6]
JJ3411 0 08:15 0 09:22 VIX GIG
JJ3411 0 10:02 0 12:05 GIG POA
JJ3410 0 12:47 0 14:41 POA GIG
JJ3410 0 15:22 0 16:20 GIG VIX
JJ3137 0 17:00 0 18:28 VIX CGH
JJ3226 0 20:19 1 01:05 CGH NAT

Rota[32 - 5]
JJ3745 0 10:45 0 14:56 CGB CGH
JJ3938 0 15:30 0 16:32 CGH SDU
JJ3945 0 17:15 0 18:08 SDU CGH
JJ3276 0 19:24 0 20:22 CGH RAO
JJ3275 0 21:02 0 22:04 RAO CGH

Rota[33 - 1]
JJ3871 0 13:15 0 14:15 BEL MAB

Rota[34 - 3]
JJ3563 0 13:10 0 18:43 RBR BSB
JJ3563 0 19:10 0 21:05 BSB GRU
JJ3177 0 22:00 0 23:35 GRU FLN