UNIVERSIDADE ESTADUAL DE CAMPINAS

Faculdade de Ciências Aplicadas

LUIZ MICHEL ARAM MAIOLO

Algoritmo genético multiobjetivo com codificação em ponto flutuante aplicado a problemas de finanças e *scheduling*

Limeira

2019

Luiz Michel Aram Maiolo

Algoritmo genético multiobjetivo com codificação em ponto flutuante aplicado a problemas de finanças e *scheduling*

Dissertação apresentada à Faculdade de Ciências Aplicadas da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia de Produção e de Manufatura, na área de Pesquisa Operacional e Gestão de Processos.

Orientadora: Prof[a]. Dr[a]. Priscila Cristina Berbert Rampazzo

Co-orientador: Prof. Dr. Cristiano Torezzan

Este exemplar corresponde à versão final da Dissertação defendida pelo aluno Luiz Michel Aram Maiolo e orientada pela Prof[a]. Dr[a]. Priscila Cristina Berbert Rampazzo.

Limeira

2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Faculdade de Ciências Aplicadas
Renata Eleuterio da Silva - CRB 8/9281

Maiolo, Luiz Michel Aram, 1990-
M285a Algoritmo genético multiobjetivo com codificação em ponto flutuante aplicado a problemas de finanças e scheduling / Luiz Michel Aram Maiolo. – Limeira, SP : [s.n.], 2019.

Orientador: Priscila Cristina Berbert Rampazzo.
Coorientador: Cristiano Torezzan.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Ciências Aplicadas.

1. Otimização multiobjetivo. 2. Algoritmos genéticos. 3. Finanças. 4. Agenda de execução (Administração). I. Rampazzo, Priscila Cristina Berbert, 1984-. II. Torezzan, Cristiano, 1976-. III. Universidade Estadual de Campinas. Faculdade de Ciências Aplicadas. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Floating-point multiobjective genetic algorithm applied to finance and scheduling problems
Palavras-chave em inglês:
Multiobjective optimization
Genetic algorithm
Finance
Production scheduling
Área de concentração: Pesquisa Operacional e Gestão de Processos
Titulação: Mestre em Engenharia de Produção e de Manufatura
Banca examinadora:
Priscila Cristina Berbert Rampazzo [Orientador]
Akebo Yamakami
Washington Alves de Oliveira
Data de defesa: 31-07-2019
Programa de Pós-Graduação: Engenharia de Produção e de Manufatura

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-0219-1731
- Currículo Lattes do autor: http://lattes.cnpq.br/9116236142094918

BANCA EXAMINADORA – DISSERTAÇÃO DE MESTRADO

Candidato: Luiz Michel Aram Maiolo RA: 117839

Data da defesa: 31 de julho de 2019

Título da dissertação: Algoritmo genético multiobjetivo com codificação em ponto flutuante aplicado a problemas de finanças e scheduling.

Profª. Drª. Priscila Cristina Berbert Rampazzo (Presidente, FCA/UNICAMP)

Prof. Dr. Akebo Yamakami (FEEC/UNICAMP)

Prof. Dr. Washington Alves de Oliveira (FCA/UNICAMP)

A ata da defesa, com as respectivas assinaturas dos membros da Banca Examinadora, encontra-se no processo de vida acadêmica do aluno.

À minha esposa Mariana.

À minha mãe Mara.

Agradecimentos

Agradeço,

À minha mãe Mara, que sempre me incentivou e apoiou, pelo amor e carinho.

À minha esposa Mariana, que esteve ao meu lado em todos os momentos desta jornada, pelo amor, carinho e dedicação.

À minha orientadora Priscila, pela ajuda nos momentos cruciais deste trabalho.

Ao meu coorientador Cristiano, por ter contribuído com meu ingresso na vida acadêmica.

Resumo

Grande parte dos problemas encontrados no ramo da otimização apresentam vários objetivos que devem ser satisfeitos ao mesmo tempo. Em muitos casos, tais objetivos são conflitantes, sendo que a otimização de determinado objetivo pode implicar na degradação de outro. Sendo assim, tais problemas não possuem apenas uma solução ótima, mas sim um conjunto delas. Este trabalho discute vários métodos que podem ser utilizados para obtenção de soluções para estes problemas, bem como propõe um Algoritmo Genético facilmente adaptável e baseado em alguns aspectos do NSGA-II, proposto por Deb et al. (2000). O Algoritmo Genético proposto foi aplicado em dois problemas importantes de áreas e características bastante distintas, sendo um deles a alocação de capital em instituições do setor financeiro e o outro a alocação de aeronaves em portões de embarque, e mostrou-se ser abrangente e facilmente adaptável, sendo necessário apenas mudanças nas codificações dos indivíduos para que o mesmo resolvesse os problemas de forma bastante satisfatória tanto em qualidade das soluções como em tempo de execução.

Palavras-chave: Otimização multiobjetivo. Algoritmos Genéticos. Finanças. Agenda de execução.

Abstract

Most of the problems encountered in the field of optimization present several goals that must be satisfied at the same time. In many cases, such objectives are conflicting, and the optimization of one objective may result in the degradation of another. Thus, such problems have not only an optimal solution but a set of them. This work approaches several methods of existing solutions to these problems, as well as propose a Genetic Algorithm easily adaptable and based on some aspects of NSGA-II, proposed by Deb et al. (2000). The proposed Genetic Algorithm was applied in two important problems of quite distinct areas and characteristics, one of them being the allocation of capital in financial institutions and the other the allocation of aircraft at boarding gates, and proved to be comprehensive and easily adaptable, necessitating only of changes in the codifications of the individuals so that the mentioned problems were solved in a very satisfactory way in both the quality of the solutions and at the time of execution.

Keywords: Multiobjective optimization. Genetic Algorithms. Finance. Scheduling.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PO	Pesquisa Operacional
NSGA	*Non-dominated Sorting Genetic Algorithm*
NSGA-II	*Nondominated Sorting Genetic Algorithm II*
P^*	Conjunto Pareto-ótimo
FP^*	Fronteira de Pareto
P_g	População inicial do NSGA-II
Q_g	População de filhos do NSGA-II
R_g	União de P_g e Q_g
TP	Tamanho da população de pais do NSGA-II
PC	Probabilidade de cruzamento (ou crossover) de um indivíduo em um Algoritmo Genético
PM	Probabilidade de mutação de um indivíduo em um Algoritmo Genético
MPT	*Modern Portfolio Theory* ou Teoria Moderna de Portfólios
CEA	Capital Econômico Alocado
CMN	Conselho Monetário Nacional
EVA	*Economic Value Added* ou Valor Econômico Adicionado
RAAROC	*Risk adjusted return on capital* ou Retorno Ajustado ao Risco no Capital
VaR	*Value at Risk* ou Valor em Risco
GAP	*Gate Assessment Problem* ou Problema de alocação de aeronaves em portões de embarque
G	Quantidade de gerações de um Algoritmo Genético
FSND	Fronteira de soluções não dominadas

Lista de símbolos

$\boldsymbol{\Omega}$	Conjunto de restrições de um problema
$\boldsymbol{x}$	Vetor de variáveis de decisão de problema
$\boldsymbol{F}(\boldsymbol{x})$	Função vetorial em que cada elemento representa um objetivo de um problema
$\Re$	Conjunto dos números reais
$\mathbb{Z}$	Conjunto dos números inteiros
Λ	Região factível correspondente ao espaço das funções objetivo
ε	Número positivo tão pequeno quanto se queira

Sumário

Introdução

Contextualização

A tomada de decisões é uma tarefa inerente da gestão, independentemente do nível (estratégico, gerencial, operacional), e, em muitos casos, não é uma questão simples. As escolhas devem ser feitas com base em alternativas de soluções viáveis ao problema em análise. Com o intuito de proporcionar maiores opções de soluções viáveis e facilitar a tomada de decisão, uma série de técnicas e estudos desenvolvidos durante a Segunda Guerra Mundial deu origem à Pesquisa Operacional (PO) (MARINS, 2011).

Os problemas em que a Pesquisa Operacional auxilia na resolução estão espalhados em praticamente todas as áreas onde se faz necessária a tomada de alguma decisão. Inúmeras empresas e indústrias enfrentam problemas em relação ao Sequenciamento de Tarefas, que podem ser causados pela alocação indevida de recursos e processos mal definidos. Pode-se otimizar os processos realizando o planejamento do sequenciamento de tarefas, o que resulta na melhoria do controle do fluxo de produção, cumprindo os prazos de entrega e programando as tarefas de maneira a melhor utilizar os recursos disponíveis. Além disto, outro problema que é beneficiado pela adoção de técnicas de PO no auxílio de tomadas de decisão é o problema de Seleção de Portfólios, ou alocação de capital, o qual consiste em determinar de que maneira um investidor deve aplicar um capital em um determinado conjunto de opções de investimentos: ativos ou produtos. Para aplicar determinado capital numa carteira de ações, por exemplo, inicialmente seleciona-se as ações que comporão a carteira e, em sequência, determina-se a quantidade (ou a porcentagem) do capital a ser investido em cada ativo selecionado. Para este fim, usualmente considera-se o retorno esperado de cada ativo, bem como o risco atrelado ao mesmo.

Para que técnicas de PO sejam eficientemente aplicadas, usualmente se faz necessário alguns procedimentos padrões que visam delimitar o problema abordado. Em geral, tais procedimentos são a identificação do problema, a formulação dos objetivos que se deseja atingir, a identificação das restrições e parâmetros, bem como dos elementos (variáveis) que podem ser alterados de modo que os objetivos sejam atingidos (MARINS, 2011).

Uma vez que grande parte dos problemas enfrentados pelos decisores são complexos e possuem vários objetivos que devem ser simultaneamente considerados, os processos de otimização evoluíram de técnicas cuja abordagem resolvia problemas com apenas um objetivo, para técnicas de otimização multiobjetivo, tornando a literatura mais próxima da prática. Dentre estes problemas, há aqueles que, além de apresentarem

diversos objetivos, possuem objetivos que são conflitantes, ou seja, a solução ótima de um objetivo não é a solução ótima de outro e, comumente, uma solução que melhora um objetivo, piora outro, e vice-versa. Por exemplo, a minimização do custo de fabricação de um objeto e a minimização de sua rugosidade superficial, que é normalmente atingida através de um processo de usinagem, são dois objetivos que não apresentam uma única solução ótima (DEB; DATTA, 2011). Neste sentido, uma das principais diferenças entre os problemas de otimização multiobjetivo e aqueles com somente um objetivo é que, enquanto o segundo, normalmente, apresenta uma única solução, o primeiro possui um conjunto de soluções aceitáveis que, quando não dominadas, são melhores do que as demais soluções existentes. Este conjunto de soluções não dominadas pode ser um conjunto de soluções Pareto-ótimas, sendo que as soluções do conjunto Pareto-ótima caracterizam-se pelo *trade-off* que apresentam em relação aos objetivos do problema, onde a melhora em um objetivo, acarreta piora em outro. Tal conjunto provavelmente incluirá soluções ótimas individuais e muitas outras soluções que são ótimas com relação a certas compensações entre outros objetivos (DEB; DATTA, 2011). Após a obtenção da fronteira de soluções Pareto-ótimas, cabe ao decisor escolher uma solução que atenda suas opções preferidas a partir do *trade-off* envolvido.

Existem inúmeras técnicas voltadas à resolução de problemas de otimização multiobjetivo. Tais técnicas contemplam as chamadas tradicionais (programação matemática) e as denominadas alternativas (heurísticas e metaheurísticas).

Alguns dos métodos de programação matemática, usualmente necessitam que sejam feitas atribuições de pesos e/ou classificações de preferência, de modo que o problema multiobjetivo torna-se um problema com um único objetivo (STEUER, 1986). A resolução de problemas por tais métodos, apesar de encontrar uma solução ótima, não garante que esta seja a melhor solução para o problema original, pois, para encontrá-la, são necessários vários ajustes no problema original, o que pode, em alguns casos, alterar significativamente as características dos objetivos iniciais do problema. Além disso, a cada execução, tais métodos, usualmente, apresentam apenas um ponto da Fronteira de Pareto, fato que exige inúmeras execuções para montar uma única fronteira de soluções não dominadas. Dentre tais métodos, alguns dos mais comumente utilizados são programação por metas, soma ponderada e ε-restrito.

Embora nem sempre encontrem solução ótima, metaheurísticas são uma ótima opção para resolução de problemas com múltiplos objetivos, pois, quando bem formuladas, tais técnicas apresentam soluções de alta qualidade e com complexidade computacional relativamente baixa quando comparadas com técnicas de programação matemática. Dentre as metaheurísticas, os Algoritmos Genéticos destacam-se em função de sua grande adaptabilidade e do compromisso entre o tempo e a qualidade das soluções que entregam. Algoritmos Genéticos, além de tratarem todos os objetivos do problema explicita e simul-

taneamente, sem a necessidade de atribuição de pesos, por exemplo, são capazes de formar uma fronteira de soluções não dominadas em uma única execução, o que os torna uma ferramenta útil na busca por soluções em problemas com múltiplos objetivos.

Objetivo e problemas para estudo de caso

Em função da grande variedade e importância dos problemas de Otimização Multiobjetivo, este trabalho visa discutir alguns métodos de soluções capazes de gerar soluções satisfatórias aos problemas desta classe, de modo que os decisores possam estar munidos de informações que os auxilie durante o processo de tomada de decisão.

Além de discutir alguns dos mais importantes métodos presentes em literatura, será implementado um Algoritmo Genético baseado em alguns aspectos do *Nondominated Sorting Genetic Algorithm II* (NSGA-II), proposto por Deb et al. (2000), uma vez que este último obteve sucesso no tratamento de diversos tipos de problemas relatados na literatura. O Algoritmo Genético proposto neste trabalho utiliza-se das funções de *ranking* (classificação por não dominância) e *crowding distance* (distância de aglomeração) presentes no NSGA-II. Assim como o NSGA-II, o algoritmo aqui apresentado considera explicita e simultaneamente diferentes funções objetivo, encontrando um conjunto de soluções não dominadas no final de uma execução.

O algoritmo proposto neste trabalho será aplicado em dois problemas importantes de áreas distintas, de modo que se possa verificar sua adaptabilidade e capacidade de encontrar soluções com bom compromisso de qualidade e tempo computacional.

O primeiro problema onde o algoritmo será aplicado consiste na alocação de determinada quantidade de capital disponível em diversas operações de crédito. Tais operações estão sujeitas a riscos e retornos distintos e possuem limite máximo de alocação em função da disponibilidade de clientes que tomarão tais créditos (demanda). Trata-se de um problema de programação linear e seus objetivos consistem em minimizar a quantidade de capital alocado e em maximizar o lucro total proveniente destas alocações. Sendo a solução para o problema uma fração (ou percentual) do capital total disponível e o algoritmo genético proposto possuindo codificação em ponto flutuante, a aplicação do método na busca pela solução do problema é intuitiva.

O segundo problema onde o algoritmo será aplicado consiste na alocação de aeronaves em portões de embarque (*gates*). Tal problema, na forma proposta neste trabalho, possui três objetivos conflitantes, os quais são a minimização da soma de atrasos nas decolagens de aeronaves, a minimização das distâncias percorridas pelos passageiros entre as operações de *check-in* e embarque e entre as operações de desembarque e coleta de bagagem, bem como a minimização de aeronaves destinadas ao *apron* (garagem ou *gate* distante) – muitas vezes, por falta de vagas em *gates*, aeronaves são destinadas ao *apron*,

o que gera um problema às companhias aéreas, pois, além de terem de arcar com custos adicionais (disponibilização de transporte para os passageiros – ônibus, na maior parte dos casos), sofrem danos de reputação por não prestarem um serviço considerado adequado e ainda podem ter impactos nos planejamentos iniciais, o que acarreta, possivelmente, em maiores custos operacionais. Tal problema apresenta variáveis de decisão inteiras e é um problema de *scheduling*, que é da classe NP-difícil (GAREY; JOHNSON; SETHI, 1976). A aplicação do Algoritmo Genético neste segundo problema visa mostrar sua grande adaptabilidade, pois apesar de ser codificado em ponto flutuante, será utilizado para resolver um problema com variáveis inteiras. Além disto, ao aplicar o algoritmo no problema de alocação de aeronaves em portões de embarque, busca-se evidenciar que o primeiro é capaz de entregar uma solução com bom compromisso entre qualidade e tempo computacional, uma vez que espera-se obter soluções satisfatórias em razoável tempo computacional na resolução de um problema NP-difícil.

Organização da Dissertação

Este trabalho está organizado da seguinte maneira:

- Contextualização dos problemas de Otimização Multiobjetivo, bem como dos problemas de estudo de caso.
- Capítulo 1: apresentação do Referencial Metodológico, com alguns dos principais métodos, técnicas de resolução e algoritmos propostos na literatura relacionados à Otimização Multiobjetivo.
- Capítulo 2: revisão literária acerca de problemas de alocação de portfólio e revisão literária em relação aos problemas de agendamento (*scheduling*) e suas aplicações no setor aéreo, mais especificamente na alocação de aeronaves em portões de embarque, bem como as formulações matemáticas dos problemas explorados nesta dissertação.
- Capítulo 3: apresentação do Algoritmo Genético proposto e suas adaptações de implementação para resolução dos problemas discutidos.
- Capítulo 4: comparações e discussões entre os resultados obtidos na resolução dos problemas.
- Capítulo 5: apresentação de conclusões.
- Perspectivas futuras de estudos complementares.

1 Otimização Multiobjetivo

Grande parte dos problemas encontrados no ramo da otimização apresentam vários objetivos que devem ser satisfeitos ao mesmo tempo. Na maioria destes casos, tais objetivos são conflitantes, ou seja, atingir determinado objetivo implica na piora de outro.

Problemas envolvendo múltiplos objetivos estão presentes em diversas áreas, sendo, alocação de capital, definição de rotas de transporte, gerenciamento de produção industrial, alocação de tarefas a recursos, seleção de fornecedores, apenas alguns exemplos.

Uma vez que em um problema de otimização multiobjetivo pode haver infinitas soluções não comparáveis entre si, após a obtenção de um conjunto ótimo de soluções, cabe a algum analista (decisor) a tomada de decisão em relação a qual solução aplicar ao problema, sendo que, para tanto, o mesmo deve considerar os objetivos gerais do problema.

Um exemplo de problema real de otimização multiobjetivo, presente no dia a dia das instituições financeiras, consiste na necessidade de maximizar a concessão de crédito e, ao mesmo tempo, minimizar a inadimplência em relação aos empréstimos. Uma vez que esta é uma das principais atividades de empresas do setor financeiro, gerir adequadamente este problema é crucial para o bom desempenho delas.

1.1 Conceitos Básicos

Otimização multiobjetivo define-se pelo problema de encontrar um vetor de variáveis de decisão $\boldsymbol{x}$ que satisfaça $\boldsymbol{\Omega}$ e otimize[1] a função vetorial $\boldsymbol{F}(\boldsymbol{x})$, cujos elementos representam as funções objetivos (OSYCZKA, 1985).

$$\text{minimizar} \qquad \boldsymbol{F}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})] \quad m \geqslant 2 \tag{1.1}$$

$$\text{s.a.:} \qquad \boldsymbol{x} \in \boldsymbol{\Omega} \tag{1.2}$$

Onde, $\boldsymbol{\Omega} = \{\boldsymbol{x} \in \Re^n \quad | \quad \boldsymbol{G}(\boldsymbol{x}) \leqslant 0, \quad \boldsymbol{H}(\boldsymbol{x}) = 0, \quad \boldsymbol{x_l} \leqslant \boldsymbol{x} \leqslant \boldsymbol{x_u}\}$.

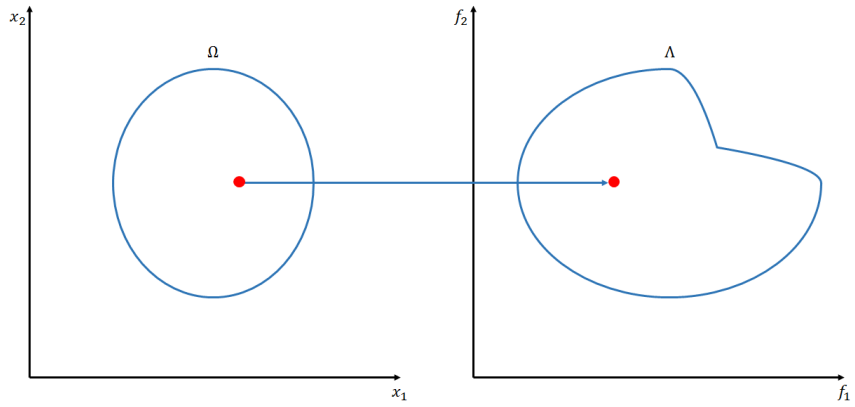
A região factível, correspondente ao espaço das funções objetivo, ou seja, a região que aloca todas as soluções que resolvem o problema e respeitam as restrições, é definida da seguinte forma:

$$\Lambda = \{\boldsymbol{k} \in \Re^m \quad | \quad \boldsymbol{k} = \boldsymbol{F}(\boldsymbol{x})_{\boldsymbol{x} \in \boldsymbol{\Omega}}\} \tag{1.3}$$

[1] São apresentadas definições para o problema de minimização; definições para o problema de maximização são análogas

A Figura 1 mostra um um exemplo de mapeamento do espaço das soluções no espaço das funções objetivo para o caso bidimensional.

Figura 1 – Mapeamento do espaço das soluções no espaço das funções objetivo.



Fonte: produzido pelo autor.

Uma vez que $\boldsymbol{F}(\boldsymbol{x})$ é um vetor, se quaisquer componentes deste último competirem, não existirá uma solução única para o problema e sim um conjunto de soluções, onde os *trade-offs* devem ser levados em consideração no momento de tomada de decisão. Em função disto, deve-se utilizar o conceito de otimalidade de Pareto para a caracterização e obtenção das soluções.

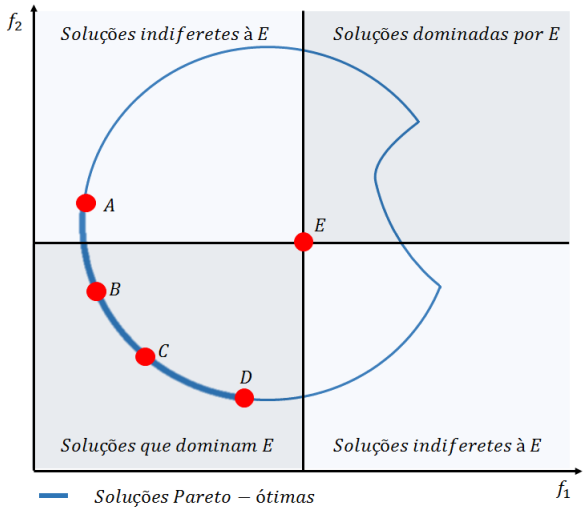
Uma solução é dita Pareto-ótima (não dominada, não inferior ou eficiente) quando a melhora em um de seus objetivos acarreta em decréscimo em outro (FERREIRA, 1999).

A Figura 2 mostra um exemplo de soluções Pareto-ótimas entre os pontos A e D, em um caso bidimensional. Todos os pontos contidos no segmento $\overline{AD}$ são soluções Pareto-ótimas, sendo que cada solução é melhor em um objetivo do que em outro, porém, são equivalentes em termos gerais do problema. A solução do ponto, B, por exemplo, é melhor do que a solução do ponto C para o objetivo f_2, porém, esta última solução é melhor do que a primeira em relação ao objetivo f_1. Além disso, na Figura 2, é possível observar as regiões de dominância em relação à uma solução arbitrária E. Em otimização multiobjetivo, uma determinada solução pode apresentar três diferentes tipos de relação entre seus vizinhos: dominar, ser dominada ou indiferença (nem domina e nem é dominada).

Na sequência serão abordadas algumas definições[2] importantes referentes às soluções Pareto-ótimas. Tais definições foram baseadas no conteúdo apresentado em Deb (1999).

[2] São apresentadas definições para o problema de minimização; definições para o problema de maximização são análogas

Figura 2 – Fronteira de Pareto-ótimo e relações de dominância de soluções.



Fonte: produzido pelo autor.

Dominância de Pareto: sejam $\boldsymbol{a}$ e $\boldsymbol{b}$ pertencentes a $\boldsymbol{\Omega}$, então $\boldsymbol{a}$ domina $\boldsymbol{b}$, ou seja, $\mathbf{F}(\boldsymbol{a}) \preceq \mathbf{F}(\boldsymbol{b})$, se e somente se:

$$\forall \quad i \in \{1, 2, ..., m\}, \quad f_i(\boldsymbol{a}) \leqslant f_i(\boldsymbol{b}) \quad \text{e} \quad \exists \quad j \in \{1, 2, ..., m\}, \quad f_j(\boldsymbol{a}) < f_j(\boldsymbol{b}) \tag{1.4}$$

Ou seja, para que $\boldsymbol{a}$ domine $\boldsymbol{b}$, $\boldsymbol{a}$ não pode ser pior do que $\boldsymbol{b}$ em nenhum dos objetivos e deve ser melhor do que $\boldsymbol{b}$ em pelo menos um deles. Caso $\boldsymbol{a}$ não domine $\boldsymbol{b}$ e $\boldsymbol{b}$ também não domine $\boldsymbol{a}$, $\boldsymbol{a}$ e $\boldsymbol{b}$ são soluções equivalentes.

Otimalidade de Pareto: uma solução $\boldsymbol{x}^*$ é Pareto-ótimo caso não exista uma outra solução $\boldsymbol{x}$ que possa melhorar algum objetivo, sem acarretar na piora de pelo menos mais um objetivo. Ou seja, nenhum outro vetor do espaço de busca $\boldsymbol{\Omega}$ domina $\boldsymbol{x}^*$.

Conjunto Pareto-ótimo: um conjunto Pareto-ótimo, P^*, é definido pela existência de determinado vetor $\boldsymbol{x}$ que atenda as restrições do problema, de modo que não exista um vetor $\boldsymbol{y}$ contido em $\boldsymbol{\Omega}$, em que $\boldsymbol{F}(\boldsymbol{y})$ preceda ou seja igual a $\boldsymbol{F}(\boldsymbol{x})$. Matematicamente, seria:

$$P^* = \{\boldsymbol{x} \in \boldsymbol{\Omega} \quad | \quad \nexists \quad \boldsymbol{y} \in \boldsymbol{\Omega}, \quad \boldsymbol{F}(\boldsymbol{y}) \preceq \boldsymbol{F}(\boldsymbol{x})\} \tag{1.5}$$

Fronteira de Pareto: uma fronteira de Pareto, FP^*, é definida pela função

vetorial $\boldsymbol{F}(\boldsymbol{x})$ formada pelos vetores $\boldsymbol{x}$ pertencentes ao conjunto de Pareto-ótimo, ou seja:

$$FP^* = \{\boldsymbol{F}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})] \quad | \quad \boldsymbol{x} \in P^*\} \tag{1.6}$$

Conjunto Pareto-ótimo local: é definido pela não existência, para todo $\boldsymbol{x}$, pertencente a determinado conjunto $\boldsymbol{P}$, de uma solução $\boldsymbol{y}$, $||\boldsymbol{x} - \boldsymbol{y}||_\infty \leqslant \varepsilon$, que domine qualquer solução contida em $\boldsymbol{P}$. Satisfeita tal condição, todas as soluções contidas em tal conjunto $\boldsymbol{P}$ perfazem um conjunto Pareto-ótimo local. Sendo ε um número positivo tão pequeno quanto se queira e $\boldsymbol{y}$ um vetor resultante de uma perturbação nas proximidades de $\boldsymbol{x}$.

Conjunto Pareto-ótimo global: caso não haja qualquer solução no espaço de busca que domine qualquer membro de $\boldsymbol{P}$, as soluções contidas em $\boldsymbol{P}$ foram o Conjunto Pareto-ótimo global.

1.2 Métodos de Otimização Multiobjetivo

Existem inúmeras técnicas voltadas à resolução de problemas de otimização multiobjetivo. Tais técnicas contemplam as chamadas tradicionais (programação matemática) e as denominadas alternativas (heurísticas e metaheurísticas).

Além da busca por soluções, outra vertente presente nos problemas de vários objetivos é a tomada de decisão (HORN, 1997). Ou seja, além da busca por soluções Pareto-ótimas, há também o problema de definir qual das soluções ótimas deverá ser utilizada e, para tanto, faz-se necessária a presença de um analista.

De acordo com a interação entre o analista responsável pela decisão e o método de otimização proposto, as técnicas de otimização multiobjetivo podem ser classificadas em:

- A-priori ou tomada de decisão antes da busca;
- A-posteriori ou tomada de decisão depois da busca;
- Interativa ou tomada de decisão durante a busca.

A seguir, tais métodos serão brevemente discutidos.

Métodos a-priori

Métodos a-priori exigem que o analista defina suas preferências antes da resolução do problema. As preferências do analista podem ser utilizadas, basicamente, de

duas formas durante a resolução do problema: 1) classificação por ordem de preferências e, 2) ponderação de objetivos.

Na classificação ordinal de preferências, o analista define uma ordem de preferência pelos objetivos. Dada esta definição, resolve-se o problema otimizando um objetivo por vez, do primeiro ao último, respeitando a ordem de preferência do analista. Todos os objetivos devem respeitar a solução encontrada pelos objetivos que o precedem.

A ponderação de objetivos consiste em atribuir pesos para cada objetivo do problema e, então, combiná-los em um único. Através disto, pode-se utilizar técnicas de otimização tradicionais de problemas mono-objetivo.

Embora diminuam a complexidade computacional, tais métodos necessitam que o analista conheça previamente possíveis características da solução do problema.

Métodos a-posteriori

O analista não interfere diretamente na solução como acontece no caso do método a-priori. Aqui, o analista deve realizar a tomada de decisão após ter sido concluída a otimização do problema.

Apesar de fornecer todas (ou quase todas) as possibilidades de solução ótima, métodos a-posteriori apresentam algumas dificuldades computacionais.

Métodos interativos

Nos métodos interativos, a interferência do analista ocorre durante o processo de busca pelas soluções Pareto-ótimas. Neste tipo de método, informações sobre preferências são indicadas pelo analista durante o processo de otimização, norteando a busca na direção de regiões onde existem soluções relevantes. O analista é capaz de ajustar as preferências após um determinado número de iterações, ao passo que detecta novas oportunidades. Tais fatores tornam os métodos interativos mais elaborados do que os apresentados anteriormente.

Mesmo dispensando o conhecimento a-priori do problema e permitindo ao analista aprender sobre o problema ao passo em que ocorre a busca das soluções Pareto-ótimas, métodos interativos podem levar a maus resultados, a depender das preferências indicadas pelo analista.

1.2.1 Métodos de programação matemática

Alguns dos métodos de programação matemática, usualmente necessitam que sejam feitas atribuições de pesos e/ou classificações de preferência, de modo que o problema multiobjetivo torna-se um problema com um único objetivo (STEUER, 1986). A resolução

de problemas por tais métodos, apesar de encontrar uma solução ótima, não garante que esta seja a melhor solução para o problema original, pois, para encontrá-la, são necessários vários ajustes no problema original, o que pode, em alguns casos, alterar significativamente as características dos objetivos iniciais do problema. Além disso, a cada execução, tais métodos, usualmente, apresentam apenas um ponto da Fronteira de Pareto, fato que exige inúmeras execuções para montar uma única fronteira de soluções não dominadas. Dentre tais métodos, alguns dos mais comumente utilizados são programação por metas, soma ponderada e ε-restrito.

A seguir serão abordadas as principais características destes métodos.

Método de programação por metas

Existem várias maneiras de abordar problemas multiobjetivo através do método de programação por metas, entretanto, o mais comum deles é aquele que associa pesos $\boldsymbol{w} = [w_1, w_2, ..., w_m]$ e metas $\boldsymbol{F}^* = [f_1^*, f_2^*, ..., f_m^*]$ para cada objetivo $\boldsymbol{F}(\boldsymbol{X}) = [f_1\boldsymbol{x}, f_2\boldsymbol{x}, ..., f_m\boldsymbol{x}]$. Neste método, os vários objetivos são convertidos em um único, que consiste em minimizar o desvio dos objetivos originais em relação às metas estabelecidas para os mesmos (ARENALES et al., 2007).

Tal técnica permite uma ampliação do espaço de soluções factíveis, permitindo que os objetivos se encontrem acima ou abaixo das metas estabelecidas.

Matematicamente, o método pode ser representado da seguinte maneira:

$$\text{minimizar} \qquad \varphi \tag{1.7}$$

$$\text{sujeita a:} \quad f_i(\boldsymbol{x}) - w_i\varphi \leqslant f_i^* \tag{1.8}$$

$$\boldsymbol{x} \in \boldsymbol{\Omega} \tag{1.9}$$

$$\boldsymbol{\varphi} \in \Re \tag{1.10}$$

Método das ponderações

A resolução de problemas multiobjetivo através do método das ponderações (ou método da soma ponderada) consiste em transformar os vários objetivos do problema em apenas um através da atribuição de pesos para cada objetivo. É importante que a atribuição de pesos seja normalizada, de modo a manter uma escala única para os objetivos, facilitando a distinção entre a importância de cada objetivo.

A obtenção da solução passa pela variação do vetor de pesos $\boldsymbol{w} \in$ W= $\{\boldsymbol{w} : w \in \Re^m, \quad w_i \geqslant 0 \quad \text{e} \quad \sum_{i=1}^{m} w_i = 1\}$. Ou seja, deve-se gerar várias iterações com valores diferentes de pesos para cada objetivo, de modo que se possa obter soluções Pareto-ótimas.

Matematicamente, a otimização consiste em:

$$\text{minimizar} \quad \sum_{i=1}^{m} w_i f_i(\boldsymbol{x}) \tag{1.11}$$

$$\text{sujeita a:} \quad \boldsymbol{x} \in \boldsymbol{\Omega} \tag{1.12}$$

O presente método não é capaz de gerar todas as soluções Pareto-ótimas quando o espaço Λ não é convexo, uma vez que, em função dos pesos do vetor $\boldsymbol{w}$, o mesmo gera diferentes retas suportes a cada iteração e nem todos os pontos de Pareto-ótimo admitem reta suporte (ARROYO, 2002).

Método ε-restrito

Tal método consiste na otimização de um objetivo do problema, transformando os demais em restrições de desigualdade. Deseja-se resolver problemas $P(\boldsymbol{\varepsilon})$, onde $\boldsymbol{\varepsilon}$= $[\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m]$. Matematicamente, tem-se:

$$\text{minimizar} \quad f_i(\boldsymbol{x}) \tag{1.13}$$

$$\text{sujeita a:} \quad f_j \leqslant \varepsilon_j \quad \forall j \neq i \tag{1.14}$$

$$\boldsymbol{x} \in \boldsymbol{\Omega} \tag{1.15}$$

A busca por soluções Pareto-ótimas consiste em variar, de maneira adequada, os limites de ε_i. Uma dificuldade do método consiste na possibilidade de não encontrar soluções factíveis caso a variação dos limitantes de ε_i não seja adequada.

Ao contrário do método da soma ponderada, é possível encontrar soluções factíveis mesmo em Λ não convexo.

Simplex multiobjetivo

Ao contrário dos métodos de programação matemática anteriores, através do Simplex Multiobjetivo é possível tratar todos os objetivos do problema ao mesmo tempo, não sendo necessária a classificação ou atribuição de pesos aos mesmos.

O fato de a região factível apresentar uma quantidade finita de pontos extremos (fato garantido por tal região ser um poliedro), permite ao presente método "caminhar" de um ponto extremo Pareto-ótimo para outro ponto do mesmo tipo até que o conjunto de Pareto seja completamente formado. O conjunto de Pareto é formado tanto pelos pontos extremos de Pareto-ótimo quanto pelas faces Pareto-ótimas.

O método simplex multiobjetivo é limitado à resolução de problemas lineares

do seguinte tipo:

$$\text{minimizar} \quad \boldsymbol{C}\boldsymbol{x} \tag{1.16}$$

$$\text{sujeita a:} \quad \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b} \tag{1.17}$$

$$\boldsymbol{x} \geqslant 0 \tag{1.18}$$

Onde:

- $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_m]^t \in \Re^{(m \times n)}$
- $\boldsymbol{c} \in \Re^{(1 \times n)}$
- $\boldsymbol{x} \in \Re^n$
- $\boldsymbol{A} \in \Re^{(l \times n)}$
- $\boldsymbol{b} \in \Re^l$

1.2.2 Métodos heurísticos e metaheurísticos

A escolha de determinado método a ser utilizado para resolver um dado problema de otimização está diretamente relacionada com a qualidade da solução desejada e custo computacional relacionado à obtenção da mesma. Uma vez que, em problemas de aplicação real, a solução desejada comumente precisa ser, senão ótima, de alta qualidade, não é possível desenvolver, para a maioria destes problemas, algum algoritmo exato que possa ser executado com razoável complexidade computacional. Para tais problemas é necessário a utilização de estratégias heurísticas[3], as quais, quando bem formuladas e adaptadas aos problemas a que se propõem resolver, podem apresentar soluções de boa qualidade e com complexidade computacional relativamente baixa, quando comparadas com algoritmos exatos (ARROYO, 2002).

Uma estratégia heurística é caracterizada pela utilização de técnicas simplificadas empregadas em problemas considerados até então complexos (SILVER et al., 1980). Como exemplo, pode ser considerado um problema de estratégia de coleta de lixo de uma cidade qualquer. Neste problema, deseja-se coletar o lixo de toda a cidade da maneira mais rápida possível utilizando-se dos recursos disponíveis (caminhões de coleta). De maneira simplificada, define-se a estratégia heurística de que cada caminhão disponível atuará em determinado bairro da cidade.

Ao passo que estratégias heurísticas são empregadas na resolução de problemas específicos, procedimentos metaheurísticos, os quais não deixam de ser um tipo de heurística, por serem flexíveis e adaptáveis, são aplicados em ampla gama de problemas. O

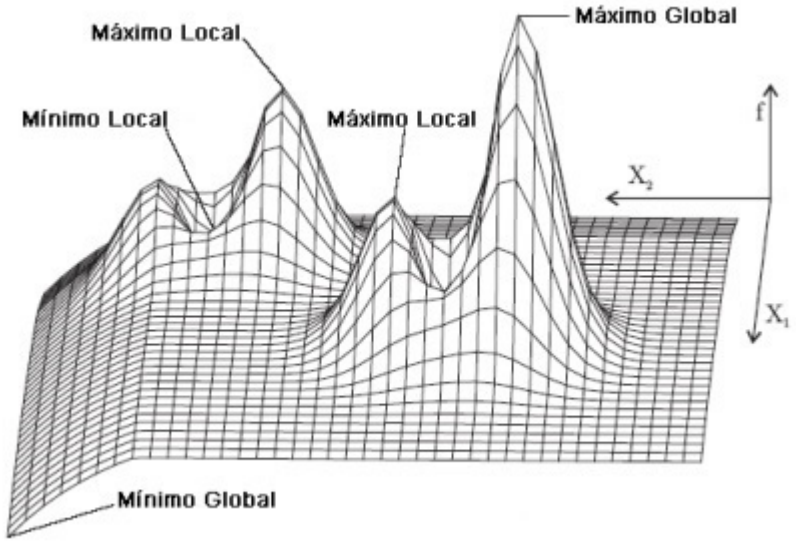
[3] O termo heurística advém da palavra grega *heuriskein*, cujo significado é encontrar ou descobrir; porém, em otimização, o significado mais comum associado a ela é "como buscar boas soluções" (GARCIA, 2005).

termo metaheurística teve sua primeira citação em Glover (1986), onde é definido como procedimento iterativo para resolver problemas de otimização combinatória que inclui heurísticas tradicionais como subprocedimentos. Por sua vez, rotinas heurísticas podem ser definidas como processos que buscam soluções para problemas, sem necessariamente encontrar a solução ótima (GREENBERG, 2017).

Por possuírem estruturas com componentes genéricos que se adaptam ao problema que se busca solucionar, procedimentos metaheurísticos são considerados métodos inteligentes e flexíveis na resolução de problemas complexos. As metaheurísticas permitem que sejam encontradas tanto soluções melhores quanto piores do que as já encontradas, sendo esta uma importante característica, pois potencializa a busca por soluções globalmente ótimas, superando a principal limitação das heurísticas tradicionais, as quais tendem a convergir para uma solução localmente ótima (ARROYO, 2002; GARCIA, 2005).

A Figura 3 mostra uma região de busca por soluções onde podem ser observados máximos e mínimos locais e globais para uma função bidimensional.

Figura 3 – Região de busca por soluções para uma função bidimensional.



Fonte: disponível em Silva (2011).

A busca por soluções no espaço mostrado na Figura 3 tende a ser mais eficiente através de metaheurística do que por técnicas exaustivas e aleatórias, pois, ao passo em que a busca exaustiva precisa checar todos os pontos do espaço solução e a busca aleatória verifica apenas alguns pontos deste espaço e salva o melhor resultado encontrado, metaheurísticas utilizam-se de uma solução aleatória aliada ao conhecimento de resultados anteriores, otimizando o tempo de busca e a qualidade da solução obtida (SILVA, 2011).

Há um grande conjunto de metaheurísticas descritas na literatura, sendo que

estas apresentam grande variedade quanto aos princípios e estratégias utilizados para a busca por soluções de problemas. Dentre as mais comuns, destacam-se as seguintes: Busca Tabu, *Simulated Annealing*, GRASP, além de métodos baseados em algoritmos genéticos.

A seguir encontra-se uma breve descrição dos métodos mencionados.

Busca Tabu

O método Busca Tabu teve início no fim dos anos 1960 e início dos anos 1970, porém, a forma como é utilizada atualmente foi proposta em 1986, por Glover (1986).

A metaheurística Busca Tabu procura encontrar uma solução através de uma heurística de busca local. O método armazena informações sobre soluções já visitadas, de modo a evitar o retorno a pontos de mínimo local, em busca de ultrapassar a otimalidade local e atingir o ótimo global ou algo próximo dele.

O método parte de uma solução inicial, aleatória ou definida por algum critério específico, movendo-se, a cada iteração, para a melhor solução possível na vizinhança. Esse movimento não retorna a pontos já visitados, pois os mesmos encontram-se armazenados em uma lista tabu. Esta lista, por sua vez, é mantida por um período de tempo ou iterações predefinidas. Deste modo, o método tende a encontrar uma solução ótima global ou algo próximo disso. A solução final tem pouca ou nenhuma dependência da solução inicial, pois o método é orientado a fugir de ótimos locais.

A associação da palavra tabu ao método metaheurístico descrito advém do fato de que tabus podem ser modificados ou adaptados ao longo do tempo, através de relações socais, assim com as listas tabus do método podem ser modificadas ao longo do processo de busca (GLOVER, 1986).

Simulated Annealing

Simulated Annealing (ou recozimento simulado) é uma técnica de busca local focada em encontrar soluções ótimas ou próximas disso. *Simulated Annealing* é baseada no processo termodinâmico de solidificação de metais conhecido como recozimento.

O processo de recozimento consiste num tratamento térmico no qual um material é exposto a elevada temperatura, podendo ou não ocorrer sua fusão, por longo período de tempo e, então é lentamente resfriado até a temperatura ambiente. O objetivo de tal processo consiste no alivio de tensões, aumento de maciez, ductilidade e tenacidade e/ou produção de uma microestrutura específica. Durante o processo de resfriamento, os átomos tendem a organizarem-se de maneira a minimizar a energia interna do sistema, em busca de atingir o estado mínimo de energia, o que reduz a quantidade de defeitos de grão e gera as características metalúrgicas desejadas (CALLISTER, 2007).

O algoritmo para a metaheurística de *Simulated Annealing* é baseado na sequência de resfriamento enfrentada pelo metal e é descrito da seguinte maneira: parte-se de um estado inicial i, onde a energia do sólido é descrita por E_i, ocorre a geração de um estado seguinte j através de um mecanismo de perturbação, cuja energia é denotada por E_j. O estado j é considerado o novo estado inicial caso possua menor energia associada do que o estado i. Caso o estado i possua energia maior ou igual ao estado j, o estado j é aceito com uma probabilidade determinada por:

$$e^{\frac{E_i - E_j}{k_b T}} \tag{1.19}$$

Onde T é equivalente a temperatura do material e k_b é a constante de Boltzmann[4] (GARCIA, 2005).

Considerando-se o critério de aceitação para que determinado estado se torne o estado inicial, é possível criar uma sequência de soluções para o problema em análise. Para isto, a função objetivo deve ser de minimização, análoga ao problema termodinâmico apresentado e com as seguintes condições (GARCIA, 2005):

- As soluções do problema de otimização devem ser equivalentes aos estados do sistema termodinâmico;
- O valor atribuído a uma solução deve ser equivalente à energia de um estado;
- As soluções vizinhas, cuja obtenção se derem pelo mecanismo de perturbação, devem corresponder aos estados subsequentes.

GRASP

GRASP é a sigla para *Greedy Randomized Adaptive Search Procedure*, ou em tradução livre, Procedimento de Pesquisa Adaptativa Aleatória Gananciosa, cuja proposição foi feita por Feo e Resende (1995). Sua essência consiste em duas etapas, sendo a primeira voltada à criação de uma solução inicial e a segunda com objetivo de efetuar uma busca local para melhorar a qualidade da solução inicial. Esta metaheurística é dita construtiva, pois foca em gerar uma solução inicial de melhor qualidade e utiliza a busca local apenas para pequenas melhorias.

O método executa múltiplas aplicações de busca local, cada uma iniciando de uma solução diferente. Cada busca consiste em uma iteração, sendo estas independentes, isto é, uma iteração não leva em conta o resultado obtido por outra. Normalmente, o critério que finaliza o algoritmo é dado por um número máximo de iterações. Ao atingir o critério de parada, a solução apresentada será aquela de melhor resultado, independentemente da

[4] k_b é uma constante física que relaciona temperatura e energia de moléculas, cujo valor é 1,38 $\times$ 10^{-23} J/K, (CALLISTER, 2007)

iteração onde a mesma foi encontrada.

Além dos algoritmos de Busca Tabu, *Simulated Annealing* e GRASP, outra importante classe de métodos é a dos Algoritmos Genéticos. Tais algoritmos permitem explorar várias regiões do espaço de soluções em cada iteração, dado que, ao longo das iterações, não é construída uma única trajetória de busca de solução, uma vez que as novas soluções são sempre obtidas através de combinações das soluções anteriores.

Como já mencionado, a busca pelo conjunto das soluções Pareto-ótimas em otimização multiobjetivo, requer, em vários problemas, algoritmos com tempo de processamento exponencial – algo de grande custo computacional – mesmo que a otimização de alguns objetivos de maneira isolada seja simples. Sendo assim, os métodos heurísticos se tornam extremamente interessantes na tratativa destes problemas, em especial, as metaheurísticas baseadas em Algoritmos Genéticos, dadas suas características de combinações de soluções, que oferecem grandes possibilidades de encontrar o conjunto Pareto-ótimo ou algo muito próximo dele, com baixo custo computacional.

1.2.3 Algoritmos evolutivos

Algoritmos evolutivos são procedimentos computacionais aplicados à resolução de problemas e podem ser considerados técnicas de computação bioinspirada ou computação natural, pois são ferramentas que utilizam técnicas baseadas em conceitos biológicos, especialmente àquelas relacionadas à evolução e genética (GABRIEL; DELBEM, 2017).

Inicialmente propostos na década de 1930, voltados à criação de algoritmos para exploração de múltiplos picos de uma função objetivo, os algoritmos evolutivos passaram a ganhar relevância somente cerca de trinta anos mais trade, onde o acesso a computadores começou a intensificar-se (JONG, 2006).

A área de estudos relacionada aos algoritmos evolutivos vem se expandindo rapidamente e, dentre os motivos para isto, destacam-se a capacidade de os mesmos encontrarem soluções satisfatórias para problemas complexos, principalmente àqueles não resolvidos por técnicas convencionais da computação. Além disso, são algoritmos relativamente simples de serem implementados e altamente adaptáveis aos problemas comuns de várias áreas de conhecimento (GABRIEL; DELBEM, 2017).

Ao contrário da maior parte das técnicas exatas de otimização, que normalmente geram uma solução por iteração, os algoritmos evolutivos constroem conjuntos de soluções, fato que, em muitos casos, permite uma resolução mais rápida e menos custosa computacionalmente. A construção da solução é função de uma metaheurística baseada na seguinte sequência básica: realização de reprodução com herança genética, introdução de variações aleatórias, estimulo de competições e seleção de indivíduos de uma população.

Este processo ocorre através de interações, onde cada iteração é considerada uma geração de soluções (KNOWLES; CORNE; DEB, 2007).

Em problemas de otimização multiobjetivo, onde a qualidade da solução é definida com base na sua adequação em relação a vários objetivos possivelmente conflitantes, boa parte dos métodos de programação matemática (ou exatos), na prática, buscam condensar os objetivos do problema em um único, atribuindo pesos aos mesmos, por exemplo, e resolvendo, então, apenas uma função objetivo.

Porém, a escolha de pesos adequados requer um conhecimento prévio das características da solução, de modo que possa ser feita uma definição de preferências a-priori, o que, além de nem sempre ser possível, acaba por inserir maior grau de subjetividade ao processo de busca pela solução. Por este motivo, a utilização de técnicas que buscam soluções que apresentem compromisso entre os diversos objetivos do problema sem a necessidade de prévio conhecimento do mesmo ganha relevância.

Neste sentido, os algoritmos evolutivos surgem como opção, pois, além de serem facilmente adaptados aos vários tipos de problemas , geram não somente uma, mas sim um conjunto de soluções possivelmente ótimas.

Outro aspecto importante presente nos algoritmos evolutivos consiste no fato de que tais métodos apresentam grandes possibilidade de escapar de ótimos locais e atingir soluções globalmente ótimas.

1.2.3.1 Algoritmo Genético

Inicialmente apresentados por John Holland, em 1975, os Algoritmos Genéticos constituem parte da Computação Evolutiva e são um conjunto de métodos computacionais inspirados na teoria biológica de evolução natural das espécies, apresentada por Charles Darwin. Os Algoritmos Genéticos são heurísticas facilmente adaptáveis e podem gerar soluções de boa qualidade em problemas complexos e de grande porte, com custo computacional viável. Fato que faz com que tais algoritmos venham sendo utilizados com grande sucesso em ampla gama de problemas de otimização combinatória e NP-difíceis (GOLDBERG, 1989).

Essencialmente, Algoritmos Genéticos visam encontrar a solução ótima de problemas, baseados em estratégia de busca paralela, estruturada e estocástica. Embora seja aleatória, a busca é considerada direcionada ao passo que considera informações históricas (soluções já encontradas).

O processo de busca pela solução envolve um procedimento relativamente simples, onde, em uma população inicial formada por indivíduos (que caracterizam um conjunto inicial de soluções), ocorre uma combinação, de modo que sejam gerados novos indivíduos, os quais são avaliados e selecionados, afim de que volte a acorrer um processo

de combinação entre os indivíduos selecionados para que uma nova população seja formada, avaliada e recombinada até que um determinado critério de parada seja atingido. A solução é aquela com o melhor valor dentre todas as que foram encontradas, mesmo que esta tenha sido descartada durante parte do processo de combinação.

Ao passo em que a maioria dos algoritmos voltados para otimização iniciam o processo de busca pela solução através de uma única solução, Algoritmos Genéticos partem de um conjunto de soluções, sendo tal conjunto um fator diretamente relacionado com número de soluções que serão encontradas. As soluções, por sua vez, são representadas por vetores, os quais também podem ser chamados de cromossomos, sendo que o número de genes em cada cromossomo é função da quantidade de parâmetros do problema e representam a maneira de otimizá-lo, onde o valor atribuído a cada gene, que nada mais é do que uma variável do problema, é denominado alelo (RAMPAZZO, 2012).

A população inicial pode ser fornecida através de alguma solução previamente conhecida, encontrada por um método qualquer ou por dado histórico, ou poder ser gerada aleatoriamente – sendo este último o mais comum e mais simples dos casos, pois dispensa conhecimentos prévios acerca de possíveis soluções. Entretanto, a inicialização de solução a partir de indivíduos já conhecidos pode facilitar a evolução e agilizar o processo de busca.

O processo de evolução das soluções contempla a avaliação e seleção dos indivíduos. Tal processo é executado através de técnicas denominadas operadores unários, sendo estas: cruzamento (ou *crossover*), mutação e seleção.

Operadores unários de cruzamento são considerados de ordem elevada e geram novos indivíduos através de recombinação de características entre dois ou mais genitores, fazendo com que o processo de herança genética seja atingido. Os operadores genéticos de mutação são responsáveis pela criação de novos indivíduos partindo de um único genitor, corroborando para a manutenção da diversidade genética da população. O operador de cruzamento é predominante em relação ao de mutação, sendo o primeiro, portanto, aplicado com maior probabilidade do que o último.

A velocidade de incremento de novas soluções está intimamente relacionada com o valor da probabilidade de cruzamento. Maiores probabilidades de cruzamento resultam na inserção mais rápida de novos indivíduos na população, porém, isto nem sempre é vantajoso, pois pode ocasionar perda de soluções com boa aptidão (*fitness*). Entretanto, baixas probabilidades de cruzamento podem tornar o algoritmo lento. A partir do cruzamento, busca-se gerar filhos com boas heranças genéticas dos progenitores, de modo a melhorar as características da nova população. Durante o processo de seleção, é indicado que a população genitora não seja totalmente descartada, permitindo maior diversificação em cruzamentos futuros.

A inserção de mutação é fundamental, pois a mesma, quando em baixas

probabilidades, contribui para que a busca por soluções não fique presa em regiões de busca local, evitando soluções de ótimo local. Entretanto, probabilidades elevadas de mutação tornam a busca essencialmente aleatória.

Uma maneira eficiente de selecionar os indivíduos que comporão as futuras populações é através da seleção elitista, a qual privilegia os indivíduos com maior aptidão, garantindo que estes continuem na futura geração de soluções (RAMPAZZO, 2012).

É necessário introduzir um critério de parada nos Algoritmos Genéticos, sendo os seguintes, os mais comuns::

- Definição de número máximo de gerações. Apesar de ser uma forma simples, corre-se o risco de o algoritmo parar e a solução ótima ainda não ter sido encontrada;
- Interrupção do processo quando julga-se ter uma boa solução. Um fator negativo é a necessidade de conhecer soluções prévias para comparação;
- Nível de convergência da população. Interrompe-se o algoritmo quando uma certa quantidade da população possuir características iguais;
- Tempo de execução.

De maneira geral e simplificada, um Algoritmo Genético, cujo critério de parada é o número de gerações G (ou iterações), pode ser estruturado da seguinte maneia:

Procedimento Algoritmo Genético (população, gerações)
{
inicialização (população);
avaliação (população);
para $t = 1$ até G
{
seleciona pais (população);
crossover (população);
mutação (população);
avaliação (população);
}
}

1.2.3.2 Algoritmo Genético e otimização multiobjetivo

As técnicas de otimização de problemas multiobjetivo apresentadas na subseção 1.2.1, apesar de gerarem, em determinados casos, soluções satisfatórias, podem não ser são eficientes na otimização de alguns problemas com diversos objetivos. Isto devido a alguns motivos, dentre os quais destacam-se a geração de apenas um elemento do conjunto Pareto-ótimo cada vez que a técnica é executada e a necessidade, em alguns casos, de atribuição de pesos ou ordem de preferência aos objetivos.

Por sua vez, os Algoritmos Genéticos são capazes de gerar vários elementos do conjunto Pareto-ótimo a cada execução do algoritmo, não requerem a introdução de parâmetros adicionais aos problemas, como pesos, por exemplo. Tais fatores tornam estes algoritmos excelentes ferramentas para a solução de problemas de otimização multiobjetivo.

Com perspectivas de solucionar os problemas apresentados pelos métodos de programação matemática, o primeiro Algoritmo Genético voltado para otimização multiobjetivo foi apresentado por Schaffer (1985). Tal algoritmo recebeu a denominação de *Vector Evaluated Genetic Algorithm* (VEGA), cuja grande diferença em relação aos métodos aplicados em otimização multiobjetivo até então consistiu na avaliação individual de cada objetivo do problema. Porém, dentre seus limitantes, encontram-se a criação de elementos que se destacam muito somente em alguns objetivos e a incapacidade de obtenção de diversidade adequada nas soluções ao longo da fronteira de Pareto (TICONA, 2003).

Em função das desvantagens contidas no método VEGA, foi proposto por Goldberg (1989) um novo procedimento, onde as soluções são classificadas de acordo com o conceito de dominância de Pareto, o qual fornece um valor de aptidão para determinada solução com base na quantidade de soluções que a mesma domina. Além disso, com o intuito da manutenção de diversidade de soluções, foi introduzido um método de compartilhamento que determina o nicho de cada solução dentro da fronteira em que a solução está contida. A partir das definições apresentadas por Goldberg (1989), novos algoritmos surgiram, dentre os quais destacam-se: *Multiobjective Genetic Algorithm* (MOGA), *Non-dominated Sorting Genetic Algorithm* (NSGA) e *Niched-Pareto Genetic Algorithm* (NPGA).

O conceito de seleção por elitismo colaborou para o surgimento de novos métodos voltados à resolução de problemas de otimização multiobjetivo. A utilização do elitismo na solução permite, por exemplo, o armazenamento de soluções não dominadas encontradas até o momento, de modo a preservar as melhores soluções. Algoritmos Genéticos que utilizam elitismo no processo de seleção produzem soluções melhores do que aqueles que não o fazem (ZITZLER; DEB; THIELE, 2000).

A seguir serão discutidos alguns dos principais Algoritmos Genéticos utilizados em problemas de otimização multiobjetivo.

VEGA – *Vector Evaluated Genetic Algorithm*

Foi o primeiro algoritmo genético voltado para otimização multiobjetivo, tendo sido apresentado por Schaffer (1985). Este algoritmo tem como principal diferença em relação aos algoritmos genéticos convencionais o fato de fazer uso de uma forma especial de seleção. A cada iteração são geradas várias subpopulações, sendo uma para cada objetivo do problema. Os indivíduos destas subpopulações são gerados aleatoriamente com base em uma roleta ponderada e selecionados em função da aptidão. Após selecionados, tais indivíduos são misturados e passam a compor a nova população, na qual são aplicados os operadores genéticos de *crossover* e mutação.

Um problema apresentado pelo VEGA consiste na geração de indivíduos que se destacam em relação aos demais apenas quando considera-se determinados objetivos, podendo ocorrer, deste modo, a seleção de indivíduos que não apresentam comportamento satisfatório quando todos os objetivos são considerados, impedindo, portanto, boa caracterização do conjunto Pareto-ótimo. Além disso, tal método não utiliza o conceito de dominância de Pareto no processo de seleção.

A seguir serão abordados algoritmos que utilizam o conceito de dominância de Pareto.

MOGA – *Multiobjective Genetic Algorithm*

O algoritmo MOGA, proposto por Fonseca, Fleming et al. (1993), considera o conceito de dominância na classificação das soluções e, para isto, faz uso de um método onde a classificação de cada indivíduo i é função da quantidade de outros indivíduos que este domina. Esta função de classificação determina que um indivíduo i recebe o valor um acrescido de mais um para cada indivíduo que o domina. Os indivíduos não dominados são retirados da população total e recebem uma classificação (a melhor dentre todas as demais populações). Dos indivíduos remanescentes é então formado um novo grupo de indivíduos não dominados, o qual recebe também uma classificação, inferior àquela concedida ao grupo que já foi retirado. O processo é repetido até que toda a população esteja agrupada e classificada.

Em sequência deve-se atribuir um valor de aptidão aos indivíduos da população total. Esse *fitness* é calculado através de uma interpolação entre o melhor e o pior valor de classificação, normalmente utilizando-se de uma função linear ou exponencial. É então atribuído, para cada indivíduo i, a média de *fitness* do grupo ao qual o mesmo pertence, garantindo que as melhores soluções possuam melhores aptidões.

Com o intuito de evitar o problema de convergência prematura e garantir

diversidade de soluções, pode-se utilizar, para cada classificação, o método de formação de nichos. Uma vez obtidos os nichos, é calculado o *fitness* compartilhado de cada solução, levando-se em consideração a ordem de classificação. Esse método faz com que soluções contidas em nichos pouco povoados tenham melhores valores de aptidão. Um dos fatores negativos presentes no MOGA consiste na necessidade de o usuário ser o responsável pela determinação do fator de compartilhamento, o qual é um fator crítico do processo.

NSGA – *Nondominated Sorting Genetic Algorithm*

A técnica NSGA foi apresentado por Srinivas e Deb (1994) e, assim como o MOGA, utiliza conceitos de dominância de Pareto na classificação de soluções. O algoritmo é baseado em um processo de classificação de indivíduos em múltiplas camadas de dominância. A primeira camada é composta por todos os indivíduos da população inicial que não são dominados por nenhum outro. Aos indivíduos desta camada é atribuído um valor de *fitness*, proporcional ao tamanho da população da camada, ocasionando igual potencial de reprodução para todos os indivíduos – este *fitness* igualitário também pode ser chamado de *fitness dummy*. Posteriormente, aos indivíduos da primeira camada, é aplicado o *fitness sharing* (ou aptidão por compartilhamento), penalizando seus respectivos *fitness* com base no número de indivíduos que compartilham a mesma vizinhança (ou nicho) de modo a manter a diversidade da população – o tamanho da vizinhança é determinado por um parâmetro denominado raio de nicho. Concluído este processo, os indivíduos da primeira camada são removidos da população inicial e uma nova etapa de classificação se inicia para os indivíduos restantes, sendo que o *fitness dummy* atribuído aos indivíduos da segunda camada deve ser menor do que o conferido aos indivíduos da primeira. O processo de classificação é continuamente executado até que toda a população remanescente seja não dominada.

Ao passo que o *fitness* dos indivíduos da primeira camada é maior, a probabilidade destes participarem do processo de reprodução também o é, potencializando, deste modo, a busca por soluções não dominadas e, consequentemente, a velocidade de convergência.

Uma das principais vantagens do NSGA consiste na possibilidade de resolver problemas com qualquer quantidade de objetivos, aliada a divisão proporcional da função de aptidão, de modo a permitir melhor distribuição dos indivíduos no espaço objetivo.

Apesar das vantagens, a ordenação das soluções com base no conceito de dominância Pareto é de alta complexidade computacional e, assim como no MOGA, é responsabilidade do usuário especificar o parâmetro de compartilhamento.

NPGA – *Niched Pareto Genetic Algorithm*

Proposto por Horn, Nafpliotis e Goldberg (1994), o NPGA utiliza uma técnica de seleção baseada em dominância de Pareto. A seleção acontece por meio de um torneio cuja disputa não é restrita exclusivamente a dois indivíduos. Neste método utiliza-se cerca de 10% da população total, selecionada aleatoriamente, como parâmetro de comparação para determinação da relação de dominância durante o torneio. No torneio, dois indivíduos aleatórios competem com os indivíduos da subpopulação selecionada e o vencedor é aquele que não for dominado pelo conjunto de comparação. Em caso de empate, quando os dois indivíduos são dominados ou não dominados pela subpopulação, a definição do vencedor passa a ser função da quantidade de indivíduos do nicho ao qual cada um dos dois indivíduos pertence, sendo o vencedor aquele que pertencer ao grupo menos populoso.

Embora seja um algoritmo rápido, em função do tamanho da subpopulação selecionada para comparação, o NPGA é bastante afetado pelo tamanho desta subpopulação. Uma subpopulação muito grande fará com que ocorra conversão prematuramente para apenas determinada parte da fronteira de Pareto, ao passo que uma subpopulação muito pequena, poderá gerar muitas soluções dominadas na população final.

Os próximos algoritmos apresentados são técnicas onde foram incorporadas os conceitos de elitismo no processo de seleção.

SPEA – *Strength Pareto Evolutionary Algorithm*

A abordagem SPEA foi proposta por Zitzler e Thiele (1999) e acrescentou algumas características importantes de melhoria em relação aos métodos existentes até então. Dentre estas características destacam-se o armazenamento em arquivo independente das soluções não dominadas encontradas durante o processo, aliado a um método de *clustering* voltado para redução do número de soluções não dominadas que são armazenadas e a utilização da relação de dominância de Pareto para a atribuição de *fitness* para cada indivíduo.

O método tem início pela etapa de classificação dos indivíduos da população original em dominados e não dominados, sendo estes últimos armazenados no arquivo externo. Posteriormente é atribuído *fitness* aos indivíduos, sendo que esta etapa divide-se em duas fases: na primeira delas é atribuído um valor relacionado com a força de Pareto de cada indivíduo i contido no arquivo – este valor é proporcional ao número de indivíduos da população atual que este indivíduo i domina; em seguida, é calculado um valor de *fitness* para cada indivíduo i externo aos armazenados, de maneira a somar a força de Pareto de cada indivíduo contido no arquivo que domina o indivíduo externo i. Realiza-se, então o processo de seleção, que ocorre através de torneio binário.

Embora o SPEA tenha se mostrado bastante competitivo quando comparado aos métodos NPGA, VEGA e NSGA, o mesmo apresenta algumas fraquezas, dentre as quais destaca-se o fato de a atribuição de *fitness* ser igual para indivíduos externos ao arquivo que são dominados pelos mesmos indivíduos dentro do arquivo, fator que pode levar o algoritmo a um comportamento de busca aleatória em casos onde exista apenas um indivíduo no arquivo.

SPEA 2 – *Strength Pareto Evolutionary Algorithm 2*

Com o objetivo de resolver as fraquezas encontradas no SPEA, Zitzler, Laumanns e Thiele (2001) propuseram o SPEA 2. As principais diferenças do SPEA 2 em relação ao seu antecessor são: modificação na função de aptidão, onde a avaliação de cada indivíduo passa a considerar o número de indivíduos que são dominados por ele e também o número de indivíduos que o dominam, evitado que ocorram valores iguais de *fitness*; utilização de informações sobre a concentração de indivíduos da população total, de modo a permitir busca por soluções em regiões pouco exploradas; truncamento dos indivíduos do arquivo, evitando a perda de indivíduos não dominados da fronteira de Pareto.

PAES – *Pareto Archived Evolution Strategy*

Desenvolvido por Knowles e Corne (2000), o algoritmo PAES tem como ideia fundamental preservar um conjunto composto por soluções não dominadas, extraindo, deste conjunto, a cada iteração, um indivíduo responsável por realizar pequenas perturbações na vizinhança. A partir destas perturbações, avalia-se candidatos a comporem o conjunto de soluções não dominadas. A admissão de uma solução a este conjunto acontece quando a solução candidata domina a solução que gerou a perturbação e, atrelada a esta condição, a solução candidata não pode ser dominada por nenhuma solução do presente conjunto. Além disso, o processo de aceitação do indivíduo candidato também considera a manutenção da diversidade da população.

PESA – *Pareto Envelope-based Selection Algorithm*

Apresentado por Corne, Knowles e Oates (2000) o algoritmo PESA combina seleção e preservação de diversidade trabalhando com duas populações, uma principal e uma secundária, sendo que esta última mantém todos os indivíduos da população e a principal é incumbida de armazenar os indivíduos não dominados.

O método utiliza uma estrutura em grade para avaliar a distribuição de indivíduos na população secundária e selecionar indivíduos que originarão um novo indivíduo. A partir desta análise, são realizados dois torneios simples, sendo que o vencedor de cada torneio é aquele indivíduo que ocupar uma região de menor densidade populacional em relação

a seu rival. A partir da recombinação de dois indivíduos vencedores dos torneios, surge um terceiro indivíduo que, em sequência, é submetido a um processo de mutação. Para que este novo indivíduo seja incluído na população secundária, avalia-se também a concentração desta, sendo que a participação na população principal está sujeita somente à relação de dominância de Pareto, ou seja, a população principal aceita apenas indivíduos não dominados.

NPGA 2 – *Niched Pareto Genetic Algorithm 2*

Proposto por Erickson, Mayer e Horn (2001), o algoritmo NPGA 2 tem como objetivo solucionar a grande fraqueza do NPGA, cuja solução é altamente dependente da subpopulação utilizada na comparação para determinação da relação de dominância de Pareto durante o torneio de seleção. Para isto, no NPGA 2 a classificação passa a ser baseada no grau de dominância de Pareto.

A seção seguinte é dedicada ao *Nondominated Sorting Genetic Algorithm II* (NSGA-II), proposto por Deb et al. (2000). Um tópico exclusivo para este algoritmo será desenvolvido para apresentá-lo com maior grau de detalhes, uma vez que o Algoritmo Genético proposto neste trabalho utiliza-se das funções de *ranking* (classificação por não dominância) e *crowding distance* (distância de aglomeração) presentes no NSGA-II.

1.2.3.3 NSGA-II - *Nondominated Sorting Genetic Algorithm II*

Dentre os Algoritmos Genéticos voltados para otimização multiobjetivo apresentados na literatura, o *Nondominated Sorting Genetic Algorithm II*, proposto por Deb et al. (2000) obteve sucesso no tratamento de diversos tipos de problemas. O algoritmo considera explicita e simultaneamente diferentes funções objetivo, encontrando um conjunto de soluções não dominadas no final das gerações, de modo que o decisor possa optar pela solução que mais se enquadra no contexto do problema.

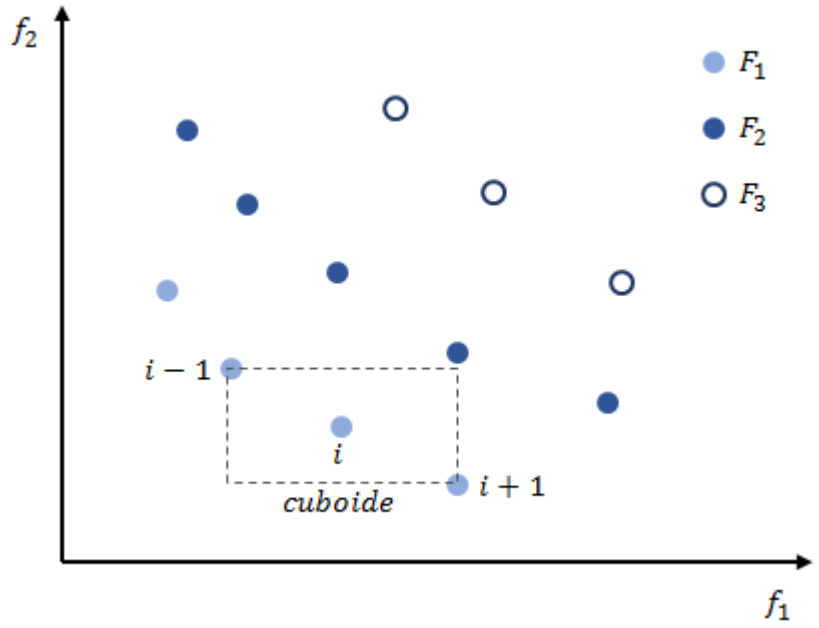
O Algoritmo Genético proposto neste trabalho reunirá algumas das principais características propostas no NSGA-II e já conhecidas como importantes ou necessárias na resolução de problemas de otimização multiobjetivo, as quais são as funções de *ranking* (classificação por não dominância) e *crowding distance* (distância de aglomeração). Além disto, novas funcionalidades necessárias e específicas para a resolução dos problemas de estudo de caso serão acrescentadas. A seguir, com base no conteúdo apresentado em Deb et al. (2000) e em Deb et al. (2002), é descrita a forma de funcionamento do NSGA-II.

A primeira etapa do algoritmo consiste na inicialização, usualmente aleatória, de uma população P_g, seguida de sua posterior avaliação em relação aos valores das funções objetivo e das restrições do problema. Posteriormente, para cada indivíduo de P_g

é associada uma classificação (*rank*) e uma distância de aglomeração (*crowding distance*).

O processo de classificação consiste em separar cada indivíduo em diferentes categorias (ou diferentes fronteiras) de acordo com critérios de dominância. Tal processo inicia-se pela formação da primeira fronteira, que é composta somente por indivíduos não dominados (melhores indivíduos). Em seguida, a segunda fronteira é obtida através da aglutinação dos indivíduos não dominados que restaram após a seleção daqueles que compõem a população da primeira fronteira. Isto se repete até que não restem mais indivíduos na população fora de alguma fronteira. Caso existam restrições, considera-se a factibilidade das soluções durante a classificação, sendo que as soluções infactíveis recebem o mesmo e maior valor de *rank* dentre as soluções avaliadas. A Figura 4 mostra um exemplo de fronteiras (F_1, F_2 e F_3) para um problema com dois objetivos (f_1 e f_2).

Figura 4 – Ordenação da população em fronteiras e cálculo de distância de aglomeração.



Fonte: adaptado de Deb et al. (2002).

A distância de aglomeração (*crowding distance*) é utilizada para privilegiar pontos extremos ou pontos mais espalhados pelas diferentes fronteiras. O objetivo é utilizar a distância de cada indivíduo i em relação aos indivíduos mais próximos $(i-1)$ e $(i+1)$, formando um cuboide, conforme mostrado na Figura 4. Esta abordagem não requer nenhum parâmetro definido pelo analista para a manutenção da diversidade da população.

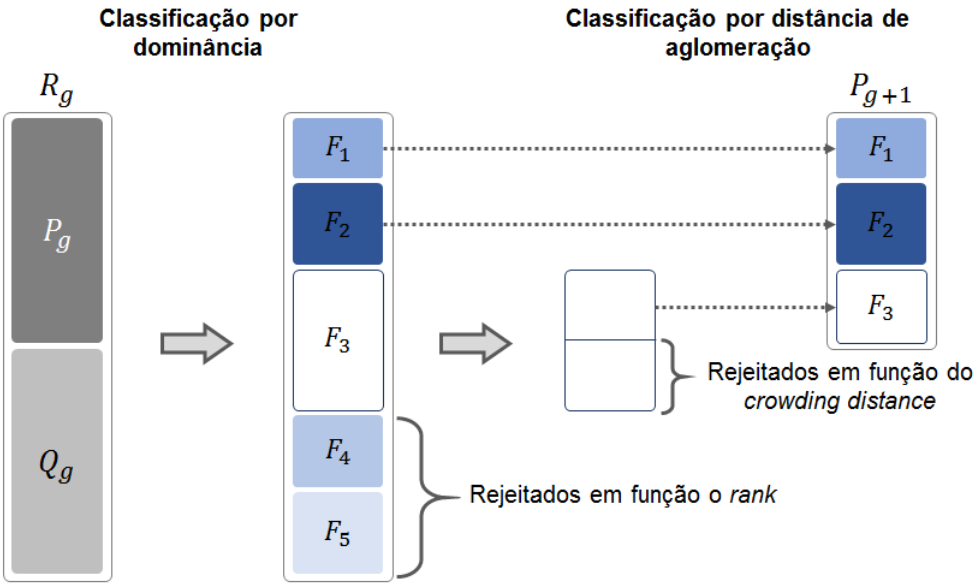
Uma vez avaliados, todos os indivíduos factíveis da população atual P_g são selecionados para criação de uma população de pais de tamanho TP. Como operador de seleção utiliza-se o torneio. Neste método de seleção, duas soluções factíveis, p e q são selecionadas aleatoriamente da população P_g para competirem. Se $p \preceq q$, p fará

parte da população de pais; caso contrário, q será escolhida. O processo se repete até que a população de pais se complete com TP indivíduos. Pares de pais são formados aleatoriamente com os indivíduos desta seleção que passaram pelo torneio, podendo estes indivíduos passarem por cruzamento e mutação, dependendo das probabilidades PC e PM, as quais são, respectivamente, probabilidade de cruzamento e probabilidade de mutação.

Após a população de pais, eventualmente, passar pelos operadores de mutação e cruzamento, é formada a população de filhos Q_g, onde dois pais são utilizados para a formação de dois filhos. Desta forma, Q_g também será composta por TP indivíduos. A combinação da população atual P_g com a população de filhos Q_g formará a população auxiliar R_g. Esta população auxiliar, de tamanho $2TP$, será classificada de acordo com os critérios de *rank* e *crowding distance*.

Esta nova ordem parcial permite a utilização conjunta dos critérios de dominância e distância de aglomeração durante o processo de seleção dos indivíduos da população auxiliar R_g que farão parte da população da próxima geração ($P_{(g+1)}$). O esquema geral deste processo é apresentado na Figura 5.

Figura 5 – Esquema de processo seletivo para formação da geração $g + 1$.



Fonte: adaptado de Deb et al. (2002).

Se $F1$ (primeira fronteira) é menor que o tamanho da população TP, todos os membros de $F1$ são escolhidos para a nova população ($P_{(g+1)}$). Os membros restantes são escolhidos das fronteiras subsequentes. Supondo que $F3$ (com base na ilustração da Figura 5) é a última fronteira antes que a nova população atinja o tamanho TP, mas nem todos os membros de $F3$ podem ser acomodados, então, para a escolha dos membros de

R_g que completarão a nova população, prefere-se as soluções com maiores valores de *crowd distance*.

Ou seja, de maneira geral, durante o processo de formação de novas gerações, entre duas soluções de diferentes fronteiras, prefere-se a solução com menor *rank*. Se duas soluções pertencem à mesma fronteira, a preferência é dada àquela que pertence à região menos povoada, ou seja, com maior *crowd*.

2 Problemas para estudo de caso

Este capítulo tem por objetivo apresentar e discutir os problemas selecionados para estudo de caso que serão resolvidos pelo Algoritmo Genético posposto neste trabalho.

Inicialmente serão abordados os contextos e características do problema de alocação de portfólio (ou alocação de capital), em especial suas aplicações por empresas do setor financeiro. Em sequência, será abordado o problema de alocação de aeronaves em portões de embarque de aeroportos. Entretanto, como este último é um problema contido na grande área de problemas de agendamento do setor aeroportuário, será feita uma breve análise acerca de problemas de agendamento deste setor.

2.1 Alocação de portfólio

O gerenciamento eficaz de portfólios é vital para o sucesso das empresas. Em um mercado global e competitivo, selecionar de maneira eficiente a alocação de capital pode ser a diferença entre companhias de sucesso e falências. Neste sentido, o gerenciamento de portfólios recai sobre escolhas estratégicas – em quais mercados, produtos e tecnologias serão feitos os investimentos, por exemplo. Assim como é importante para empresas, a alocação eficiente de capital também é fundamental para pessoas físicas ou instituições que investem no mercado de capitais.

De maneira geral, problemas de seleção de portfólios, ou carteira de investimentos, consistem em determinar de que maneira um investidor deve aplicar determinado capital em dado conjunto de opções de investimento.

Apesar de ter passado quase sete décadas desde sua proposição, a Teoria Moderna de Portfólios (*Modern Portfolio Theory* – MPT), desenvolvida por Markowitz (1952), ainda é amplamente utilizada por investidores do mercado de capitais. Tal teoria auxilia o investidor na escolha do portfólio de menor risco (neste caso, medido de acordo com o grau de variabilidade de preço das ações) e é baseado na suposição de que os investidores são avessos ao risco, preferindo uma carteira com menor risco para um determinado nível de retorno. Sob tal hipótese, os investidores só assumirão investimentos de alto risco se puderem esperar uma recompensa maior. De acordo com a teoria, o risco não sistêmico de um portfólio pode ser minimizado com a escolha eficiente de ativos, sendo tal escolha baseada na variação de cada ativo, juntamente com as correlações entre pares de ativos. As correlações de ativos afetam o risco total da carteira, gerando um desvio padrão menor do que seria encontrado por uma soma ponderada.

O MPT tem como objetivo minimizar a variância de um portfólio para um

dado nível de retorno esperado, supondo que o único objetivo do investidor é minimizar o risco. Porém, assim como em outros problemas práticos, diferentes investidores possuem diferentes objetivos e, em muitos casos, múltiplos objetivos, os quais podem ser conflitantes. Isto posto, surge a necessidade da consideração de modelos de otimização multiobjetivo, pois estes refletem melhor uma realidade complexa e possibilitam a exploração de uma gama maior de soluções.

Neste sentido, Armananzas e Lozano (2005) propõem um modelo de otimização de portfólio multiobjetivo que utiliza as técnicas *Greedy Search*, *Simulated Annealing* e Colônia de Formigas, ao passo em que Skolpadungket, Dahal e Harnpornchai (2007) formulam um modelo multiobjetivo que busca maximizar o retorno e minimizar o risco em um determinado período de investimento e, para a obtenção da solução, utiliza os algoritmos VEGA, Fuzzy VEGA, MOGA, SPEA 2 e NSGA-II. Assim, como Skolpadungket, Dahal e Harnpornchai (2007), Zhu et al. (2011) também propõem um modelo cujos objetivos são maximizar o retorno e minimizar o risco de um portfólio de ações. Para resolver o problema, utilizam uma técnica de otimização por enxame de partículas.

Uma abordagem do problema de otimização de portfólios cujos objetivos são retorno, risco e liquidez é proposta por Li e Xu (2013). Para criar o modelo, os autores consideram tanto dados históricos como opiniões de especialistas, de forma que a preferência individual do investidor é refletida nos parâmetros do modelo. Para a solução do problema, foi desenvolvido um algoritmo genético, acompanhado de um exemplo numérico para exemplificação.

Buscando maximizar o lucro e minimizar o risco, Lwin, Qu e Kendall (2014) desenvolvem um algoritmo evolucionário, onde uma estratégia de geração de solução orientada para o aprendizado é incorporada ao processo de otimização multiobjetivo. O problema é modelado considerando o lote padrão de compras de ações, bem como condições que limitem o número de ativos de uma carteira ou que exigem que determinados ativos sejam incluídos na mesma.

Saborido et al. (2016) sugerem um algoritmo genético para resolver o problema cujos objetivos são maximização do retorno, minimização do risco de queda no preço dos ativos e minimização da assimetria de um determinado portfólio, levando em conta restrições orçamentárias. O desempenho do algoritmo foi avaliado utilizando um conjunto de dados do mercado acionário espanhol e mostrou-se eficiente na busca pelos objetivos propostos.

Embora sejam mais comuns publicações focadas em otimização de portfólios de ativos do mercado financeiro, há também autores que tratam do tema alocação de capital com o enfoque nas necessidades de empresas, em especial, empresas do setor financeiro. Gueyié, Guidara e Lai (2019), por exemplo, embora não proponham nenhuma forma que otimize a alocação de capital, fazem uma análise de demonstrações financeiras

de seis grandes bancos canadenses e identificam que, ao longo do tempo, de 1982 até 2018, tais bancos reduziram suas exposições em setores de maiores riscos, migrando para setores de serviço, obtendo maiores receitas. Em trabalho com características similares, Zarutskie (2013), examina como as carteiras de empréstimos dos bancos comerciais dos EUA mudaram em resposta à ascensão dos mercados de securitização e aos processos de desregulamentações do mercado bancário entre os anos de 1976 e 2003. De acordo com o apresentado neste último estudo, os bancos inclinam cada vez mais seus portfólios para empréstimos garantidos por imóveis, sendo que os bancos maiores e os bancos mais jovens deslocam de forma desproporcional seus empréstimos para operações lastreadas em imóveis, enquanto bancos menores e bancos mais antigos mantêm maiores parcelas de suas carteiras de empréstimos nas modalidades pessoais e comerciais. Empréstimos com garantia apresentam menores riscos, no entanto, apresentam taxas de retornos menos elevadas quando comparados ao empréstimos sem garantias.

Antes de discutir a alocação de capital sob a ótica de investimento nas empresas de setor financeiro, deve-se atentar que, em virtude de as atividades centrais deste ramo serem fortemente relacionadas ao risco de crédito[1], as empresas deste setor, para cada investimento que realizam[2], devem possuir uma quantidade de capital reserva, normalmente designado pelo termo Capital Econômico Alocado (CEA), de modo que tal capital funcione como uma garantia contra perdas esperadas. O cálculo do CEA deve ser feito de acordo com a Circular nº 2.682/99 do CMN[3] do Banco Central do Brasil. De maneira geral, o Capital Econômico Alocado incorpora os seguintes componentes: risco de crédito (incluindo perda esperada), risco operacional, risco de mercado e risco de subscrição de seguros, sendo o de maior impacto, o risco de crédito. De acordo com a circular anteriormente mencionada, quanto maior o risco de crédito de determinada operação, maior é o valor do CEA, sendo este definido como um percentual do valor da operação de crédito. Quando uma operação de crédito é realizada, compulsoriamente, 0,5% do valor concedido na mesma deve ser destinado ao CEA, porém, conforme o risco de *default*[4] aumenta, o percentual atribuído ao CEA também aumenta, ao passo que 100% do valor da operação de concessão de crédito é alocado quando o atraso no pagamento, por parte do tomador de crédito, atinge cento e oitenta dias.

Embora a Circular nº 2.682/99 do CMN determine que o CEA seja realizado conforme a dívida é observada, muitas empresas do setor, principalmente aquelas onde é verificado maior conservadorismo perante ao controle de risco, utilizam-se de modelos

[1] Risco de crédito é a possibilidade de perdas decorrentes do não cumprimento pelo tomador, emissor ou contraparte de suas respectivas obrigações financeiras nos termos pactuados, da desvalorização de contrato de crédito em consequência da deterioração na classificação de risco do tomador, do emissor, da contraparte, da redução de ganhos ou remunerações, das vantagens concedidas em renegociações posteriores e dos custos de recuperação (IU, 2015).

[2] Neste caso, investimentos devem ser interpretados como concessão de crédito.

[3] Conselho Monetário Nacional.

[4] Sinônimo de risco de crédito.

de previsão de *default* para classificar seus clientes em grupos de risco antes mesmo de realizarem a concessão de crédito, de modo que, ao passo que concedem crédito, verificam qual a probabilidade do cliente não realizar o pagamento da dívida e, prontamente, alocam o CEA correspondente ao risco presumidamente assumido.

Em virtude de não ser o foco deste trabalho, não serão abordadas metodologias de cálculo de risco de *default* dos clientes, porém, caso o leitor se interesse pelo assunto, dentre outros, os seguintes trabalhos abordam o referido tema: Frey e McNeil (2002), Rockafellar e Uryasev (2002), Tasche (2002) e Gordy (2003).

Retomando a discussão sobre alocação de capital em empresas do setor financeiro, nota-se que a maior parte das publicações relacionadas ao tema o abordam basicamente sob a ótica do risco de crédito ou de aspectos regulatórios. Furfine (2001), por exemplo, desenvolve um modelo dinâmico e estrutural para analisar como os bancos ajustam suas carteiras de empréstimos ao longo do tempo. No modelo, os bancos experimentam choques de capital, enfrentam uma demanda incerta de empréstimos futuros e incorrem em custos com base em sua proximidade com os requisitos mínimos de capital regulatório. O modelo estimado é usado para prever a resposta ótima dos bancos em relação a mudanças na exigência de capital regulatório (CEA), mudanças na intensidade do monitoramento regulatório e desacelerações econômicas.

Stoughton e Zechner (2007) mostram que os investimentos ocorrem de maneira distintas em empresas com múltiplas divisões de tomada de decisão quando comparados com aqueles feitos em empresas com apenas uma camada de gerenciamento de risco. As avaliação das decisões de alocação de capital foram baseadas nos mecanismos de orçamento EVA[5] e RAROC[6]. Assim, como há diferenças na forma de investimentos em função dos níveis existentes para tomada de decisão dentro da empresa, de acordo com o apresentado em Morck, Yavuz e Yeung (2011), bancos cuja estrutura de controle é familiar ou em países onde o setor é altamente regulado pelo governo, apresentam baixa eficiência na alocação de capital.

Dell'Ariccia e Marquez (2004) analisam a alocação de capital em operações de crédito em função do conhecimento prévio de informações privadas do tomador de crédito. Por regulamentação legal, informações privadas não podem ser compartilhadas entre os bancos, o que torna tais informações uma fonte de insumos muito valiosas no momento de alocações de capital em operações de crédito. Dell'Ariccia e Marquez (2004) mostram que, em cenários onde há grande competitividade entre bancos, os mesmos normalmente tendem a focar seus esforços em clientes cujas informações privadas já são conhecidas, o que caracteriza busca por operações menos rentáveis, uma vez que são cobradas taxas

[5] EVA significa *Economic Value Added*, ou Valor Econômico Adicionado, e é um indicador que demonstra a criação ou destruição de valor do capital aplicado.

[6] RAROC é o *Risk adjusted return on capital*, ou Retorno Ajustado ao Risco no Capital, e é uma mensuração da rentabilidade baseada no risco de determinada operação.

menores destes tipos de clientes, porém menos arriscadas.

Carey (2002) propõe uma base amostral em cenário de estresse (deterioração econômica e eventual incremento do risco de crédito) para estimar os níveis de perdas do portfólio de crédito de uma instituição bancária, bem como os níveis adequados de alocação de capital econômico em tal cenário. Ao passo em que İç (2012), por sua vez, desenvolve um modelo de alocação de concentração de risco de crédito para que bancos possam determinar os limites de concentração de risco de crédito de suas unidades regionais. O modelo proposto é baseado nas abordagens *Fuzzy TOPSIS* e Programação Linear e combina várias restrições determinadas pelos próprios bancos.

Santos et al. (2012) trazem uma abordagem para o gerenciamento de risco ativo com base nos recentes regulamentos de Basileia II[7] para obter portfólios ótimos com alocações mínimas de capital. O desempenho ajustado ao risco da abordagem proposta mostrou-se superior ao observado em carteiras simuladas de VaR[8] mínimo e VaR mínimo estressado.

A Tabela 1 resume os objetivos abordados nas publicações voltadas à alocação de capital em empresas do setor financeiro abordadas nesta seção. As letras de A até I, indicam, respectivamente, as seguintes publicações: Gueyié, Guidara e Lai (2019), Zarutskie (2013), Furfine (2001), Stoughton e Zechner (2007), Morck, Yavuz e Yeung (2011), Dell'Ariccia e Marquez (2004) Carey (2002), İç (2012), Santos et al. (2012). Conforme indica a Tabela 1, um terço das publicações analisadas faz uma análise histórica do perfil de investimento de instituições financeiras, duas delas abordam o tema alocação de capital sob a ótica de regulações enfrentadas pelo setor, outras duas observam a influência da estrutura de decisão e da forma de administração nas decisões de alocação de capital, outras duas tratam da alocação de capital em função de eventual deterioração no cenário econômico no qual as empresas estão inseridas, e, finalmente, uma delas trata da alocação de capital em função de informações previamente obtidas do cliente e dos níveis de concorrência enfrentados.

Embora não explicitamente, todos os artigos citados na Tabela 1 abordam questões relacionadas a algum tipo de risco, uma vez que a gestão de risco é inerente às atividades do setor financeiro. Ao passo em que os autores abordam os temas de adequação aos quesitos de Basileia II, por exemplo, os mesmos estão indiretamente tratando de risco, uma vez que tal regulação trata principalmente do tomada de atitudes, por parte das instituições financeiras, que visem a mitigação do risco de crédito ao qual estas estão

[7] Basileia II (proposto em 2004) é a segunda versão do acordo de Basileia (proposto em 1988), o qual é definido pelo Comitê de Basileia para Supervisão Bancária (*Basel Committee on Banking Supervision*), que por sua vez é o fórum internacional para discussão e formulação de recomendações para a regulação prudencial e cooperação para supervisão bancária, composto por 45 autoridades monetárias e supervisoras de 28 jurisdições (BCB, 2019).

[8] *Value at Risk* (VaR), ou Valor em Risco, é um indicador que estima a perda potencial máxima de um investimento para um período de tempo, com um determinado intervalo de confiança.

sujeitas. Além disto, quando abordam o tema rentabilidade, através de indicadores como EVA e RAROC, estão indiretamente mencionando gestão de riscos, uma vez que tais indicadores estão relacionados aos riscos e retornos das operações de crédito.

Tabela 1 – Objetivos abordados nas publicações analisadas nesta seção.

Objetivos	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
Análise de mudança no perfil de investimento de bancos ao longo dos anos.	X	X	X						
Adequações de investimentos em função de regulações do setor.			X						X
Análise da influência dos níveis de hierarquia ou da forma de administração na alocação de capital.				X	X				
Adequações de investimentos em função de concorrência e informações conhecidas do cliente.						X			
Alocação de crédito em função de deterioração do cenário econômico.							X	X	

Fonte: Produzido pelo autor.

2.1.1 O problema de alocação de capital em empresas do setor financeiro

Além de alinhar o problema abordado nesta seção aos que foram vistos nos artigos analisados, buscou-se trazê-lo para próximo da realidade de operação de uma das maiores instituições financeiras do Brasil. Tal problema consiste na alocação de determinada quantidade de capital disponível em diversas operações (produtos) de crédito possíveis. Tais produtos estão sujeitos a riscos e retornos (lucros) distintos e possuem limite máximo de alocação em função da disponibilidade de clientes que tomarão tais créditos (demanda limitada). Os objetivos deste problema consistem em maximizar o lucro total e em minimizar a quantidade de capital alocado, sendo este, função da qualidade (risco) e quantidade de crédito concedido em cada produto. A solução para o problema consiste em determinar a fração (ou percentual) do capital total disponível que será alocado em cada produto disponível.

O problema aqui abordado reflete apenas uma parte das operações presentes na referida instituição financeira, sendo que, quanto menor for a alocação de capital necessária para obter lucro máximo (ou satisfatório), melhor, pois o capital excedente pode ser direcionado para outras operações existentes na empresa.

Os parâmetros considerados na formulação do problema de alocação de capital são os seguintes:

P: conjunto de produtos.

lu_i: lucro unitário do produto i.

r_i: risco associado ao produto i.

cr_i: crédito médio concedido aos clientes do produto i.

d_i: demanda máxima pelo produto i.

C: capital total disponível para alocação.

As variáveis de decisão do problema consistem no percentual do capital total disponível que será alocado em cada produto, a qual é descrita por p, cujo intervalo é [0, 1] e nas variáveis auxiliares $p\_início$ e b_i, que garantem que, caso haja alocação de capital em determinado produto i, tal capital permitirá que pelo menos 10% da demanda pelo referido produto seja atendida. $p\_início$ consiste na alocação inicial de capital e dará origem à variável p, de acordo com o valor de b_i, que é definida da seguinte forma:

$$b_i = \left\{ \begin{array}{ll} 1 & \text{se } p\_início_i \; C \geqslant 0{,}1 \; cr_i d_i r_i \\ 0 & \text{caso contrário} \end{array} \right\}$$

As formulações dos objetivos do problema, bem como suas restrições, são apresentadas a seguir:

$$\max \quad f_1 = \sum_{i=1}^{P} p_i \frac{lu_i C}{r_i cr_i} \tag{2.1}$$

$$\min \quad f_2 = \sum_{i=1}^{P} p_i C \tag{2.2}$$

$$\text{s.a.:} \quad p_i \geqslant p\_início_i b_i \tag{2.3}$$

$$p_i C \leqslant cr_i d_i r_i \tag{2.4}$$

$$\sum_{i=1}^{P} p_i \leqslant 1 \tag{2.5}$$

$$p_i \geqslant 0 \tag{2.6}$$

Onde:

- f_1: Maximização do lucro;
- f_2: Minimização do capital alocado.

A Equação 2.3 garante que, caso haja alocação de capital em determinado produto i, tal capital permitirá que pelo menos 10% da demanda pelo referido produto seja atendida. A Equação 2.4, por sua vez, impede que mais capital do que o necessário para o atendimento total da demanda seja alocado. A Equação 2.5 garante que não haverá alocação de capital além do disponível. E, finalmente, a Equação 2.6 não permite alocações negativas de capital.

Para detalhes de como foram obtidas a Equação 2.1 e Equação 2.2, deve-se consultar o Apêndice A.

Ao observa-se a Equação 2.1 e a Equação 2.2, percebe-se o conflito entre os objetivos do problema, uma vez que a variável de decisão p_i está contida no numerador

de ambas, sendo que uma delas busca maximizar e a outra tem por objetivo minimizar determinado valor.

O problema de alocação capital remete imediatamente à Otimização Multiobjetivo, já que possui objetivos conflitantes. A obtenção de soluções alternativas torna-se importante para o processo de tomada de decisão.

Os dados utilizados no problema desta seção são provenientes de uma instituição do setor financeiro brasileiro e, por motivos de confidencialidade, não podem ser divulgados.

2.2 Alocação de aeronave em portão de embarque

Esta seção apresenta uma revisão literária acerca de problemas de agendamento (*scheduling*) e suas aplicações no setor aéreo, mais especificamente na alocação de aeronaves em portões de embarque, bem como as formulações matemáticas do problema de alocação de aeronaves em portões de embarque que será abordado neste trabalho.

2.2.1 Agendamento

Uma vez que um dos principais objetivos do problema proposto nesta seção consiste na alocação de aeronaves em portões de embarque, o que, em essência, recai sobre um problema de agendamento (*scheduling*), na sequência, serão apresentados breves conceitos acerca do tema, com foco no setor de aviação.

Problemas de agendamento consistem na alocação de recursos para execução de tarefas em determinado intervalo de tempo. Neste sentido, há uma enorme gama de atividades onde tais tipos de problemas estão presentes, dentre as quais, pode-se citar: agendamento de navio em portos, agendamento de produção em fábricas, agendamento de etapas de produção na construção civil, agendamento de atendimento de clientes em uma fila, agendamento de despacho de produtos em centros de distribuição e, mas não somente, agendamento de alocação de aeronaves em portões de embarque.

De maneira generalista, pode-se definir problemas de agendamento como a necessidade de alocação de recursos e atividades, respeitando determinadas restrições, de maneira ótima ao longo do tempo.

Em função de suas origens na indústria, problemas de agendamento normalmente são representados através da notação de *job machine* (PINEDO; HADAVI, 1992), sendo:

- Um conjunto de n tarefas (*jobs*), que representam as atividades; e
- Um conjunto de m máquinas (*machines*) para as quais as tarefas devem ser agendadas.

O conjunto de máquinas, na verdade, pode representar qualquer tipo de recursos, tais como equipamentos, tempo, centros de distribuição ou portões de embarque.

Problemas de Programação da Produção ou *Scheduling Problems* representam uma classe de problemas de tomada de decisão muito importante na otimização. Pertencem à esta classe: sequenciamento de tarefas em máquina única, sequenciamento de tarefas em máquinas paralelas (idênticas, uniformes ou não relacionadas), *flow shop* (máquinas em série, com fluxo de tarefas unidirecional), *job shop* (máquinas em série com fluxo de tarefas não unidirecional), etc.

Dentre as possíveis funções objetivo dos *Scheduling Problems*, podemos destacar: a minimização do *makespan*, que consiste em minimizar o maior tempo de término em um conjunto de tarefas; a minimização do tempo de atraso total, que consiste determinar uma sequência, onde a soma de atrasos das tarefas seja mínima; a minimização do tempo de espera, ou seja, o tempo que a tarefa gasta desde que chega na máquina até começar a ser executada; e a minimização do adiantamento de tarefas que determina uma sequência com o menor valor do adiantamento total. Os seguintes parâmetros podem ser considerados no problema: tempo de chegada, prazo de execução, ordem de precedência das tarefas, tempo de processamento, tempo de preparação, etc (LEUNG, 2004).

De acordo com a formulação adotada (fundamentalmente dependente das restrições, parâmetros e quantidade de tarefas e máquinas) podemos resolver o problema de forma exata em tempo polinomial, ou ainda lidar com problemas da classe NP, sem conhecimento de uma solução exata para o problema em baixo tempo computacional. Nestes casos, é necessário abandonar a busca de uma solução ótima e simplesmente procurar uma solução de qualidade, através de procedimentos heurísticos.

2.2.2 Problemas de agendamento no setor aéreo

De maneira geral, os problemas de agendamento presentes no setor de aviação são ainda mais complexos do que os problemas tradicionais de agendamento. Enquanto que, normalmente, um problema de agendamento comum precisa resolver o agendamento e sequenciamento de atividades a serem executadas em uma ou poucas máquinas, no setor aéreo, os problemas de agendamento e sequenciamento precisam lidar com sistemas de problemas complexos e interdependentes. Dentre estes problemas, destacam-se a atribuição de frotas diferentes para milhares de voos, a definição de rota para cada aeronave, o agendamento de períodos de manutenção para as aeronaves e a alocação de tripulação, sendo que todos estes requisitos precisam atender legislações específicas, que, em alguns casos, podem mudar em função da região de operação da companhia aérea (QI; YANG; YU, 2004).

Os problemas de agendamento do setor aéreo são divididos em quatro grandes

categorias: programação da aeronave, agendamento da equipe, agendamento de operações irregulares em tempo real e, por último, tem-se uma categoria que é formada pela combinação de problemas de agendamento que podem ser modelados como problemas tradicionais de agendamento de máquinas (QI; YANG; YU, 2004).

A seguir, será feita uma breve discussão sobre as categorias supracitadas, de acordo com as definições apresentadas em (QI; YANG; YU, 2004).

Programação da aeronave

Problemas desta categoria preocupam-se com a atribuição de modelos de aeronaves que deverão ser destinados a uma determinada rede de voos operados por uma companhia aérea. Tais atribuições devem levar em consideração demanda de passageiros, capacidade da aeronave (de passageiros, carga e percurso), necessidades de manutenção, distância a ser percorrida, possibilidade de conexões, bem como legislações específicas. Diferenças sutis de alocações podem acarretar em grandes variações de rentabilidade financeira para as companhias aéreas.

Agendamento da equipe

Também chamado de agendamento de tripulação, seu objetivo consiste em determinar a alocação de tripulantes (pilotos e comissários de bordo) para voos individuais. Assim como o problema de agendamento de aeronave, tal problema busca minimizar o custo total de alocação, respeitando várias restrições, dentre as quais estão a preferência de determinados pilotos por determinadas rotas, o conhecimento dos pilotos sobre os tipos de aeronave da frota, o tempo de trabalho permitido por lei para a tripulação, dentre outras.

Agendamento de operações irregulares em tempo real

Comumente chamado de gerenciamento de interrupção, este problema está relacionado com as necessidades de alterações, em função de algum imprevisto, nas programações previamente estabelecidas. Nas operações diárias, é muito comum que ocorram interrupções na programação de voo, agendamento de aeronaves e agendamento de tripulação devido, em grande parte, mas não somente, às ocorrências frequentes de eventos disruptivos, como mau tempo, falha mecânica e indisponibilidade da tripulação. Dados coletados por uma *Gulf Carrier*[9] mostram que tais tipos de interrupções resultaram em 33% dos voos atrasados no período de junho a agosto de 2014, sendo que em 11% destes, o atraso foi maior do que quinze minutos (AHMED; MANSOUR; HAOUARI, 2015).

[9] *Gulf Carrier* é um termo normalmente utilizado para se referir a alguma companhia aérea baseada em algum país localizado ao longo do Golfo Pérsico. Sendo tal expressão normalmente mais utilizada para se referir às companhias Emirates, Etihad Airways e Qatar Airways.

Deste modo, é imprescindível que decida-se, em tempo real, quais as melhores alterações a serem feitas após a ocorrência de algum evento disruptivo. Entretanto, deve-se considerar que uma alteração nos planos iniciais pode acarretar em custo elevado e que os referidos planos poderão ser retomados após determinado período de tempo.

Problemas de agendamento que podem ser modelados como problemas tradicionais de agendamento de máquinas

Dentre os problemas presentes nesta categoria estão o sequenciamento de pouso da aeronave, a programação da classe de treinamento do piloto, o agendamento da força de trabalho para a entrega da bagagem, bem como o problema de agendamento da aeronave aos portões de embarque.

Em termos práticos, cada categoria apresenta uma solução para um problema de agendamento específico, sendo a solução passada para a próxima categoria como um dado de entrada do problema a ser resolvido. A necessidade por tomadas de decisão de maneira rápida, usualmente limita sequencialmente o compartilhamento de informações e a interação entre diferentes categorias, o que faz com que o resultado do planejamento normalmente recaia apenas sobre algum ótimo local (CHEN; LIU; CHOU, 2013).

2.2.3 O problema de agendamento de aeronaves em portões de embarque

As quatro principais classes de decisões pelas quais a administração de companhias aéreas é responsável são: programação da aeronave, agendamento da equipe, interrupção, bem como outros problemas de que podem ser modelados como problemas tradicionais de agendamento de máquinas. É dentro desta última que se encontram os problemas agendamento de aeronaves em portões de embarque (*Gate Assessment Problem* – GAP), um dos tópicos mais importantes e mais complexos de gerenciamento de problemas do setor aéreo.

A atribuição de aeronaves em portões de embarque deve ser conveniente tanto para as companhias áreas e administração aeroportuária quanto para os passageiros. Definições de GAP devem ser elaboradas considerando um longo horizonte de eventos, uma vez que os planejamentos devem vislumbrar os parâmetros do problema, os quais incluem, mas não se limitam, ao tamanho da aeronave e da área do portão de embarque, a capacidade de passageiros da aeronave e do portão de embarque, aos horários de pouso e decolagem das aeronaves, ao horário de funcionamento do aeroporto, as distâncias a serem percorridas pelos passageiros desde o local de *check-in* até o portão de embarque ou do local de desembarque até a área de coleta de bagagens, as condições de carregamento de bagagens, as preferências de certas companhias por determinados portões, ao tempo

mínimo entre desocupação e reocupação do *gate*. Tais restrições geram problemas com pequenas, porém importantes variações.

Tradicionalmente, problemas de alocação de aeronaves em *gates*, quando bem construídos, precisam satisfazer as duas seguintes restrições (DREXL; NIKULIN, 2008):

- Duas ou mais aeronaves não podem ser alocadas ao mesmo *gate* simultaneamente, ou seja, cada *gate* processa apenas uma aeronave por vez;
- Cada aeronave precisa ser alocada a exatamente um *gate*, isto é, uma aeronave não pode mudar de posição depois de ter sido designada a determinado *gate*.

Os objetivos típicos de GAP, apesar de não estarem limitados a estes, são os seguintes (DORNDORF et al., 2007):

- Minimização de tempo de espera da aeronave para alocação;
- Maximização de preferências de certas aeronaves por portões particulares;
- Minimização de distância a ser percorrida pelos passageiros;
- Minimização de desvios em relação à programação inicial (problemas típicos de GAP em situações de interrupção);
- Minimização de procedimentos de reboque de aeronaves.

Todos esses requisitos tornam um GAP muito complicado tanto do ponto de vista teórico quanto prático, com as características de múltiplos critérios e múltiplas restrições tornando improvável que uma solução ótima possa ser encontrada e verificada. Portanto, é preciso determinar uma solução que forneça um compromisso adequado entre todos os diferentes objetivos.

Os dados básicos de entrada para o problema de alocação de aeronaves incluem horários de chegada e partida das aeronaves, bem como especificações adicionais dos voos: a origem e destino de um voo, o tipo de aeronave, o número de passageiros, o volume de carga, preferências por *gates*, dentre outros.

Em função da grande variação de detalhes e possibilidades de modelos com focos em diferentes objetivos, há uma vasta gama de trabalhos disponíveis na literatura acerca de GAP. As áreas de pesquisa cobertas por estes trabalhos podem ser voltadas para problemas cujos objetivos são orientados aos passageiros ou aos aeroportos e companhias aéreas. Objetivos orientados aos passageiros normalmente visam a minimização da distância percorrida pelos mesmos entre as áreas de *check-in* e *gate* e entre este último e a área de coleta de bagagem. Em contrapartida, objetivos orientados aos aeroportos e companhias

aéreas, normalmente, buscam a minimização de tempo de espera para alocação de aeronaves, o que minimiza os atrasos nas decolagens, ou minimização de procedimentos de reboque de aeronaves, por exemplo (DORNDORF et al., 2007).

Dentre as abordagens que focam nos objetivos relacionados aos passageiros, pode-se citar Babić, Teodorović e Tošić (1984), que formulam um problema linear inteiro cujo objetivo é minimizar a distância total percorrida tanto pelos passageiros que estão desembarcando quanto por aqueles que ainda tomarão seus voos. Tanto Haghani e Chen (1998) quanto Bihr (1990) propõe a solução de um problema muito semelhante àquele verificado em Babić, Teodorović e Tošić (1984), sendo que Bihr (1990) usa programação linear inteira binária para resolver o problema. Haghani e Chen (1998) resolvem um GAP cujo objetivo é minimizar as distâncias de deslocamento de passageiros dentro da área do terminal, através da relaxação de uma programação linear inteira e de uma heurística, sendo que a heurística apresentou tempo de execução 113,5 vezes menor do que o tempo de execução da programação linear inteira com relaxamento. Na abordagem proposta por Xu e Bailey (2001), o objetivo do problema é minimizar os tempos totais de conexão durante os quais os passageiros caminham para pegar seus voos de conexão, sendo que o problema foi formulado como um problema de programação inteira binária quadrática e, em sequência, ajustado para um problema binário inteiro misto com função objetivo linear. A formulação de Lim, Rodrigues e Zhu (2005) utilizou como contexto um cenário onde os horários de chegada e partida de voos podem mudar e buscou minimizar as distâncias de caminhada dos passageiros.

Em Braaksma (1977) é proposta a atribuição de *gates* para encurtar as distâncias percorridas pelos passageiros nos terminais do aeroporto, com ênfase na utilização mais eficaz das instalações existentes. Analisou-se os procedimentos de alocação de *gates* em um terminal do Aeroporto Internacional de Toronto, Canadá, durante um período de vários anos e descobriu-se que a distância média de caminhada dos passageiros poderia ser reduzida em até 57% através do gerenciamento efetivo do agendamento dos portões de embarque.

Bandara e Wirasinghe (1992) têm os objetivos do problema voltados à satisfação do passageiro, porém de maneira a otimizar a construção do aeroporto. Sendo assim, propõem uma geometria de construção de aeroportos que busca minimizar a distância média percorrida por todos os passageiros. Para uma grande variedade de passageiros e números de portões, uma configuração de píer semi-centralizada mostrou-se a melhor em relação à circulação de passageiros.

Entre as publicações cujos objetivos são voltados aos aeroportos e companhias aéreas, há diversas variações nas formatações dos GAP. Tang e Wang (2013) propõem um GAP voltado para portões de embarque de uso exclusivo de determinada companhia aérea – este tipo de GAP é formulado de acordo com a perspectiva da companhia aérea e não

da autoridade aeroportuária. Wang, Zhu e Xu (2014) estabelecem um modelo que visa a atribuição de aeronaves em portões de embarque de maneira a se obter alta eficiência e um arranjo razoável; tal modelo leva em consideração características específicas de voos (tipo de voo, modelo de aeronave, o tempo de inatividade e o número de passageiros) e *gates* preferidos. Problemas cujos *gates* estão bloqueados no momento da chegada da aeronave que deveria ocupá-lo (*gate blockage problems*) são tratados por Genç et al. (2012) e por Castaing et al. (2016); tal problema surge em função da variabilidade nos horários de partida e chegada de voos e pode acarretar impactos negativos, incluindo atraso de passageiros, conexões perdidas, queima excessiva de combustível, bem como a necessidade de um completo novo agendamento das alocações de aeronaves em *gates*.

Uma das primeiras tentativas de minimizar o acontecimento de *gate blockage* foi proposta por Bolat (1999), onde foi desenvolvido um modelo matemático para alocar aeronaves respeitando um tempo mínimo de não utilização do *gate* entre uma alocação e outra. Com isto, pequenas mudanças nos dados de entrada não afetem a funcionalidade do agendamento definido. Procedimentos interativos ótimos e heurísticas foram utilizadas para resolver simulações com dados coletados no aeroporto internacional de Riade, Arábia Saudita, e o resultado mostrou grande melhoria na taxa de aeronaves remotamente atendidas e rebocadas.

Quando a partida de uma aeronave é atrasada e este atraso é significativo o suficiente para atrasar a entrada da mesma no portão ao qual foi alocada e, consequentemente atrasar a entrada da aeronave que utilizaria o *gate* em sequência, surge a necessidade de revisar as designações ao referido *gate*, de modo que sejam minimizados os tempos de atraso extras. Gu e Chung (1999) descrevem uma abordagem de Algoritmo Genético para encontrar soluções de tempo mínimo de atraso extra cujos resultados se mostraram tão ou mais eficazes do que as soluções geradas por gerentes de *gate* experientes.

Também em busca de tornar os agendamentos robustos, ou seja, à prova de pequenos desvios nos dados de entrada, Diepen et al. (2012) propõem uma abordagem que visa formular o agendamento de aeronaves com dados coletados no Aeroporto de Amsterdã Schiphol, Holanda. É apresentada uma formulação de programação linear inteira baseada em *gate plans*. Nesta proposta, cada *gate plan* consiste em um subconjunto de voos que podem ser atribuídos a um único tipo de *gate* – portões de embarque com características iguais são agregados em tipos de *gate*. O problema de atribuição de *gate* se resume a selecionar o melhor subconjunto de *gate plans*, de modo que cada voo pertença a um *gate plan* selecionado e que o número de *gate plans* selecionados para um determinado tipo de *gate* seja igual ao número de *gates* daquele tipo. Resultados computacionais com dados coletados no referido aeroporto indicam que o algoritmo proposto para a resolução da formulação é capaz de resolver instâncias reais em tempos de execução bastante pequenos.

Além das publicações cujos GAP restringem-se a um único objetivo, existem

inúmeras outras onde vários objetivos são combinados de modo a gerar soluções mais abrangentes aos problemas. Dentre as publicações analisadas, a pioneira neste tipo de abordagem, proposta por Yan e Huo (2001), mostrou um modelo binário inteiro, cujos objetivos eram a minimização da distância total percorrida pelos passageiros, bem como a minimização do tempo total de espera dos mesmos. Em função de sua importância como métrica de avaliação da qualidade de serviços prestados por aeroportos, muitas das publicações seguintes também abordaram, dentre os multiobjetivos do GAP, a minimização da distância total percorrida pelos passageiros. Foi o caso de Drexl e Nikulin (2008), que modelarem um problema inteiro, onde um dos objetivos, bem como algumas das restrições, era de ordem quadrática. Além da minimização da distância total percorrida pelos passageiros, a abordagem visou minimizar o número de voos não alocados em *gates*, bem como maximizar preferências do tomador de decisão quanto a alocação de determinadas aeronaves em *gates* específicos.

Kim, Feron e Clarke (2013), apresentaram uma abordagem que buscava tanto a minimização do tempo total de caminhada dos passageiros, quanto a minimização do tempo de taxiamento da aeronave na rampa de acesso ao *gate*, ponderado pela quantidade de passageiros a bordo. Como resultado, o fluxo no terminal de passageiros e nas rampas mostrou-se mais eficiente.

Assim como em Yan e Huo (2001), os dois objetivos do problema abordado em Jiang, Zeng e Luo (2013) possuem os passageiros como protagonistas. No problema apresentado, o modelo de designação de *gate* é baseado na minimização da distância total percorrida pelos passageiros e no balanceamento da distância média de caminhada dos passageiros entre as diferentes companhias aéreas que prestam serviço no aeroporto.

Marinelli, Dell'Orco e Sassanelli (2015) e Dell'Orco, Marinelli e Altieri (2017) apresentam uma abordagem em que um dos objetivos também recai sobre a minimização da distância total percorrida pelos passageiros. O outro objetivo a compor o problema, no entanto, é a minimização da quantidade de aeronaves alocadas ao *apron* (*remote gate*). Quando as aeronaves são alocadas em portões de embarque comuns, normalmente, os passageiros deslocam-se ao terminal por meio de conectores móveis, chamados de ponte de jato ou ponte de embarque de passageiros. Porém, quando a aeronave precisa ser alocada ao *apron*, normalmente em função de indisponibilidade de *gate*, o deslocamento de passageiros até o terminal, na maioria das vezes, é feito por ônibus das companhias aéreas ou do próprio aeroporto, o que incorre em custos financeiros adicionais e em eventuais prejuízos de reputação para as companhias aéreas e aeroportos. A resolução do problema apresentado deu-se por uma metaheurística baseada na metodologia de *Bee Colony Optimization*; os resultados da comparação com o agendamento de horários reais utilizados no aeroporto de Milão-Malpensa, na Itália, destacaram a eficácia do método proposto.

A movimentação de maneira eficiente das aeronaves em torno do terminal afeta

diretamente a rentabilidade de um voo específico, e consequentemente, de uma companhia aérea como um todo, haja que vista que a redução de atrasos, aliada a movimentações eficientes, faz com que os motores permaneçam acionados por menos tempo, o que poupa a queima de combustível, que é um insumo de elevado custo. Isto posto, Behrends e Usher (2016), propuseram um modelo que busca minimizar atrasos tanto em função de taxiamento ineficiente quanto provenientes de movimentação de cargas e pessoas em direção até a aeronave.

Zhang, Xue e Jiang (2017), buscam gerar um sistema de agendamento robusto ao propor um modelo que visa minimizar tanto a chance de conflito entre aeronaves, como a alocação das mesmas ao *apron*. A chance de conflito entre duas aeronaves é dada por uma função de probabilidade que indica a possibilidade de uma aeronave ter algum problema de emergência e precisar utilizar o *gate* por maior tempo do que o previsto e, com isto, acabar por gerar conflito com a aeronave alocada em sequência.

A abordagem proposta por Daş (2017) mostrou um objetivo inédito até então nas formulações de GAP multiobjetivo. Ao passo que a maior parte das abordagens propunham objetivos de minimizações de aeronaves não alocadas, minimização de aeronaves alocadas ao *apron*, minimização de atrasos de taxiamento, dentre outros objetivos relacionadas às movimentações em terra da aeronave, o estudo de Daş (2017) propôs um problema cujo objetivo geral dos modelos construídos era maximizar o número de passageiros nos portões próximos aos estabelecimentos comerciais internos do aeroporto e minimizar a distância total percorrida pelos mesmos, de modo que estes tivessem mais tempo livre para fazer compras dentro do terminal, afim de aumentar as receitas ao administrador aeroportuário.

A Tabela 2 resume os objetivos abordados nas publicações multiobjetivos analisadas nesta seção. As letras de A até I, indicam, respectivamente, as seguintes publicações: Yan e Huo (2001), Drexl e Nikulin (2008), Kim, Feron e Clarke (2013), Jiang, Zeng e Luo (2013), Marinelli, Dell'Orco e Sassanelli (2015), Behrends e Usher (2016), Zhang, Xue e Jiang (2017), Daş (2017) e Dell'Orco, Marinelli e Altieri (2017). Dentre as nove publicações analisadas, em sete delas o objetivo que busca minimizar a distância de deslocamento de passageiros esteve presente, evidenciando sua importância em GAP; o segundo objetivo mais presente foi a minimização de aeronaves alocadas ao *apron* (*remote gate*); a minimização de atrasos em função do taxiamento da aeronave apareceu em duas publicações, enquanto que os demais objetivos apareceram uma única vez.

Tabela 2 – Objetivos abordados nas publicações multiobjetivos analisadas nesta seção.

Objetivos	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
Minimizar distância de deslocamento de passageiros.	X	X	X	X	X			X	X
Minimizar tempo que os passageiros esperam para embarcar/desembarcar.	X								
Minimizar quantidade de aeronaves não alocadas em *gates*.		X							
Maximizar preferências de alocações do tomador de decisão.		X							
Minimizar atrasos em função de taxiamento da aeronave.			X			X			
Minimizar diferença entre tempos de deslocamentos de passageiros de diferentes companhias aéreas.				X					
Minimizar alocação de aeronaves ao *apron* (*remote gate*).					X		X		X
Minimizar atrasos em função da movimentação de cargas até a aeronave.						X			
Minimizar conflito de aeronaves alocadas ao mesmo *gate*.							X	X	
Maximizar gastos de passageiros com compras dentro das lojas do aeroporto.									X

Fonte: Produzido pelo autor.

2.2.4 O GAP abordado neste trabalho

Conforme é mostrado na subseção 2.2.3, a distância percorrida pelo passageiro dentro do aeroporto é uma importante medida da qualidade do serviço prestado por este último. Isto se torna evidente ao passo que, dentre as nove publicações que abordam GAP com formulações multiobjetivo analisadas nesta dissertação, sete contém o objetivo de minimizar as distâncias percorridas pelos passageiros. Em função disto, tal objetivo estará presente na formulação de GAP apresentado nesta seção.

Além da minimização de distâncias percorridas pelos passageiros, outro objetivo muito importante consiste na minimização de alocações de aeronaves ao *apron*, pois, como descrito anteriormente, alocações em *gates* remotos são prejudiciais tanto aos passageiros, pois acarretam em maiores tempos de viagem, quanto ao aeroporto e companhias áreas, ao passo que estes têm incremento em seus custos operacionais e também podem sofrer danos de reputação. Isto posto, outro objetivo presente na formulação de GAP apresentado neste trabalho é a minimização de aeronaves alocadas ao *apron*.

Alocações eficientes, além de atenderem aos objetivos citados, devem evitar que haja atrasos nas decolagens, pois estes podem repercutir em toda a cadeia de agendamento, o que é prejudicial tanto para as companhias aéreas e administração aeroportuária, quanto para os passageiros. Sendo assim, um outro objetivo presente no GAP abordado nesta dissertação consiste em minimizar a soma dos atrasos de decolagem das aeronaves.

Com isto, os objetivos do GAP onde será aplicado o Algoritmo Genético

proposto consistem em:

- Minimizar a soma dos atrasos de decolagem das aeronaves;
- Minimizar o número de aeronaves alocadas ao *apron*;
- Minimizar a distância total percorrida pelos passageiros.

Dados estes objetivos e as características do problema abordado, propõe-se uma formulação de Programação Linear Inteira, baseada em aspectos de problemas de *scheduling*. Tal formulação é apresentada na sequência.

Parâmetros considerados:

M: conjunto de *gates* (portões de embarque).

N: conjunto de aeronaves.

m: número total de *gates*, onde M indica o *apron*.

n: número total de aeronaves.

a_i: tempo de chegada (*arrival*) da aeronave i.

d_i: tempo de partida (*departure*) da aeronave i.

p_i: tempo de processamento da aeronave i (reportado na literatura como sendo, em média, 30 minutos, variando em função da capacidade da aeronave).

wd_j: distância entre a área de *check-in* e o *gate* j.

wa_j: distância entre o *gate* j e a área de coleta de bagagem após o desembarque.

qd_i: quantidade de passageiros que embarcarão na aeronave i.

qa_i: quantidade de passageiros que desembarcarão da aeronave i.

L: número inteiro tão grande quanto se queira.

Através da analogia de máquinas com *gates* e tarefas com aeronaves, definiu-se as seguintes variáveis de decisão:

$$z_{ij} = \left\{ \begin{array}{ll} 1 & \text{se a aeronave } i \text{ está alocada no } \textit{gate } j \text{ (incluindo o } \textit{apron}) \\ 0 & \text{caso contrário} \end{array} \right\}$$

$$w_{ik} = \left\{ \begin{array}{ll} 1 & \text{se a aeronave } i \text{ é alocada antes da aeronave } k \\ 0 & \text{caso contrário} \end{array} \right\}$$

w_{ik} pode ser um ou zero quando as aeronaves i e k são alocadas ao mesmo *gate* j, mas é sempre zero quando i e k são alocados em gates distintos.

x_i: início de processamento da aeronave i.

As formulações dos objetivos do problema, bem como suas restrições, são apresentadas a seguir:

$$\min \quad f_1 = \sum_{i=1}^{N} max\{0,\ x_i + p_i - d_i\} \tag{2.7}$$

$$\min \quad f_2 = \sum_{i=1}^{N} z_{iM} \tag{2.8}$$

$$\min \quad f_3 = \sum_{i=1}^{N}\sum_{j=1}^{M} wd_j qd_i z_{i,j} + \sum_{i=1}^{N}\sum_{j=1}^{M} wa_j qa_i z_{i,j} \tag{2.9}$$

$$\text{s.a.:} \quad \sum_{j=1}^{M} z_{ij} = 1 \quad \forall\ i \tag{2.10}$$

$$z_{ij} + z_{kj} - w_{ik} - w_{ki} \leqslant 1 \quad \forall\ i \neq k,\ j \neq m+1 \tag{2.11}$$

$$x_i + p_i \leqslant x_k + (1 - w_{ik})L \quad \forall\ i, k \tag{2.12}$$

$$z_{ij} + z_{kh} + w_{ik} + w_{ki} \leqslant 2 \quad \forall\ i \neq k,\ j \neq h \tag{2.13}$$

$$x_i \geqslant a_i \quad \forall\ i. \tag{2.14}$$

$$w_{ik}, z_{ij} \in \{0, 1\} \quad \forall\ i, k, j \tag{2.15}$$

$$x_i \in \mathbb{Z} \quad \forall\ i \tag{2.16}$$

Onde:

- f_1: Minimização da soma dos atrasos de decolagem das aeronaves;
- f_2: Minimização do número de aeronaves alocadas ao *apron*;
- f_3: Minimização da distância total percorrida pelos passageiros.

A Equação 2.10, a Equação 2.11, a Equação 2.12, a Equação 2.13 e a Equação 2.14 são baseadas no conteúdo apresentando por Barbosa (2014).

A Equação 2.10 estabelece que cada aeronave será alocada em apenas um *gate*. A Tabela 3, a seguir, simula a restrição da Equação 2.10 para o caso em que $M = 3$ e $N = 4$.

Na simulação mostrada na Tabela 3, a soma das linhas (combinação da alocação entre aeronave i e *gate* j) precisa ser igual a um, o que garante que uma aeronave será alocada e que isto ocorrerá apenas uma vez. A linha que representa $i = 1$ indica que a

Tabela 3 – Simulação da Equação 2.10.

$i \backslash j$	**1**	**2**	**3**	$m+1$
1	1	0	0	0
2	1	0	0	0
3	0	0	1	0
4	0	0	0	1

Fonte: Produzido pelo autor.

aeronave um está alocada ao *gate* um ($j = 1$), enquanto que a aeronave quatro ($i = 4$) foi alocada ao *apron*, por exemplo.

Se as aeronaves i e k forem alocadas ao mesmo *gate*, então i deve ser processada antes de k, ou vice versa (Equação 2.11). Com isto, garante-se que haverá alocação de apenas uma aeronave por vez em cada *gate*. Segue, na Tabela 4, uma representação da funcionalidade da Equação 2.11.

Tabela 4 – Exemplo de situações da Equação 2.11.

Situação	$z_{i,\,j}$	$+$	$z_{k,\,j}$	$-$	$w_{i,\,k}$	$-$	$w_{k,\,i}$	$\leqslant$	**1**
1	0	$+$	0	$-$	0	$-$	0	$\leqslant$	0
2	1	$+$	0	$-$	0	$-$	0	$\leqslant$	1
3	1	$+$	1	$-$	1	$-$	0	$\leqslant$	1

Fonte: Produzido pelo autor.

Descrição das situações representadas na Tabela 4:

- **Situação 1**: nenhuma aeronave é alocada aos *gates* – restrição atendida com somatório igual a zero;
- **Situação 2**: aeronave i é alocada ao *gate* j e aeronave k não é alocada ao referido *gate* – restrição atendida com somatório igual a um;
- **Situação 3**: aeronaves i e k são alocadas ao *gate* j, porém, a aeronave i é processada antes da alocação da aeronave k ao referido *gate* – restrição atendida com somatório igual a um.

A Equação 2.12 não permite que aeronaves diferentes sejam alocadas ao mesmo tempo em um único *gate*.

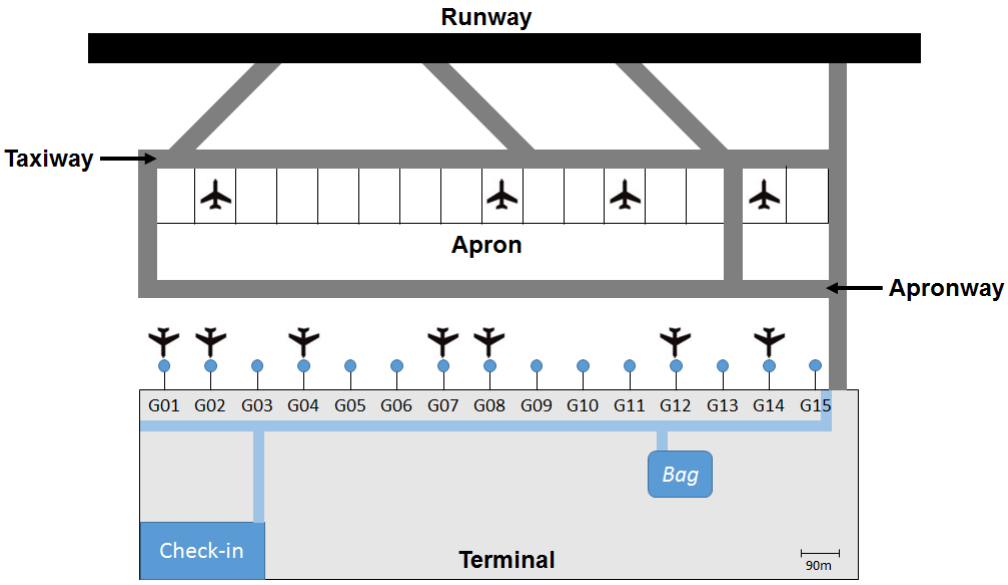
- Se i e k não são alocadas ao mesmo *gate*, w_{ik} tem valor zero, fazendo com que o lado direito da Equação 2.12 seja tão grande quanto se queira;
- Se i e k são alocadas ao mesmo *gate*, então w_{ik} pode ter valor um e, consequentemente, o lado direito da equação será representado pelo valor de x_k, que deverá ser maior ou igual que a soma de x_i com p_i.

Quando as aeronaves i e k são alocadas em diferentes *gates*, a variável w_{ik} é sempre zero (Equação 2.13). A Equação 2.14 não permite que as aeronaves sejam alocadas antes que cheguem ao aeroporto. A Equação 2.15 garante que as variáveis de decisão w_{ik} e z_{ij} sejam binárias, enquanto que a Equação 2.16 garante que a variável de decisão x pertença ao conjunto dos números inteiros.

Simultaneidade pode ocorrer para aeronaves alocadas ao *apron*, pois este comporta mais de uma aeronave (para o problema formulado neste trabalho, considera-se que o *apron* possui capacidade ilimitada).

A Figura 6, adaptada de Zhang, Xue e Jiang (2017), representa um aeroporto típico das instâncias que serão abordadas no GAP desta dissertação.

Figura 6 – Representação esquemática de um aeroporto.



Fonte: adaptado de (ZHANG; XUE; JIANG, 2017).

Para efeito de simplificação do modelo e dos cálculos, não são consideradas a existência de aeronaves que precisem ser alocadas somente em determinados portões de embarque devido a questões físicas como envergadura (distância entre as pontas das asas) e comprimento. Em alguns casos, quando a aeronave apresenta elevada envergadura, pode ser necessário, dependendo das configurações do aeroporto, que os portões de embarque adjacentes àquele onde tal aeronave foi alocada não sejam ocupados enquanto tal aeronave é processada. Além disso, não considerou-se o tempo mínimo de intervalo que um *gate* deve ficar livre entre a acomodação de duas aeronaves e, por fim, adotou-se que há apenas uma área de *check-in* e uma de coleta de bagagem.

Apenas tomando por base representação esquemática do aeroporto, conforme mostrado na Figura 6, pode-se deduzir que os objetivos do problema estudado são confli-

tantes. A otimização da Equação 2.7 poderia ser atingida caso todas as aeronaves fossem alocadas ao *apron*, pois o tempo de espera para alocação de aeronaves tenderia à zero, uma vez que a capacidade do *apron* pode ser considerada virtualmente ilimitada, fazendo com que não houvessem atrasos nas decolagens. Porém, tal cenário acarretaria, provavelmente, na pior solução possível para Equação 2.8, a qual busca minimizar a alocação de aeronaves ao *apron*. Pode-se notar também que, por exemplo, a minimização da Equação 2.9, provavelmente demandaria grande concentração de alocação de aeronaves nos *gates* 3 e 12, os quais são mais próximos, respectivamente, das áreas de *check-in* e coleta de bagagem, porém, tais alocações acabariam por não favorecer a minimização da Equação 2.7.

Os dados referentes aos horários previstos de chegada e partida de aeronaves, bem como a quantidade de *gates* e a distância entre os mesmos e as áreas de *check-in* e coleta de bagagem são baseados no artigo denominado *Multi-objective gate assignment based on robustness in hub airports*, elaborado por Zhang, Xue e Jiang (2017). Os dados de quantidades de passageiros (ocupação de voos), bem como a capacidade média das aeronaves consideradas no problema, foram baseados em dados históricos disponíveis em ANAC (2017). Todos os dados utilizados nas execuções do Algoritmo Genético podem ser consultados no Apêndice A.

2.2.4.1 Nível de complexidade de resolução do GAP abordado neste trabalho

Uma parte dos problemas encontrados são de fácil solução e são classificados na Teoria da Complexidade como problemas de classe P (*Polinomial Time*), pois podem ser solucionados por meio de algoritmos determinísticos, cuja complexidade é determinada por um polinômio $p(n)$. Entretanto, em problemas onde não é conhecido um algoritmo que encontre uma solução em tempo polinomial, diz-se que os mesmos pertencem à classe NP (*Non Deterministic Polinomial Time*), sendo considerados de difícil resolução.

Problemas de otimização combinatória com apenas um objetivo são considerados NP-difícil. Porém, em problemas de otimização combinatória com múltiplos objetivos, a complexidade computacional envolve uma componente relacionada com a contagem do número de soluções do problema, fator que aumenta o custo computacional. Além disso, conforme demonstrado por Ullman (1975), todo problema de encontrar um cronograma ótimo para um conjunto de tarefas (*scheduling*) é um problema NP-difícil. Logo, o GAP abordado nesta dissertação é caracterizado como um problema NP-difícil.

3 Algoritmo Genético proposto

Neste capítulo será apresentado um Algoritmo Genético cujo objetivo é ser genérico o suficiente para ser utilizado em diversos tipos de abordagens de Otimização Multiobjetivo. O referido algoritmo baseia-se em alguns aspectos presentes no *Nondominated Sorting Genetic Algorithm II*, proposto por Deb et al. (2000), uma vez que este último obteve sucesso no tratamento de diversos tipos de problemas relatados em literatura. Tais aspectos são as funções de *ranking* (classificação por não dominância) e *crowding distance* (distância de aglomeração) e, assim como o NSGA-II, o algoritmo aqui apresentado considera explicita e simultaneamente diferentes funções objetivo, encontrando um conjunto de soluções não dominadas no final de uma execução.

Para a utilização do algoritmo proposto, admite-se que não são conhecidas informações de preferências entre os objetivos, de forma que o algoritmo deve gerar alternativas para a posterior escolha do decisor. Portanto, algumas metas importantes em otimização multiobjetivo devem ser atingidas, as quais são:

- Encontrar um conjunto de soluções que esteja o mais próximo possível do Conjunto Pareto-ótimo;
- Encontrar um conjunto de soluções com a maior diversidade possível no espaço dos objetivos;
- Encontrar um conjunto de soluções com a maior diversidade possível no espaço das variáveis.

Os componentes e as principais funcionalidades do Algoritmo Genético, considerando sua adaptação ao problema de estudo de caso que se deseja resolver, são descritos nas seções seguintes.

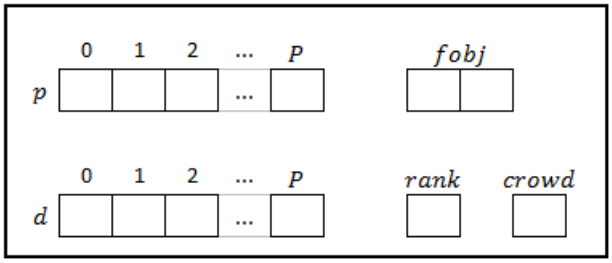
3.1 Algoritmo Genético proposto ao problema de alocação de capital em empresas do setor financeiro

Esta seção aborda os principais componentes presentes no Algoritmo Genético proposto para resolver o problema de alocação de capital em empresas do setor financeiro, descrito na subseção 2.1.1.

3.1.1 Codificação

A codificação do indivíduo está representada na Figura 7. Cada indivíduo é composto por um vetor chamado p, que define o percentual do capital total disponível, C, que será alocado em cada produto i, inclusive o percentual que não será alocado. Além de p, cada indivíduo possui outro vetor denominado d, onde cada elemento representa a demanda máxima de clientes pelo produto i. Além destes, os indivíduos possuem ainda um vetor $fobj$, onde são armazenados os valores de cada objetivo do problema e os vetores unitários $rank$ e $crowd$, os quais indicam, respectivamente, o $rank$ e a distância de aglomeração do indivíduo.

Figura 7 – Representação esquemática do indivíduo do problema de alocação no setor bancário.



Fonte: autoria própria.

3.1.2 Inicialização

No processo de inicialização da população, cada elemento i do vetor p de cada indivíduo é preenchido com um número aleatório no intervalo [0, $percentMax$], onde $percentMax$ indica o máximo percentual de C necessário para atender a demanda total existente pelo produto i. $percentMax$ é definido da seguinte maneira:

$$percentMax_i = \frac{cr_i d_i r_i}{C} \tag{3.1}$$

A inicialização no intervalo [0, $percentMax$] auxilia no processo de busca de soluções, pois atribui um conhecimento prévio do formato da solução do problema na inicialização da população.

3.1.3 Avaliação da população

A avaliação da população inicia-se pela verificação dos valores atribuídos aos elementos do vetor p dos indivíduos. Para cada indivíduo, caso a somatória dos elementos de p seja diferente de 1, significa que há capital que não foi alocado ($\sum_{i=1}^{P} p_i < 1$) ou que há

mais capital alocado do que C ($\sum_{i=1}^{P} p_i > 1$). Caso a referida somatória seja diferente de 1, é então feita uma normalização na atribuição de p_i para que 100% de C seja alocado (não alocar capital, também é uma possibilidade e é representada pela alocação no produto p_P, o qual possui zero demanda).

Além de alocar capital no produto p_P ser uma opção, o mesmo também recebe capital quando não há mais possibilidade de alocar capital em algum produto qualquer em função da alocação já existente ser suficiente para atender 100% da demanda deste produto - caso típico em que C é mais do que suficiente para atender toda a demanda.

Após estas etapas, são calculados os valores das funções objetivo e a avaliação da população é encerrada.

3.1.4 Abordagem de classificação por não dominância

O processo de seleção dos melhores indivíduos utilizou-se da abordagem proposta por Deb et al. (2000), descrita em detalhes na subseção 1.2.3.3. De maneira geral, entre duas soluções em fronteiras de dominância distintas, prefere-se aquela cujo *rank* é menor. Em casos onde duas soluções estão na mesma fronteira de dominância, ou seja, o *rank* de uma é igual ao da outra, opta-se por aquela com o maior *crowding distance*, pois significa que tal solução está localizada em uma região de menor densidade, o que confere maior variabilidade para a população.

Nesta etapa não há necessidade de penalizar soluções infactíveis, pois o processo de avaliação não permite a existência de tais tipos de soluções.

3.1.5 *Crossover*

Uma população de pais é construída através de um torneio binário (duas soluções aleatórias da população competem de acordo com os critérios *rank* e *crowding distance* e a melhor delas pertencerá à população de pais). Pares de pais desta população são selecionados para gerar pares de filhos, cujas características são recebidas dos pais de acordo com a aplicação do operador de *crossover*. Tal operador é aplicado com uma probabilidade PC para cada par de soluções selecionadas. A operação de *crossover* ocorre da seguinte maneira:

$$filho_1 = \beta * pai_1 + (1 - \beta) * pai_2$$

$$filho_2 = (1 - \beta) * pai_1 + \beta * pai_2$$

Onde β é um número aleatório no intervalo [0, 1] que define o percentual de alelos que serão modificados nos pais.

O único elemento do indivíduo modificado pela operação de *crossover* é a variável de decisão p.

3.1.6 Mutação

O operador de mutação pode ou não ser acionado, dependendo da probabilidade de mutação, PM, associada ao mesmo. A mutação é realizada nos indivíduos finais do cruzamento (os dois melhores entre pais e filhos). Para cada indivíduo da população é sorteado um número entre 0 e 1 e, caso o número atrelado ao indivíduo seja maior ou igual ao valor de PM, o indivíduo passa pelo processo de mutação, o qual, neste caso, altera somente a variável p, de modo que possam ocorrer mudanças no percentual de C atribuído à cada produto i. Foram implantados dois tipos de mutação, os quais são a Mutação Uniforme e a Mutação Não Uniforme, explicados a seguir.

Mutação Uniforme

A Mutação Uniforme considera uma direção de mutação d, aleatória, pertencente ao intervalo $[-1, 1]$. O tamanho do passo α é um valor aleatório e pertence ao intervalo $[0, \frac{p}{10}]$, ou seja, a mutação aumenta ou diminui o valor de p em até 10% de seu valor atual.

Mutação Não Uniforme

A Mutação Não Uniforme foi especialmente desenvolvida por Michalewicz (2013) para problemas de otimização com restrição e codificação em ponto flutuante e é um operador dinâmico cujo objetivo é melhorar o processo de busca. Uma vez que determinado indivíduo é selecionado para o processo de mutação, o novo valor de seu elemento p pode ser definido como p' da seguinte maneira:

$$p' = \left\{ \begin{array}{l} p + \Delta(G, p) \text{ ou} \\ p - \Delta(G, p) \end{array} \right\}$$

A função $\Delta(G, p)$ retorna um valor no intervalo $[0, p]$ de modo que a probabilidade de $\Delta(G, p)$ ser próximo de zero é maior à medida em que G (número de gerações) aumenta. Esta propriedade faz com que este operador inicialmente explore o espaço de busca de forma mais ampla nas gerações iniciais, e localmente, em gerações avançadas. Michalewicz (2013) propõe a seguinte função: $\Delta(G, y) = y(1 - r^{(\frac{1-G}{G_f})^5})$, onde G_f é o número da última geração e r é um número aleatório no intervalo $[0, 1]$.

3.1.7 Esquema de funcionamento do Algoritmo Genético proposto

De maneira geral, o Algoritmo Genético proposto para o problema de alocação de capital em empresas do setor financeiro funciona da seguinte maneira:

{
    decodifica (população);
    inicializa (população);
    avalia (população);
    para $t = 1$ até G
    {
        seleciona pais (população);
        crossover (população);
        mutação (população);
        avalia (população);
    }
}

3.2 Algoritmo Genético proposto ao problema de alocação de aeronaves em portões de embarque

Esta seção aborda os principais componentes presentes no Algoritmo Genético proposto para resolver o problema de alocação de aeronaves em portões de embarque, descrito na subseção 2.2.4.

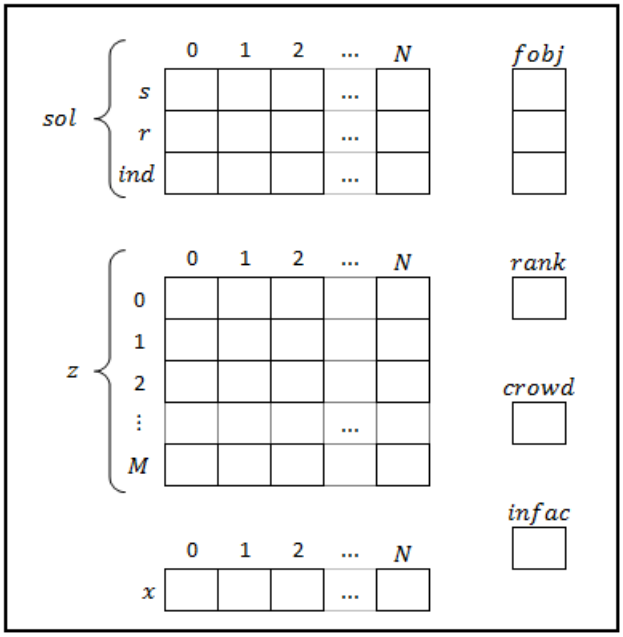
O conteúdo da subseção 3.2.1 e da subseção 3.2.3, apresentadas a seguir, é inspirado em implantações semelhantes feitas em Barbosa (2018), para a resolução de um problema de *scheduling*.

3.2.1 Codificação

A codificação do indivíduo está representada na Figura 8. Cada indivíduo é composto por uma matriz denominada *sol*, uma matriz z que indica se a aeronave i está alocada ao *gate* j, um vetor x que mostra o horário de início de processamento da aeronave i, um vetor $fobj$, onde são armazenados os valores de cada objetivo do problema e pelos vetores unitários $rank$, $crowd$ e $infac$, os quais indicam, respectivamente, o *rank*, a distância de aglomeração e a infactibilidade (existente ou não) do indivíduo.

A matriz *sol* é uma estrutura auxiliar para o preenchimento das variáveis x e z, cuja quantidade de colunas é determinada pelo número de aeronaves N do problema. Nesta estrutura, a primeira linha, s, é composta por um número real, no qual a parte inteira indica em qual *gate* a aeronave será alocada e a parte decimal define qual aeronave terá preferência na alocação do referido *gate* caso duas ou mais aeronaves sejam direcionadas ao mesmo (neste caso, a aeronave com maior valor na parte decimal de s será alocada antes daquelas cujos valores da parte decimal sejam menores). A segunda linha da referida matriz, denominada r, armazena o valor da parte decimal de s, enquanto que a terceira linha, *ind*, é referente ao índice de cada aeronave.

Figura 8 – Representação esquemática do indivíduo do problema de GAP.



Fonte: autoria própria.

3.2.2 Inicialização

A inicialização é feita de maneira aleatória, com números no intervalo [0, M], com o preenchimento de s variando de de zero até N. Além de s, neste momento também é atribuído o índice de cada aeronave ao *ind*.

O tamanho da população, TP, foi definido empiricamente com base em experimentos prévios. A população foi composta por cem indivíduos, pois durante os testes para verificação de resultados, notou-se que incrementar indivíduos na população, além do valor citado, não trazia alterações nas características das soluções encontradas.

O processo de inicialização é seguido pela avaliação da solução. Nesta parte, *sol* é decodificada para que seja feito o preenchimento de x e z. Além disso, em seguida, é realizado um processo que busca diminuir a ociosidade de *gates* gerada na solução inicial, seguido de uma nova avaliação da solução então obtida.

Os processos de avaliação e diminuição de ociosidade são explicados a seguir.

3.2.3 Avaliação da população

A avaliação da população tem início com a decodificação de *sol* para que sejam preenchidos x e z. A matriz z é preenchida em função da parte inteira de s: caso $s = 7{,}4$ e $ind = 5$, por exemplo, significa que a aeronave 6 (ind começa em zero) foi, inicialmente, alocada no *gate* 7 e, deste modo, a posição de z que representa a aeronave 6 e o gate 7 receberá o valor 1.

Para o preenchimento de x, é necessário que seja atribuído o valor decimal de s à r e que a matriz *sol* seja ordenada de maneira crescente, de modo que se possa verificar qual a ordem de preferência de atendimento das aeronaves que estão alocadas no mesmo *gate*. Isto feito, é então comparado, para cada *gate* e para cada aeronave alocada ao mesmo, quais os horários de chegada das aeronaves e quais os horários em que o *gate* está livre.

A primeira aeronave alocada em cada *gate* sempre poderá ser processada no momento imediato de sua chegada, uma vez que, neste caso, o *gate* sempre estará livre. Sendo assim, em tais casos, é atribuído, ao vetor x, o horário de chegada da aeronave. Para as demais aeronaves é necessário verificar se o *gate* estará livre no momento em que as mesmas estão previstas para chegar. Caso a aeronave chegue antes do momento em que o *gate* estará livre, a mesma deve aguardar e x receberá o valor do momento em que o *gate* estiver livre. Em casos onde a aeronave chegue e o *gate* já esteja livre, a primeira é prontamente processada e x recebe o valor da previsão de chegada da mesma.

Uma vez preenchido x, é verificado se o horário de decolagem da aeronave i será respeitado ($x_i + p_i \leqslant d_i$). Além disto, os valores dos objetivos do problema são calculados.

3.2.4 Diminuição de ociosidade

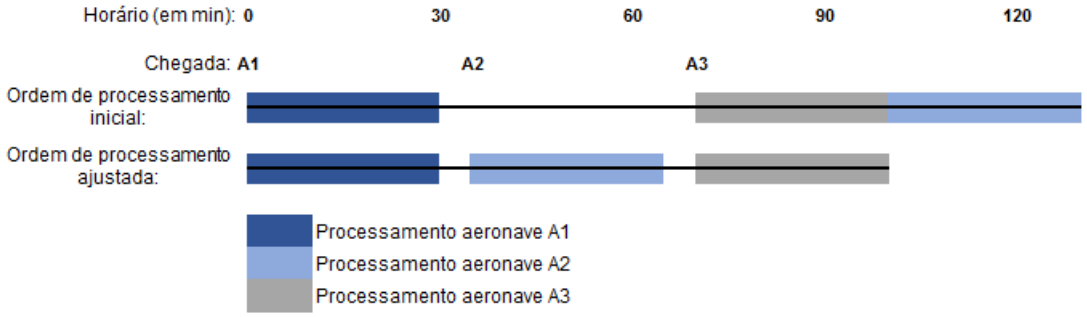
Com o objetivo de agilizar a busca pelo conjunto de soluções, este processo é responsável por aprimorar a solução da inicialização. Para tanto, procura-se adiantar o horário previsto para o processamento de aeronaves agendadas para determinado *gate*, em função de horários ociosos deste último.

De maneira geral, o processo de diminuição de ociosidade verifica quais aeronaves foram atribuídas em determinado *gate* e compara a ordem de processamento estabelecida com os horários de chegada previstos para mesmas e, caso entre o horário agendado para determinada aeronave ser processada e o horário de chegada desta última exista um tempo

ocioso no *gate* ao qual a mesma foi alocada e tal período de ociosidade seja suficiente para que esta aeronave seja processada, o horário de processamento da mesma é então adiantado, diminuindo, deste modo, o ociosidade do *gate* em análise.

A Figura 9 mostra um exemplo deste processo em um *gate* onde foram alocadas três aeronaves (A1, A2 e A3). Neste exemplo, a aeronave A2 está agendada para ter seu processamento iniciado no minuto 100. Porém, tal aeronave chega no aeroporto no minuto 35 e, entre os minutos 30 e 70, o *gate* onde A2 foi alocada está ocioso, sendo este tempo de ociosidade suficiente para o processamento da aeronave (30 minutos). Deste modo, A2 tem seu horário de início de processamento ajustado para o minuto 35, diminuindo, então, a ociosidade do *gate* ao qual tal aeronave foi alocada.

Figura 9 – Representação esquemática do processo de diminuição de ociosidade de determinado *gate*.



Fonte: autoria própria.

3.2.5 Abordagem de classificação por não dominância

O processo de seleção dos melhores indivíduos do GAP segue o mesmo mecanismo apresentado na subseção 1.2.3.3.

Porém, para o GAP, fez-se simulações onde $x_i + p_i > d_i$ (ou seja, soluções que prejudicam o objetivo da Equação 2.7) são penalizadas e recebem maiores *ranks* e onde tais soluções não são penalizadas. As comparações entre as simulações são abordadas no Capítulo 4.

3.2.6 *Crossover*

O processo de construção da população de pais que poderão passar pelo *crossover*, em função de uma probabilidade PC, é o mesmo aplicado ao problema de alocação de capital em empresas do setor financeiro. Porém, neste caso, a operação de *crossover* ocorre da seguinte maneira:

Para a primeira metade $(\frac{N}{2})$ dos elementos de s e ind de cada indivíduo da população:

$$filho_1 = pai_1$$

$$filho_2 = pai_2$$

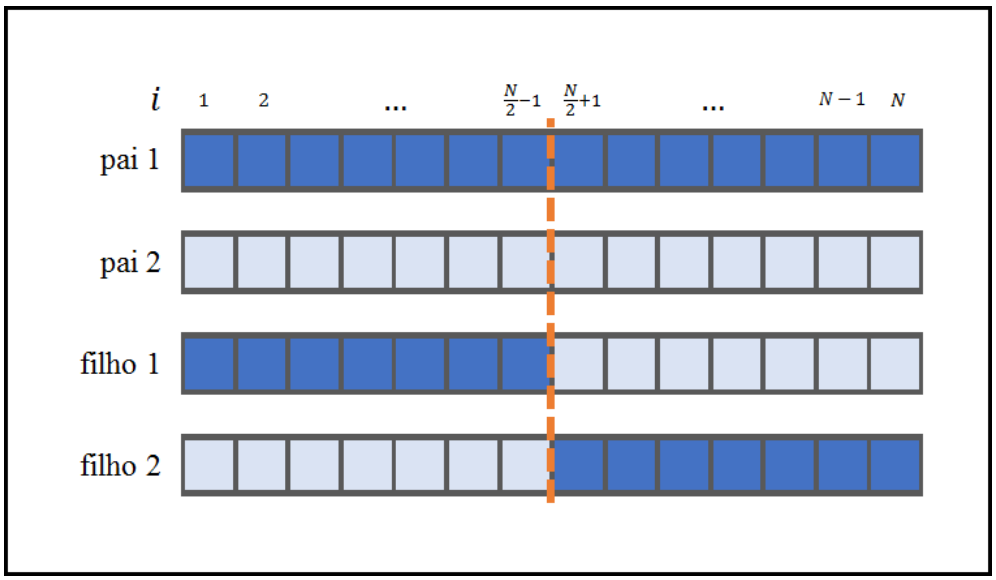
Para a segunda metade $(\frac{N}{2})$ dos elementos de s e ind de cada indivíduo da população:

$$filho_1 = pai_2$$

$$filho_2 = pai_1$$

A figura Figura 10 é uma representação esquemática do processo de *crossover* utilizado no GAP.

Figura 10 – Representação esquemática do *crossover* do problema de GAP.



Fonte: autoria própria.

As únicas partes do indivíduo que passam pelo *crossover* são s e ind, uma vez que é a partir delas que os demais membros do indivíduo são gerados.

3.2.7 Mutação

Assim como para o problema de alocação de capital em empresas do setor financeiro, foram utilizados os operadores de Mutação Uniforme e Mutação Não Uniforme, porém, neste caso, o parâmetro alvo das mutações foi o elemento de s dos indivíduos selecionados, de acordo com a probabilidade de mutação PM.

3.2.8 Esquema de funcionamento do Algoritmo Genético proposto

De maneira geral, o Algoritmo Genético proposto para GAP funciona da seguinte maneira:

{

decodifica (população);

inicializa (população);

avalia (população);

reduz ociosidade (população);

avalia (população);

para $t = 1$ até G

{

seleciona pais (população);

crossover (população);

mutação (população);

avalia (população);

}

}

4 Experimentos e Resultados

Os problemas para estudo de caso, descritos no Capítulo 2, foram resolvidos variando-se a quantidade de gerações do Algoritmo Genético proposto no Capítulo 3. As resoluções ocorreram em cenários com $G = 100$, $G = 300$, $G = 500$, $G = 800$ e $G = 1.000$, onde G é quantidade de gerações. Em todos os casos, utilizou-se uma população com cem indivíduos. Além disso, para cada problema de estudo de caso, em cada variação de G, executou-se trinta vezes o Algoritmo Genético.

Nas seções seguintes deste capítulo estão descritos os resultados obtidos nos problemas de estudo de caso.

4.1 Problema de alocação de capital em empresas do setor financeiro

Esta seção mostra os resultados obtidos nas execuções do Algoritmo Genético adaptado ao problema de alocação de capital em empresas do setor financeiro. Mostra-se a distribuição das fronteiras de soluções não dominadas em função do número de gerações com o qual o Algoritmo Genético é executado, bem como as características de alocação de capital presentes nas soluções onde foram encontrados os melhores valores para a Equação 2.1 e para a Equação 2.2.

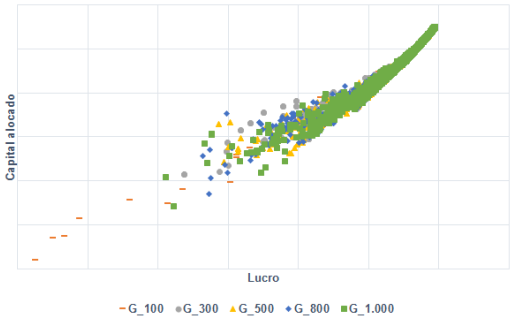
4.1.1 Fronteiras de soluções não dominadas obtidas

As fronteiras de soluções não dominadas (FSND), obtidas em função da números de gerações com o qual o Algoritmo Genético foi executado, encontram-se nas seguintes figuras: Figura 11, Figura 12, Figura 13, Figura 14, Figura 15, Figura 16, sendo, respectivamente, as FSND para $G = 100$, $G = 300$, $G = 500$, $G = 800$ e $G = 1.000$.

Conforme mencionado na subseção 2.1.1, os dados utilizados no problema desta seção foram obtidos em uma instituição do setor financeiro brasileiro e, por motivos de confidencialidade, não podem ser divulgados. Por este motivo, as escalas dos gráficos das figuras das FSND, para cada geração, não estão sendo mostradas. Ressalta-se que os gráficos mencionados possuem escalas idênticas (inclusive com valores máximos e mínimos dos eixos sendo iguais).

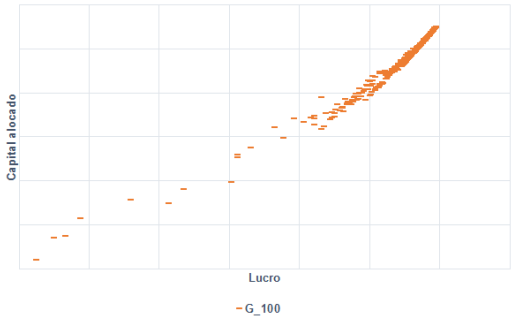
Com base nos gráficos das figuras mencionadas, percebe-se que, quanto maior a quantidade de gerações com que o Algoritmo Genético é executado, maior é a concentração de soluções não dominadas próximas do centro do gráfico. Ou seja, aumenta-se a

Figura 11 – FSND para todos G.



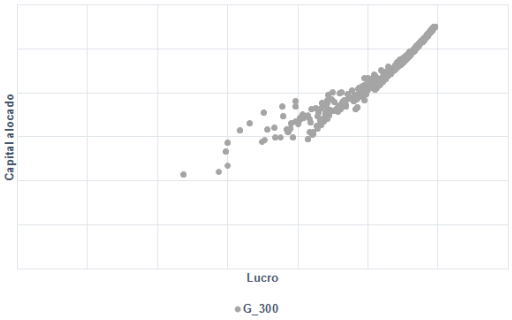
Fonte: produzido pelo autor.

Figura 12 – FSND para $G = 100$.



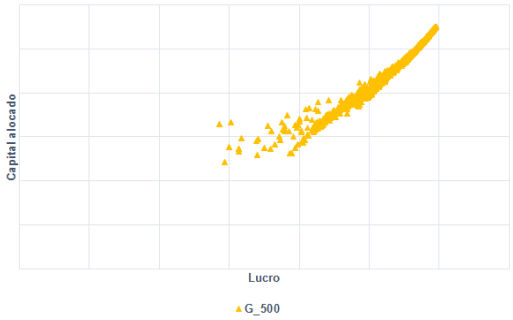
Fonte: produzido pelo autor.

Figura 13 – FSND para $G = 300$.



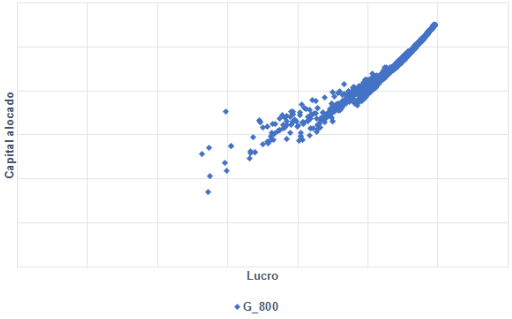
Fonte: produzido pelo autor.

Figura 14 – FSND para $G = 500$.



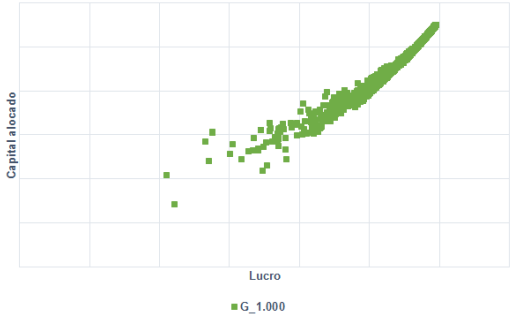
Fonte: produzido pelo autor.

Figura 15 – FSND para $G = 800$.



Fonte: produzido pelo autor.

Figura 16 – FSND para $G = 1.000$.



Fonte: produzido pelo autor.

concentração de soluções que não estão próximas do valor máximo de Equação 2.1, mas que também não estão próximas do valor mínimo de Equação 2.2.

Em termos de soluções extremas, nota-se que G influencia extremamente pouco na qualidade da melhor solução para a Equação 2.1, com o valor máximo de lucro, obtido em $G = 1.000$, sendo apenas 0,0056% maior do que o menor dos lucros máximos obtidos nas execuções com menores quantidades de gerações. Porém, nota-se que a quantidade de gerações é fortemente influente no valor mínimo obtido para a Equação 2.2. Neste caso, a solução com valor mínimo é encontrada em uma execução onde $G = 100$ e é 13,44% menor do que segunda melhor para o objetivo de minimização de alocação de capital,

encontrada em $G = 800$ (as soluções mínimas das demais gerações para a Equação 2.2 possuem valores bastante próximos ao encontrado em $G = 800$).

Embora não seja possível visualizar nos gráficos das figuras das fronteiras de soluções não dominadas, a quantidade de gerações sugere não apresentar relação direta com a quantidade de execuções do Algoritmo Genético proposto onde a quantidade máxima de indivíduos na FSND[1] é verificada. A quantidade de fronteiras com 100 indivíduos na FSND passa de 23, em $G = 100$ para 30[2] em $G = 500$, sendo 27 e 28, respectivamente, em $G = 300$ e $G = 800$. Nas execuções com mil gerações, o número de fronteiras com 100 indivíduos foi de 25, porém, obteve-se 3 fronteiras com 99 indivíduos.

4.1.2 Soluções extremas

Na Tabela 5 são mostradas as soluções extremas encontradas tanto para a Equação 2.1 quanto para a Equação 2.2. Na referida tabela, além dos percentuais de capital alocado em cada produto, há a indicação do capital disponível que não foi alocado (Não alocado) e as variações de lucro, capital alocado e RAROC (respectivamente Δ Lucro, Δ Capital alocado e Δ RAROC), de cada solução, em relação aos valores reais da instituição que forneceu os dados para análise.

As soluções mostradas na Tabela 5 correspondem ao valor mínimo de capital alocado, o qual foi obtido quando o Algoritmo Genético foi executado com cem gerações, indicado por $G = 100$**,** *min capital*, ao valor de capital mínimo obtido em execuções do Algoritmo Genético onde a quantidade de gerações é diferente de cem, mostrado por $G = 800$**,** *min capital* e, ao máximo valor de lucro encontrado em todas as execuções do Algoritmo Genético, sendo este último indicado por $G = 1.000$**,** *max lucro*.

Quando compara-se as soluções $G = 800$**,** *min capital* e $G = 1.000$**,** *max lucro*, nota-se que a maioria dos produtos recebe a mesma alocação de capital, independentemente de a solução extrema ser voltada para a maximização de lucro ou para a minimização de capital alocado. Isto ocorre em função das necessidades de alocação de capital e geração de lucro dos produtos onde isto ocorre serem similares. As grandes diferenças de alocação de capital entre estas duas soluções estão basicamente nos produtos 10, 11, 17 e 18. Sendo que, em função de buscar a minimização de capital alocado, a solução $G = 800$**,** *min capital* aloca pouco capital nos produtos 10 e 11, pois estes, apesar de trazerem lucros altos, demandam alto investimento de capital. O oposto ocorre com os produtos 17 e 18, os quais trazem lucros relativamente baixos, quando comprados aos demais, porém exigem baixa alocação de capital.

[1] A quantidade máxima de indivíduos não dominados é 100, que é o número de indivíduos que foi utilizado na população do Algoritmo Genético proposto.

[2] 30 é o valor máximo de fronteiras com 100 indivíduos em uma geração, uma vez que foram feitas 30 execuções do Algoritmo Genético proposto para cada quantidade de G analisada.

Tabela 5 – Soluções extremas.

	$G = 100$, *min capital*	$G = 800$, *min capital*	$G = 1.000$, *max lucro*
p_1	1,5%	1,5%	1,5%
p_2	2,0%	2,0%	2,0%
p_3	2,3%	2,3%	2,3%
p_4	2,8%	2,8%	2,8%
p_5	3,1%	3,1%	3,1%
p_6	6,6%	6,6%	6,6%
p_7	0,0%	13,0%	13,0%
p_8	3,4%	3,4%	3,4%
p_9	9,9%	9,9%	9,9%
p_{10}	7,2%	0,0%	7,2%
p_{11}	0,0%	1,5%	14,8%
p_{12}	0,0%	15,1%	15,1%
p_{13}	3,9%	3,9%	3,9%
p_{14}	7,9%	7,9%	7,9%
p_{15}	4,0%	0,4%	4,0%
p_{16}	3,9%	0,5%	2,4%
p_{17}	3,7%	0,4%	0,0%
p_{18}	2,4%	0,3%	0,0%
Não alocado	35,4%	25,3%	0,0%
Δ Lucro	-33,7%	-15,9%	7,3%
Δ Capital alocado	-35,4%	-25,3%	0,0%
Δ RAROC	2,6%	12,7%	7,3%

Fonte: Produzido pelo autor.

A solução $G = 100$, *min capital*, apesar de ser a que entrega o menor valor de capital alocado, é a menos eficiente em termos de RAROC, uma vez que a mesma aloca, dentre as três soluções, o maior percentual de capital nos produtos 16, 17 e 18, os quais, apesar de exigirem baixa alocação de capital, são mais arriscados e trazem menos retorno do que os demais.

Assim sendo, em um cenário econômico desfavorável, onde, por exemplo, deseja-se assumir menores riscos e alocar menos capital, mesmo que isto implique em redução de lucros, a solução proposta em $G = 800$, *min capital* é mais interessante do aquela mostrada em $G = 100$, *min capital*, pois, mesmo que na primeira o capital alocado não se reduza tanto quanto na segunda, o lucro e, especialmente o RAROC, são muito melhores.

Embora a melhor solução extrema voltada para a maximização de lucros tenha sido encontrada em uma execução do Algoritmo Genético com mil gerações (solução $G = 1.000$, *max lucro* da Tabela 5), as soluções extremas das execuções com as demais quantidades de gerações, além de serem muito próximas em termos de lucro, conforme citado na seção subseção 4.1.1, também são basicamente equivalentes em termos de RAROC. Isto, aliado ao fato de tanto o RAROC quanto o lucro da solução $G = 1.000$, *max lucro* serem 7,3% maiores dos que os mesmos indicadores encontrados na solução utilizada pela instituição financeira objeto de análise, apesar da utilização da mesma quantidade de capital, mostra que o emprego do Algoritmo Genético proposto teria impacto significativo

na melhoria, tanto do lucro, quanto do RAROC obtidos nas operações de alocação de capital da referida instituição.

4.2 Problema de alocação de aeronaves em portões de embarque

Os experimentos para busca pelas soluções do GAP abordado neste trabalho foram feitos através de três variações distintas no Algoritmo Genético, bem como pela implementação do modelo em um software de programação matemática para resolução de problemas de programação linear.

As variações aplicadas ao Algoritmo Genético ocorreram em função de dois fatores, sendo um deles a definição de um critério de factibilidade, onde soluções em que $x_i + p_i > d_i$ são consideradas infactíveis, uma vez que não contribuem para a minimização da Equação 2.7. Além deste, outro fator de modificação consistiu em alterar a Equação 2.7, de modo que tal objetivo do problema passou a ser a minimização do atraso máximo dentre as aeronaves em tal condição.

Na primeira etapa de testes, soluções infactíveis não foram penalizadas durante o processo de atribuição de *rank*. Neste cenário, as soluções concorrentes foram avaliadas apenas em função dos valores que cada uma delas alcançava nas funções objetivos do problema (Equação 2.7, Equação 2.8, Equação 2.9). Daqui em diante, tal etapa de testes será chamada de cenário A.

A segunda etapa de experimentos considerou, inicialmente, para atribuição de *rank*, o critério de factibilidade entre as soluções concorrentes, onde soluções fatíveis receberam *rank* melhor do que as não factíveis. Caso ambas soluções fossem factíveis, o melhor *rank* era definido em função do desempenho das mesmas em relação aos valores que atingiam nas funções objetivos. Esta etapa será, a partir daqui, referenciada como cenário B.

Na terceira etapa, daqui em diante denominada cenário C, o processo de atribuição de *rank* foi idêntico ao observado no cenário A. Porém, aqui, optou-se por alterar a Equação 2.7, de modo que tal objetivo do problema passou a ser a minimização do atraso máximo. Ou seja, onde $x_i + p_i > d_i$, deseja-se:

$$\text{minimizar} \quad f_1' = max(x_i + p_i - d_i) \tag{4.1}$$

Na implementação do modelo matemático em software de programação matemática para resolução de problemas de programação linear, doravante denominada cenário D, utilizou-se o IBM ILOG CPLEX Optimization Studio (em versão acadêmica, disponibilizada pela IBM[3]). O referido software apresenta linguagem de modelagem algébrica para

[3] *https://www.ibm.com/developerworks/br/downloads/ws/ilogcplex/index.html*

escrita da função objetivo que se deseja otimizar e das restrições consideradas no modelo, sendo que após a implementação do modelo, o programa é executado por um solver[4].

No cenário D, o modelo foi implementado através do Método das Ponderações, descrito na subseção 1.2.1. Devido ao elevado tempo necessário para a resolução do problema através da metodologia apresentada neste cenário, foram realizadas apenas três combinações de variações dos pesos aplicados aos objetivos do GAP. Tais combinações podem estão apresentadas na Tabela 11, da subseção 4.2.4. Matematicamente, o objetivo do problema se tornou:

$$\text{minimizar} \quad f = p_1 f_1 + p_2 f_2 + p_3 f_3 \tag{4.2}$$

Onde p_1 é o peso da Equação 2.7, representada por f_1, p_2 é o peso da Equação 2.8, representada por f_2 e p_3 é o peso da Equação 2.9, representada por f_3, sendo que $p_1 + p_2 + p_3 \leqslant 1$, $p_1 \geqslant 0$, $p_2 \geqslant 0$ e $p_3 \geqslant 0$.

A seguir, encontram-se resultados e análises dos referidos cenários. Os valores mostrados, quando referentes aos dados obtidos através da execução do Algoritmo Genético, são a média e o desvio padrão referentes às 30 execuções do mesmo feitas para cada combinação de cenário e número de gerações.

4.2.1 Impacto do número de gerações na solução do GAP

A Tabela 6 mostra os resultados de vários indicadores que medem a qualidade da solução do GAP em função do número de gerações com o qual o Algoritmo Genético foi executado. Na referida tabela, f_1 min é equivalente à Equação 2.7 e é medido em horas, f_3 min representa a Equação 2.9, sendo medida em quilômetros. Já infac min é a quantidade mínima de aeronaves que têm o horário de saída comprometido (aeronaves que atrasam). O indicador qtd FSND, por sua vez, diz qual a quantidade de soluções (indivíduos) presentes na fronteira de soluções não dominadas, enquanto tempo refere-se ao tempo de execução, em segundos, do Algoritmo Genético.

O indicador f_2 max, por sua vez, avalia um objetivo oposto ao desejado na Equação 2.8, ou seja, f_2 max indica a quantidade máxima de aeronaves alocadas ao *apron*. Adotou-se está métrica de avaliação em relação ao objetivo da Equação 2.8, pois, independentemente do número de gerações, são obtidas soluções onde a Equação 2.8 é igual a zero.

Com base nos dados apresentados na Tabela 6, nota-se que, ao passo em que se aumenta o números de gerações, todos os indicadores melhoram, com exceção do tempo de execução, o qual tem correlação praticamente igual 1 com G. Com 100 gerações, o tempo mínimo médio da soma de atraso nas decolagens (f_1 min) é 2,91 $\pm$ 1,34 horas,

[4] Solver é uma parte do software matemático que resolve o problema. Diferentes softwares de programação matemática podem utilizar solvers distintos.

Tabela 6 – Resultados obtidos nos testes do cenário A.

G	f_1 **min**	f_2 **max**	f_3 **min**	**infac min**	**qtd FSND**	**tempo**
100	2,91 ± 1,34	5,33 ± 2,02	10.333,8 ± 147,2	4,77 ± 1,65	35,83 ± 8,87	0,76 ± 0,01
300	0,78 ± 0,55	3,27 ± 1,61	9.970,3 ± 107,8	2,37 ± 1,33	45,30 ± 8,71	2,28 ± 0,03
500	0,19 ± 0,27	2,57 ± 1,31	9.811,9 ± 106,9	0,93 ± 1,03	54,87 ± 9,81	3,79 ± 0,06
800	0,09 ± 0,14	2,47 ± 1,38	9.656,6 ± 94,7	0,53 ± 0,76	58,83 ± 10,82	6,09 ± 0,07
1.000	0,04 ± 0,10	1,80 ± 1,28	9.588,2 ± 72,4	0,23 ± 0,50	61,33 ± 9,91	7,61 ± 0,08

Fonte: Produzido pelo autor.

porém, com G = 1.000, este tempo cai para apenas 0,04 ± 0,10 horas (uma redução de cerca de 98,6%). Ainda conforme ocorre aumento em G, a quantidade máxima média de aeronaves alocadas ao *apron* diminui, assim como também ocorre a diminuição de f_1 min e f_3 min, o que mostra clara melhora na qualidade das soluções, uma vez que menos aeronaves alocadas ao *apron*, teoricamente, tendem a incrementar a quantidade de atrasos. O objetivo da Equação 2.9, embora também melhore conforme mais gerações são incrementadas ao Algoritmo Genético, é menos sensível a tal incremento, tendo seu valor mínimo médio sendo reduzido em 7,2% de $G = 100$ para $G = 1.000$.

Ainda conforme a tabela Tabela 6, nota-se que o número médio de aeronaves que atrasam diminui conforme G aumenta. Além disso, o atraso médio ($\frac{f_1 \text{min}}{\text{infac min}}$) também é reduzido. Enquanto o número de aeronaves que atrasam é reduzido, em média, 56,6% de $G = 100$ para $G = 1.000$, o tempo médio de atraso passa de 0,61 horas para 0,17 para horas, ou seja, uma redução de 71,5%.

Finalmente, ainda baseado na Tabela 6, é possível notar que a quantidade de indivíduos na fronteira de soluções não dominadas, ou seja, soluções não dominadas, aumenta significativamente de $G = 100$ para $G = 1.000$. Tal aumento é de cerca de 71,2%.

A tabela Tabela 7 mostra a quantidade de soluções onde a Equação 2.7 é igual a zero, ou seja, onde não há atrasos na decolagem das aeronaves. Além disso, em tal tabela é possível ver também o percentual destas soluções em relação ao total de soluções na FSND. Quanto maior o valor de G, maiores são as quantidades de soluções onde não há atrasos nas decolagens e maiores são as quantidades de tais soluções em relação ao tamanho da fronteira de soluções não dominadas.

Tabela 7 – Distribuição de soluções de atraso zero em função da quantidade de gerações.

Gerações	100	300	500	800	1.000
Qtd. de soluções de atraso zero	0,0 ± 0,0	0,13 ± 0,56	0,67 ± 0,94	1,27 ± 1,50	1,77 ± 1,63
Qtd. de soluções em FSND	35,83 ± 8,87	45,30 ± 8,71	54,87 ± 9,81	58,83 ± 10,82	61,33 ± 9,91
Soluções de atraso zero em FSND	0,0%	0,3%	1,2%	2,2%	2,9%

Fonte: Produzido pelo autor.

A Tabela 8 mostra a taxa de melhoria dos indicadores da Tabela 6 para cada variação de G. O decréscimo nas taxas de melhoria Δ f_3 min e Δ qtd FSND indicam que tais indicadores estão cada vez mais estáveis em relação a novos aumentos de G. Por outro lado, apesar de f_1 min, f_2 max e infac min apresentarem soluções com elevado grau de qualidade, as taxas de Δ f_1 min, Δ f_2 max e Δ infac min sugerem que tais indicadores ainda podem ser melhorados caso o Algoritmo Genético seja executado com maiores quantidades de gerações.

Tabela 8 – Variações percentuais dos indicadores da Tabela 6 em função do aumento do número de gerações.

$G_i \Rightarrow G_f$	Δ f_1 **min**	Δ f_2 **max**	Δ f_3 **min**	Δ **infac min**	Δ **qtd FSND**	Δ **tempo**
100 ⇒ 300	-73,2%	-38,6%	-3,5%	-50,3%	26,4%	200,0%
300 ⇒ 500	-75,6%	-21,4%	-1,6%	-60,8%	21,1%	66,2%
500 ⇒ 800	-52,6%	-3,9%	-1,6%	-43,0%	7,2%	60,7%
800 ⇒ 1.000	-55,6%	-27,1%	-0,7%	-56,6%	4,2%	25,0%

Fonte: Produzido pelo autor.

Conforme evidenciado pela Tabela 6, pela Tabela 7 e pela Tabela 8, a quantidade de gerações com que o Algoritmo Genético proposto é executado altera significativamente o tempo de execução do mesmo, bem como a qualidade das soluções obtidas no problema de alocação de aeronaves em portões de embarque.

Os dados apresentados nesta seção são referentes ao cenário A. Os cenários B e C apresentaram características semelhantes às encontradas em A em relação às variações do números de gerações com o qual o Algoritmo Genético é executado.

4.2.2 Cenário A x Cenário B

Em busca de melhorar o processo de ranqueamento das soluções, de modo que soluções ideais (com zero atraso) para a Equação 2.7 aparecessem com maior frequência na fronteira de soluções não dominadas, penalizou-se os indivíduos que possuíssem aeronaves cujo horário de partida estivesse atrasado, de forma que tais soluções sempre recebessem o maior *rank*, para que fossem preteridas durante o processo de formação da FSND.

Porém, tal processo mostrou-se ineficiente, uma vez que soluções que poderiam vir a se tornarem adequadas foram praticamente descartadas logo de início. Para tal processo funcionar corretamente, provavelmente deve-se aumentar significativamente a quantidade de gerações, e até mesmo de indivíduos da população. Sendo assim, como pode ser verificado na Tabela 9, o cenário A, onde não há penalização para soluções infactíveis, mostrou-se muito mais eficiente na busca pelas melhores soluções do GAP do que o cenário B.

Para o mesmo número de indivíduos na população e mesma quantidade de gerações, no cenário A, as melhores soluções de f_1 min são, em média, 116,5 vezes melhores

Tabela 9 – Comparação entre os cenários A e B quando $G = 1.000$.

Cenário	f_1 **min**	f_2 **max**	f_3 **min**	**infac min**	**qtd FSND**
A	0,04 ± 0,10	1,80 ± 1,28	9.588,2 ± 72,4	0,23 ± 0,50	61,33 ± 9,91
B	4,66 ± 1,84	32,80 ± 2,59	10.037,8 ± 124,9	7,60 ± 2,32	100,00 ± 0,00

Fonte: Produzido pelo autor.

do que as encontradas no cenário B. Além disso, em nenhuma das execuções do cenário B encontrou-se alguma solução com atraso zero. Assim como ocorre com f_1 min, f_2 max, f_3 min e infac min também são melhores no cenário A. Apesar de f_2 max ser bastante elevado no cenário B, há sim soluções onde não há alocações de aeronaves ao *apron*. Embora a quantidade de indivíduos na fronteira de soluções não dominadas seja maior no cenário B do que no cenário A, isto não se reflete em soluções de qualidade superior, conforme evidenciado pelos dados da Tabela 9.

As comparações entre os cenários A e B foram feitas para as execuções de geração igual a 1.000, pois foram as de melhores resultados em ambos.

4.2.3 Cenário A x Cenário C

A tabela Tabela 10 mostra os resultados obtidos nos cenários A e C, onde os Algoritmos Genéticos executados possuem a mesma quantidade de indivíduos na população e o mesmo número de gerações, os quais, respectivamente são 100 e 1.000.

Tabela 10 – Comparação entre os cenários A e C quando $G = 1.000$.

Cenário	f_1 **min**	f_2 **max**	f_3 **min**	**infac min**	**qtd FSND**
A	0,04 ± 0,10	1,80 ± 1,28	9.588,2 ± 72,4	0,23 ± 0,50	61,33 ± 9,91
C	0,71 ± 0,24	1,80 ± 1,25	9.590,7 ± 110,7	9,80 ± 3,74	44,33 ± 9,75

Fonte: Produzido pelo autor.

Embora as soluções encontradas no cenário C sejam melhores do que as verificadas no cenário B, as soluções obtidas no cenário A ainda mostram-se ser as mais eficientes para o GAP.

Nota-se, com base na Tabela 10, que o mínimo atraso médio de uma aeronave no cenário C foi de 0,71 ± 0,24 horas, o que é maior do que média da soma de atrasos de todas as aeronaves no cenário A. Ou seja, na melhor hipótese, uma aeronave atrasada no canário C atrasa-se por mais tempo do que a soma dos atrasos de todas as aeronaves atrasadas no cenário A. Além disso, a quantidade mínima média das aeronaves atrasadas em C é cerca de 42,6 vezes maior do que em A. Outro fator que corrobora o fato de o cenário A ser melhor do que o cenário C para a resolução do GAP abordado é a não ocorrência de soluções, neste último, onde haja atraso zero na decolagem de aeronaves.

Tanto em A quanto em C, os valores médios máximos de alocação de aeronaves ao *apron* são baixos, sendo que tais cenários são praticamente equivalentes quanto a este quesito. Assim como nos cenários A e B, em todas as execuções do cenário C obteve-se soluções em que f_2 min é igual a zero.

Os valores médios mínimos percorridos pelos passageiros (f_3 min) nos cenários A e C são bastante próximos, porém, em C há maior variabilidade de soluções referentes a este objetivo.

Em termos de soluções na fronteira de soluções não dominadas, há cerca de 27,5% mais opções de escolhas no cenário A, quando comparado com o cenário C.

4.2.4 Cenário D

Conforme apresentado na Tabela 12, o tempo de execução de cada simulação executado no software IBM ILOG CPLEX Optimization Studio é muito maior do que o tempo de execução de cada iteração do Algoritmo Genético proposto (Tabela 6). Isto posto, executou-se a implementação, no referido software, com apenas três combinações de pesos aos objetivos do GAP, de modo que tais combinações proporcionassem os valores extremos (mínimos) para cada objetivo do problema. Para tanto, na simulação 1, por exemplo, Equação 2.7, recebeu peso um, enquanto que a Equação 2.8 e a Equação 2.9 receberam peso zero.

Tabela 11 – Pesos para as funções objetivos nas execuções do Método das Ponderações.

Simulação	p_1 **de** f_1	p_2 **de** f_2	p_3 **de** f_3
1	1	0	0
2	0	1	0
3	0	0	1

Fonte: Produzido pelo autor.

A Tabela 12 mostra os valores obtidos para cada objetivo do GAP, em cada simulação executada no software IBM ILOG CPLEX Optimization Studio. Para a simulação 1, onde prioriza-se o objetivo da Equação 2.7, obteve-se solução com zero atraso, porém, neste caso, além de todas as aeronaves terem sido alocadas ao *apron*, a distância total percorrida pelos passageiros foi bastante elevada quando comparada com as melhores soluções voltadas ao objetivo da Equação 2.7 encontradas pelo Algoritmo Genético executado com $G = 1.000$. Em média, o valor das melhores soluções para a Equação 2.7 foi 0,04 $\pm$ 0,10 horas (incluindo soluções com atraso zero) e, para este conjunto de soluções, a distância total percorrida pelos passageiros foi, em média, 10.653.117,7 $\pm$ 319.787,0 km, enquanto que a quantidade de aeronaves alocadas ao *apron* foi 0,73 $\pm$ 1,21, em média.

Para a simulação 3, onde prioriza-se o objetivo da Equação 2.9, a distância total percorrida pelos passageiros foi de 9.547,3 km. Tal solução não é necessariamente a mínima possível, pois, embora o solver do software IBM ILOG CPLEX Optimization Studio seja capaz de encontrar a solução ótima para o problema que nele é implementado, não permitiu-se que o software executasse o processo por mais do que 3.600 segundos. Ou seja, após uma hora de execução do solver, o processo foi interrompido, salvando a melhor solução obtida até então. Em média, as melhores soluções voltadas ao objetivo da Equação 2.9 encontradas pelo Algoritmo Genético com $G = 1.000$ apresentaram valor de 9.588,2 $\pm$ 72,4, cerca de 0,42% maior do que a solução obtida pelo software de programação matemática para resolução de problemas de programação linear. Entretanto, o Algoritmo Genético proposto ao GAP encontrou, em três de suas trinta execuções com $G = 1.000$, várias soluções melhores em relação ao objetivo da Equação 2.9, quando comparadas com a solução obtida através do IBM ILOG CPLEX Optimization Studio. Detre estas soluções, o menor valor de distância total percorrida pelos passageiros foi de 9.407,2 km, cerca de 1,5% menor do que a solução obtida pelo referido software.

Tabela 12 – Resultados obtidos através do Método das Ponderações.

Simulação	**valor f_1 (h)**	**valor f_2 (aero)**	**valor f_3 (km)**	**tempo (s)**
1	0,0	61	15.765,1	159,0
2	26,3	0	11.779,7	535,2
3	29,5	0	9.547,3	3.600

Fonte: Produzido pelo autor.

Embora as soluções obtidas para o GAP através da implementação do mesmo no software IBM ILOG CPLEX Optimization Studio possam atingir o valor ótimo para cada combinação de pesos, a obtenção de uma fronteira de soluções não dominadas com várias soluções é muito custosa computacionalmente, conforme pode-se observar pelo tempo de execução de cada uma das simulações, mostrados na Tabela 12 (cada simulação obtém uma solução na dominada). Tal fato, em conjunto com capacidade de o Algoritmo Genético proposto ser capaz de entregar uma fronteira de soluções não dominadas a cada execução, corrobora a escolha do Algoritmo Genético como ferramenta de busca pela solução do GAP apresentado neste trabalho.

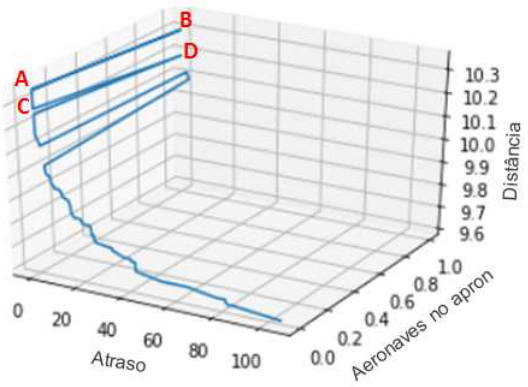
Os valores das soluções obtidas através do Algoritmo Genético apresentados nesta seção referem-se ao algoritmo executado no cenário A, o qual obteve melhores resultados quando comparado com as execuções dos cenários B e C.

4.2.5 Fronteira de soluções não dominadas

A Figura 17 mostra a fronteira de soluções não dominadas de uma das 30 execuções do Algoritmo Genético para o caso de mil gerações. Escolheu-se utilizar alguma

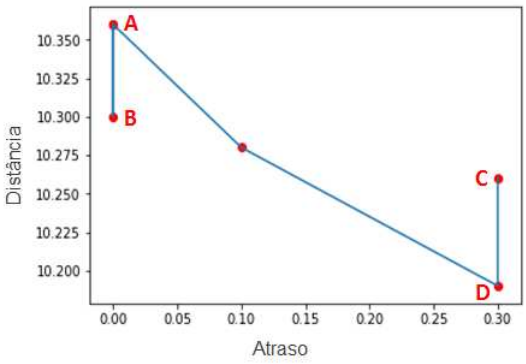
FSND gerada nas execuções de G = 1.000 pelo fato de este número de gerações ter sido responsável pela produção das melhores soluções para o problema.

Figura 17 – FSND com mil gerações.



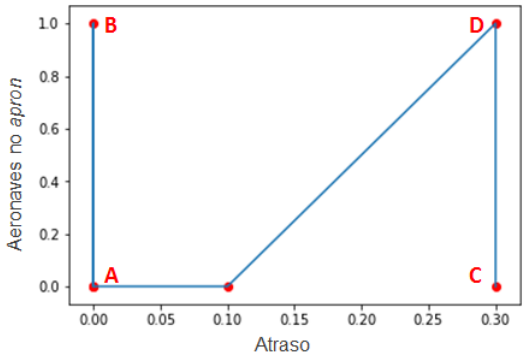
Fonte: produzido pelo autor.

Figura 18 – Parte da FSND: atraso x distância.



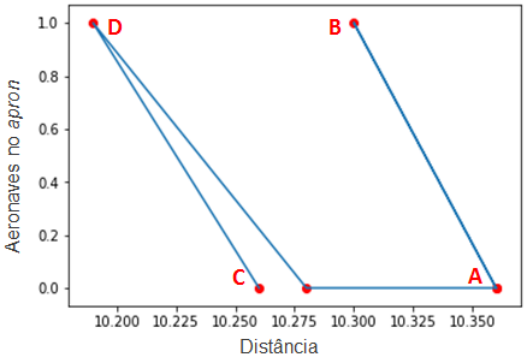
Fonte: produzido pelo autor.

Figura 19 – Parte da FSND: atraso x aeronaves no *apron*.



Fonte: produzido pelo autor.

Figura 20 – Parte da FSND: distância x aeronaves no *apron*.



Fonte: produzido pelo autor.

Os valores referentes aos pontos A, B, C e D, presentes na Figura 17, na Figura 18, na Figura 19 e na Figura 20, são mostrados Tabela 13, onde Atraso é referente ao objetivo da Equação 2.7, Aeronaves no *apron* refere-se ao objetivo da Equação 2.8 e Distância indica o objetivo da Equação 2.9.

Tabela 13 – Valores das soluções A, B, C e D.

Ponto	**Atraso (h)**	**Aeronaves no *apron***	**Distância (km $\times 10^{-3}$)**
A	0,0	0	10,36
B	0,0	1	10,30
C	0,3	0	10,26
D	0,3	1	10,19

Fonte: Produzido pelo autor.

De acordo com os pontos A, B, C e D mostrados nas figuras citadas, bem como com seus valores explicitados na Tabela 13, nota-se que, em pontos onde o atraso é igual

(A e B, onde o atraso é zero e, C e D, cujo atraso é de 0,30 horas), a alocação ou não de aeronaves ao *apron* influencia diretamente na distância total percorrida pelos passageiros, sendo que, nestes casos, quando aeronaves são alocadas ao *apron*, os passageiros precisam percorrer menores distâncias.

Além disso, nos gráficos da Figura 17 e da Figura 18, evidencia-se o conflito entre os objetivos da Equação 2.7 e da Equação 2.9. Ou seja, quando um deles é melhorado, o outro piora.

5 Conclusão

O Algoritmo Genético proposto foi aplicado em dois problemas importantes cujas áreas e características são bastante distintas. Um dos problemas, o de alocação de capital em instituições do setor financeiro, possui dois objetivos claramente conflitantes e exige que a solução seja representada por números reais. O outro problema abordado, de alocação de aeronaves em portões de embarque, por sua vez, além de possuir três objetivos conflitantes e exigir soluções com números inteiros, é um problema de *scheduling* NP-difícil. As aplicações nestes problemas com características diferentes mostraram que o Algoritmo Genético proposto é abrangente e facilmente adaptável, sendo necessário pequenas mudanças nas codificações dos indivíduos que compõem a população para que o mesmo resolva problemas complexos, porém de padrões díspares.

As soluções, em ambos os problemas abordados pelo Algoritmo Genético proposto, foram obtidas em tempo bastante satisfatório, sendo que as execuções que mais exigiram tempo para serem alcançadas foram as obtidas na resolução do GAP com execuções de mil gerações, onde o tempo médio de cada uma das trinta execuções realizadas foi de 7,61 $\pm$ 0,08 segundos.

Além do tempo de execução, as soluções obtidas também apresentaram qualidade bastante satisfatória em ambos problemas. Sendo que, para o GAP, foram necessárias execuções com mil gerações para obter-se uma quantidade razoável de soluções com atraso zero, ao passo que, para o problema de alocação de capital em instituições do setor financeiro, as soluções extremas, maximização de lucro e minimização de capital alocado, foram obtidas, respetivamente, com execuções com mil e cem gerações.

As soluções encontradas para o problema de alocação de capital em instituições do setor financeiro se mostraram satisfatórias, inclusive podendo vir a serem, de fato, adotadas na prática, uma vez que, quando comparadas com a solução padrão do problema, encontrada pela instituição que forneceu os dados para análise, mostraram-se relativamente melhores em termos de lucratividade e controle de risco e retorno (RAROC).

Além disso, em eventual cenário econômico desfavorável, onde, por exemplo, deseja-se assumir menores riscos e alocar menos capital, mesmo que isto implique em redução de lucro, há soluções que mostram-se bastante satisfatórias ao passo em que possibilitam reduções significativas de capital alocado, sem reduzir de maneira drástica o lucro. Ou seja, são soluções onde não se obtém lucro máximo, porém há elevado RAROC.

Verificou-se que, para o GAP, penalizar soluções infactíveis no momento da atribuição do *rank* não trouxe benefícios, ao contrário, fez com que mais soluções fossem descartadas nas primeiras gerações da execução do Algoritmo Genético, fazendo com que as

fronteiras de soluções não dominadas onde tal penalização ocorreu possuíssem indivíduos com piores soluções em relação às execuções do Algoritmo Genético em que não foram penalizadas as soluções infactíveis. Além disso, mesmo que o Algoritmo Genético proposto tenha sido facilmente adaptado para resolver o GAP alterando seu principal objetivo, de minimização da soma de atrasos na decolagem das aeronaves para minimização do atraso máximo das aeronaves que eventualmente excedam o horário previsto de partida, tal estratégia não mostrou-se eficiente em trazer melhores soluções ao problema.

Ao comparar-se os valores obtidos para o GAP através do Algoritmo Genético proposto com os valores encontrados através da implementação do problema no software IBM ILOG CPLEX Optimization Studio, verifica-se que a escolha do Algoritmo Genético como método de resolução para o GAP foi adequado, uma vez que tal método mostrou-se capaz de obter soluções com elevada qualidade e em excelente tempo computacional, quando compradas com as soluções obtidas através da implementação do GAP em software de programação matemática para resolução de problemas de programação linear.

Perspectivas futuras

Perspectivas de estudos futuros complementares ao apresentado incluem os seguintes pontos:

- Permitir que sejam alocados percentuais de capital independentemente da demanda que tal percentual atenderá, de maneira que se possa verificar possíveis melhorias nas soluções obtidas no problema de alocação de capital em empresas do setor financeiro;
- Adaptar a codificação proposta ao problema de alocação de capital em empresas do setor financeiro para resolver problemas de alocação de capital em carteiras de ações, afim de validar a utilidade do Algoritmo Genético em diversos cenários de alocação de capital;
- Não normalizar as soluções iniciais do problema de alocação de capital, afim de verificar a influência do processo de normalização na obtenção de soluções extremas em relação ao objetivo da Equação 2.2.
- Implementar o mecanismo de *ponto de referência*, presente no NSGA-III, proposto por Deb e Jain (2013), em substituição ao *crowding distance*, com o objetivo de encontrar maior diversidade de soluções no problema de alocação de capital.
- Executar o Algoritmo Genético proposto ao GAP com maiores números de gerações para verificar quais os parâmetros de execução garantem que as soluções verificadas para a minimização de soma de atraso atinja cenários de estabilidade, do mesmo modo que ocorreu para a minimização da distância total percorrida pelos passageiros quando o número de gerações passou de oitocentos para mil;
- Executar o cenário C proposto ao GAP com maiores quantidades de indivíduos e gerações para que se possa verificar se tal abordagem pode trazer soluções melhores do que àquelas verificadas no cenário A.

Referências

AHMED, M. B.; MANSOUR, F. Z.; HAOUARI, M. A pso approach for robust aircraft routing. In: IEEE. *2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. [S.l.], 2015. p. 219–223. Citado na página 52.

ANAC. *Dados Estatísticos: Base de dados subdividida por ano: 2016*. 2017. Agência Nacional de Aviação Civil. Disponível em: <http://www.anac.gov.br/assuntos/dados-e-estatisticas/dados-estatisticos/dados-estatisticos>. Acesso em: 02 dez 2017. Citado 3 vezes nas páginas 64, 102 e 103.

ARENALES, M.; ARMENTANO, V.; MORABITO, R.; YANASSE, H. *Pesquisa Operacional Para Cursos de Engenharia*. 1ªed. [S.l.]: Campus/Elsevier, 2007. Citado na página 24.

ARMANANZAS, R.; LOZANO, J. A. A multiobjective approach to the portfolio optimization problem. In: IEEE. *2005 IEEE Congress on Evolutionary Computation*. [S.l.], 2005. v. 2, p. 1388–1395. Citado na página 44.

ARROYO, J. E. C. *Heurísticas e metaheurísticas para otimização combinatória multiobjetivo*. Tese (Doutorado) — FEEC, Universidade Estadual de Campinas, 2002. Citado 3 vezes nas páginas 25, 26 e 27.

BABIĆ, O.; TEODOROVIĆ, D.; TOŠIĆ, V. Aircraft stand assignment to minimize walking. *Journal of Transportation Engineering*, American Society of Civil Engineers, v. 110, n. 1, p. 55–66, 1984. Citado na página 55.

BANDARA, S.; WIRASINGHE, S. Walking distance minimization for airport terminal configurations. *Transportation Research Part A: Policy and Practice*, Elsevier, v. 26, n. 1, p. 59–74, 1992. Citado na página 55.

BARBOSA, F. *O problema de alocaçao de berços: Aspectos teóricos e computacionais*. Dissertação (Dissertação de Mestrado) — IMECC, Universidade Estadual de Campinas, 2014. Citado na página 61.

BARBOSA, F. *Recents Advances on The Berth Allocation Problem*. Tese (Doutorado) — FEEC, Universidade Estadual de Campinas, 2018. Citado na página 69.

BCB. *Recomendações de Basileia*. 2019. Banco Central do Brasil. Disponível em: <https://www.bcb.gov.br/estabilidadefinanceira/recomendacoesbasileia>. Acesso em: 16 jun 2019. Citado na página 47.

BEHRENDS, J. A.; USHER, J. M. Aircraft gate assignment: using a deterministic approach for integrating freight movement and aircraft taxiing. *Computers & Industrial Engineering*, Elsevier, v. 102, p. 44–57, 2016. Citado na página 58.

BIHR, R. A. A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming. *Computers & Industrial Engineering*, Elsevier, v. 19, n. 1-4, p. 280–284, 1990. Citado na página 55.

BOLAT, A. Assigning arriving flights at an airport to the available gates. *Journal of the Operational Research Society*, Taylor & Francis, v. 50, n. 1, p. 23–34, 1999. Citado na página 56.

BRAAKSMA, J. Reducing walking distances at existing airports. In: *Airport Forum*. [S.l.: s.n.], 1977. v. 7. Citado na página 55.

CALLISTER, W. D. *Ciência e engenharia de materiais: uma introdução*. 7^{a}ed. [S.l.]: LTC, 2007. Citado 2 vezes nas páginas 28 e 29.

CAREY, M. A guide to choosing absolute bank capital requirements. In: *Ratings, rating agencies and the global financial system*. [S.l.]: Springer, 2002. p. 117–138. Citado na página 47.

CASTAING, J.; MUKHERJEE, I.; COHN, A.; HURWITZ, L.; NGUYEN, A.; MÜLLER, J. J. Reducing airport gate blockage in passenger aviation: Models and analysis. *Computers & Operations Research*, Elsevier, v. 65, p. 189–199, 2016. Citado na página 56.

CHEN, C.; LIU, T.; CHOU, J. Integrated short-haul airline crew scheduling using multiobjective optimization genetic algorithms. *IEEE Transactions On Systems, Man, and Cybernetics: Systems*, IEEE, v. 43, n. 5, p. 1077–1090, 2013. Citado na página 53.

CORNE, D. W.; KNOWLES, J. D.; OATES, M. J. The pareto envelope-based selection algorithm for multiobjective optimization. In: SPRINGER. *International conference on parallel problem solving from nature*. [S.l.], 2000. p. 839–848. Citado na página 38.

DAŞ, G. S. New multi objective models for the gate assignment problem. *Computers & Industrial Engineering*, Elsevier, v. 109, p. 347–356, 2017. Citado na página 58.

DEB, K. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary computation*, MIT Press, v. 7, n. 3, p. 205–230, 1999. Citado na página 20.

DEB, K.; AGRAWAL, S.; PRATAP, A.; MEYARIVAN, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In: SPRINGER. *International conference on parallel problem solving from nature*. [S.l.], 2000. p. 849–858. Citado 6 vezes nas páginas 7, 8, 17, 39, 65 e 67.

DEB, K.; DATTA, R. Hybrid evolutionary multi-objective optimization of machining parameters. *KanGAL Report*, v. 201105, 2011. Citado na página 16.

DEB, K.; JAIN, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 18, n. 4, p. 577–601, 2013. Citado na página 90.

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, IEEE, v. 6, n. 2, p. 182–197, 2002. Citado 3 vezes nas páginas 39, 40 e 41.

DELL'ARICCIA, G.; MARQUEZ, R. Information and bank credit allocation. *Journal of Financial Economics*, Elsevier, v. 72, n. 1, p. 185–214, 2004. Citado 2 vezes nas páginas 46 e 47.

DELL'ORCO, M.; MARINELLI, M.; ALTIERI, M. G. Solving the gate assignment problem through the fuzzy bee colony optimization. *Transportation Research Part C: Emerging Technologies*, Elsevier, v. 80, p. 424–438, 2017. Citado 2 vezes nas páginas 57 e 58.

DIEPEN, G.; AKKER, J. V. D.; HOOGEVEEN, J. a.; SMELTINK, J. Finding a robust assignment of flights to gates at amsterdam airport schiphol. *Journal of Scheduling*, Springer, v. 15, n. 6, p. 703–715, 2012. Citado na página 56.

DORNDORF, U.; DREXL, A.; NIKULIN, Y.; PESCH, E. Flight gate scheduling: State-of-the-art and recent developments. *Omega*, Elsevier, v. 35, n. 3, p. 326–334, 2007. Citado 2 vezes nas páginas 54 e 55.

DREXL, A.; NIKULIN, Y. Multicriteria airport gate assignment and pareto simulated annealing. *IIE Transactions*, Taylor & Francis, v. 40, n. 4, p. 385–397, 2008. Citado 3 vezes nas páginas 54, 57 e 58.

ERICKSON, M.; MAYER, A.; HORN, J. The niched pareto genetic algorithm 2 applied to the design of groundwater remediation systems. In: SPRINGER. *International Conference on Evolutionary Multi-Criterion Optimization*. [S.l.], 2001. p. 681–695. Citado na página 39.

FEO, T. A.; RESENDE, M. G. Greedy randomized adaptive search procedures. *Journal of global optimization*, Springer, v. 6, n. 2, p. 109–133, 1995. Citado na página 29.

FERREIRA, P. A. V. Otimização multiobjetivo: Teoria e aplicações. *Tese de livre docência*, FEEC, Universidade Estadual de Campinas, 1999. Citado na página 20.

FONSECA, C. M.; FLEMING, P. J. et al. Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In: CITESEER. *Icga*. [S.l.], 1993. v. 93, n. July, p. 416–423. Citado na página 35.

FREY, R.; MCNEIL, A. J. Var and expected shortfall in portfolios of dependent credit risks: conceptual and practical insights. *Journal of banking & finance*, Elsevier, v. 26, n. 7, p. 1317–1334, 2002. Citado na página 46.

FURFINE, C. Bank portfolio allocation: The impact of capital requirements, regulatory monitoring, and economic conditions. *Journal of Financial Services Research*, Springer, v. 20, n. 1, p. 33–56, 2001. Citado 2 vezes nas páginas 46 e 47.

GABRIEL, P. H. R.; DELBEM, A. C. B. *Fundamentos de Algoritmos Evolutivos*. 2017. Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo. Disponível em: <http://conteudo.icmc.usp.br/CMS/Arquivos/arquivos_enviados/BIBLIOTECA_113_ND_75.pdf>. Acesso em: 15 nov 2017. Citado na página 30.

GARCIA, V. J. *Metaheurísticas multiobjetivo para o problema de restauração do serviço em redes de distribuição de energia elétrica*. Tese (Doutorado) — FEEC, Universidade Estadual de Campinas, 2005. Citado 3 vezes nas páginas 26, 27 e 29.

GAREY, M. R.; JOHNSON, D. S.; SETHI, R. The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, INFORMS, v. 1, n. 2, p. 117–129, 1976. Citado na página 18.

GENÇ, H. M.; EROL, O. K.; EKSIN, İ.; BERBER, M. F.; GÜLERYÜZ, B. O. A stochastic neighborhood search approach for airport gate assignment problem. *Expert Systems with Applications*, Elsevier, v. 39, n. 1, p. 316–327, 2012. Citado na página 56.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, Elsevier, v. 13, n. 5, p. 533–549, 1986. Citado 2 vezes nas páginas 27 e 28.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. [S.l.]: Addison–Wesley Publishing Co, Reading. EEUU, 1989. Citado 2 vezes nas páginas 31 e 34.

GORDY, M. B. A risk-factor model foundation for ratings-based bank capital rules. *Journal of financial intermediation*, Elsevier, v. 12, n. 3, p. 199–232, 2003. Citado na página 46.

GREENBERG, H. *Mathematical programing glossary*. 2017. University of Colorado. Disponível em: <http://math.ucdenver.edu/~hgreenbe>. Acesso em: 12 nov 2017. Citado na página 27.

GU, Y.; CHUNG, C. A. Genetic algorithm approach to aircraft gate reassignment problem. *Journal of Transportation Engineering*, American Society of Civil Engineers, v. 125, n. 5, p. 384–389, 1999. Citado na página 56.

GUEYIÉ, J.-P.; GUIDARA, A.; LAI, V. S. Banks' non-traditional activities under regulatory changes: impact on risk, performance and capital adequacy. *Applied Economics*, Taylor & Francis, p. 1–14, 2019. Citado 2 vezes nas páginas 44 e 47.

HAGHANI, A.; CHEN, M.-C. Optimizing gate assignments at airport terminals. *Transportation Research Part A: Policy and Practice*, Elsevier, v. 32, n. 6, p. 437–454, 1998. Citado na página 55.

HORN, J. Multicriterion decision making. *Handbook of evolutionary computation*, Citeseer, v. 97, n. 1, 1997. Citado na página 22.

HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. E. A niched pareto genetic algorithm for multiobjective optimization. In: CITESEER. *Proceedings of the first IEEE conference on evolutionary computation, IEEE world congress on computational intelligence*. [S.l.], 1994. v. 1, p. 82–87. Citado na página 37.

İÇ, Y. T. Development of a credit limit allocation model for banks using an integrated fuzzy topsis and linear programming. *Expert Systems with Applications*, Elsevier, v. 39, n. 5, p. 5309–5316, 2012. Citado na página 47.

IU. *Gerenciamento de Riscos - Pilar 3*. 2015. Itaú Unibanco: Relação com Investidores. Disponível em: <https://www.itau.com.br/relacoes-com-investidores/Download.aspx?Arquivo=K0+KY8oGP7O2qfDh8Sv7iw==>. Acesso em: 15 jun 2019. Citado na página 45.

JIANG, Y.; ZENG, L.; LUO, Y. Multiobjective gate assignment based on passenger walking distance and fairness. *Mathematical Problems in Engineering*, Hindawi, v. 2013, 2013. Citado 2 vezes nas páginas 57 e 58.

JONG, K. A. D. *Evolutionary computation: a unified approach*. [S.l.]: MIT press, 2006. Citado na página 30.

KIM, S. H.; FERON, E.; CLARKE, J.-P. Gate assignment to minimize passenger transit time and aircraft taxi time. *Journal of Guidance, Control, and Dynamics*, American Institute of Aeronautics and Astronautics, v. 36, n. 2, p. 467–475, 2013. Citado 2 vezes nas páginas 57 e 58.

KNOWLES, J.; CORNE, D.; DEB, K. *Multiobjective problem solving from nature: from concepts to applications*. [S.l.]: Springer Science & Business Media, 2007. Citado na página 31.

KNOWLES, J. D.; CORNE, D. W. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, MIT Press, v. 8, n. 2, p. 149–172, 2000. Citado na página 38.

LEUNG, J. Y.-T. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. 4ªed. United States: Chapman and Hall/CRC, 2004. Citado na página 51.

LI, J.; XU, J. Multi-objective portfolio selection model with fuzzy random returns and a compromise approach-based genetic algorithm. *Information Sciences*, Elsevier, v. 220, p. 507–521, 2013. Citado na página 44.

LIM, A.; RODRIGUES, B.; ZHU, Y. Airport gate scheduling with time windows. *Artificial Intelligence Review*, Springer, v. 24, n. 1, p. 5–31, 2005. Citado na página 55.

LWIN, K.; QU, R.; KENDALL, G. A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization. *Applied Soft Computing*, Elsevier, v. 24, p. 757–772, 2014. Citado na página 44.

MARINELLI, M.; DELL'ORCO, M.; SASSANELLI, D. A metaheuristic approach to solve the flight gate assignment problem. *Transportation Research Procedia*, Elsevier, v. 5, p. 211–220, 2015. Citado 2 vezes nas páginas 57 e 58.

MARINS, F. A. S. *Introdução à pesquisa operacional*. 1ªed. São Paulo: Cultura Acadêmica: Universidade Estadual Paulista, 2011. Citado na página 15.

MARKOWITZ, H. Portfolio selection. *The journal of finance*, Wiley Online Library, v. 7, n. 1, p. 77–91, 1952. Citado na página 43.

MICHALEWICZ, Z. *Genetic algorithms+ data structures= evolution programs*. [S.l.]: Springer Science & Business Media, 2013. Citado na página 68.

MORCK, R.; YAVUZ, M. D.; YEUNG, B. Banking system control, capital allocation, and economy performance. *Journal of Financial Economics*, Elsevier, v. 100, n. 2, p. 264–283, 2011. Citado 2 vezes nas páginas 46 e 47.

OSYCZKA, A. Multicriteria optimization for engineering design. In: *Design optimization*. [S.l.]: Elsevier, 1985. p. 193–227. Citado na página 19.

PINEDO, M.; HADAVI, K. Scheduling: theory, algorithms and systems development. In: *Operations Research Proceedings 1991*. [S.l.]: Springer, 1992. p. 35–42. Citado na página 50.

QI, X.; YANG, J.; YU, G. Handbook of scheduling: Algorithms, models, and performance analysis. In: _____. 1. Berlin: CRC Press, 2004. v. 1, cap. Scheduling Problems in the Airline Industry, p. 50.1–50.20. Citado 2 vezes nas páginas 51 e 52.

RAMPAZZO, P. C. B. *PLANEJAMENTO HIDRELÉTRICO: OTIMIZAÇÃO MULTIOBJETIVO E ABORDAGENS EVOLUTIVAS*. Tese (Doutorado) — FEEC, Universidade Estadual de Campinas, 2012. Citado 2 vezes nas páginas 32 e 33.

ROCKAFELLAR, R. T.; URYASEV, S. Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, Elsevier, v. 26, n. 7, p. 1443–1471, 2002. Citado na página 46.

SABORIDO, R.; RUIZ, A. B.; BERMÚDEZ, J. D.; VERCHER, E.; LUQUE, M. Evolutionary multi-objective optimization algorithms for fuzzy portfolio selection. *Applied Soft Computing*, Elsevier, v. 39, p. 48–63, 2016. Citado na página 44.

SANTOS, A. A.; NOGALES, F. J.; RUIZ, E.; DIJK, D. V. Optimal portfolios with minimum capital requirements. *Journal of Banking & Finance*, Elsevier, v. 36, n. 7, p. 1928–1942, 2012. Citado na página 47.

SCHAFFER, J. D. Multiple objective optimization with vector evaluated genetic algorithms. In: LAWRENCE ERLBAUM ASSOCIATES. INC., PUBLISHERS. *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*. [S.l.], 1985. Citado 2 vezes nas páginas 34 e 35.

SILVA, M. F. S. *Abordagem para otimização multiobjetivo de regras heurísticas de sequenciamento em sistemas de manufatura job shop por meio de simulação computacional acoplada ao algoritmo genético*. Dissertação (Dissertação de Mestrado) — Universidade Nove de Julho, 2011. Citado na página 27.

SILVER, E. A.; VICTOR, R.; VIDAL, V.; WERRA, D. de. A tutorial on heuristic methods. *European Journal of Operational Research*, Elsevier, v. 5, n. 3, p. 153–162, 1980. Citado na página 26.

SKOLPADUNGKET, P.; DAHAL, K.; HARNPORNCHAI, N. Portfolio optimization using multi-obj ective genetic algorithms. In: IEEE. *2007 IEEE Congress on Evolutionary Computation*. [S.l.], 2007. p. 516–523. Citado na página 44.

SRINIVAS, N.; DEB, K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, MIT Press, v. 2, n. 3, p. 221–248, 1994. Citado na página 36.

STEUER, R. E. Multiple criteria optimization. *Theory, computation and applications*, Wiley, 1986. Citado 2 vezes nas páginas 16 e 23.

STOUGHTON, N. M.; ZECHNER, J. Optimal capital allocation using raroc™ and eva®. *Journal of Financial Intermediation*, Elsevier, v. 16, n. 3, p. 312–342, 2007. Citado 2 vezes nas páginas 46 e 47.

TANG, C.-H.; WANG, W.-C. Airport gate assignments for airline-specific gates. *Journal of Air Transport Management*, Elsevier, v. 30, p. 10–16, 2013. Citado na página 55.

TASCHE, D. Expected shortfall and beyond. *Journal of Banking & Finance*, Elsevier, v. 26, n. 7, p. 1519–1533, 2002. Citado na página 46.

TICONA, W. G. C. *Algoritmos evolutivos multi-objetivo para a reconstrução de árvores filogenéticas*. Tese (Doutorado) — ICMC, Universidade de São Paulo, 2003. Citado na página 34.

ULLMAN, J. D. Np-complete scheduling problems. *Journal of Computer and System sciences*, Academic Press, v. 10, n. 3, p. 384–393, 1975. Citado na página 64.

WANG, L.; ZHU, Q.-L.; XU, X.-F. Optimised assignment of airport gate configurations using an immune genetic algorithm. *Mathematical Structures in Computer Science*, Cambridge University Press, v. 24, n. 5, 2014. Citado na página 56.

XU, J.; BAILEY, G. The airport gate assignment problem: mathematical model and a tabu search algorithm. In: IEEE. *Proceedings of the 34th annual Hawaii international conference on system sciences*. [S.l.], 2001. p. 10–pp. Citado na página 55.

YAN, S.; HUO, C.-M. Optimization of multiple objective gate assignments. *Transportation Research Part A: Policy and Practice*, Elsevier, v. 35, n. 5, p. 413–432, 2001. Citado 2 vezes nas páginas 57 e 58.

ZARUTSKIE, R. Competition, financial innovation and commercial bank loan portfolios. *Journal of Financial Intermediation*, Elsevier, v. 22, n. 3, p. 373–396, 2013. Citado 2 vezes nas páginas 45 e 47.

ZHANG, H.-H.; XUE, Q.-W.; JIANG, Y. Multi-objective gate assignment based on robustness in hub airports. *Advances in Mechanical Engineering*, SAGE Publications Sage UK: London, England, v. 9, n. 2, p. 1687814016688588, 2017. Citado 5 vezes nas páginas 58, 63, 64, 102 e 103.

ZHU, H.; WANG, Y.; WANG, K.; CHEN, Y. Particle swarm optimization (pso) for the constrained portfolio optimization problem. *Expert Systems with Applications*, Elsevier, v. 38, n. 8, p. 10161–10169, 2011. Citado na página 44.

ZITZLER, E.; DEB, K.; THIELE, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, MIT Press, v. 8, n. 2, p. 173–195, 2000. Citado na página 34.

ZITZLER, E.; LAUMANNS, M.; THIELE, L. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische, v. 103, 2001. Citado na página 38.

ZITZLER, E.; THIELE, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, IEEE, v. 3, n. 4, p. 257–271, 1999. Citado na página 37.

Apêndices

APÊNDICE A – Determinação das funções objetivo do problema de alocação de capital em empresas do setor bancário

Originalmente, a Equação 2.1 e a Equação 2.2 podem ser definidas da seguinte maneira:

$$f_1 = \sum_{i=1}^{P} lu_i d_i^* \tag{A.1}$$

$$f_2 = \sum_{i=1}^{P} r_i cr_i d_i^* \tag{A.2}$$

Onde d_i^* representa a demanda total pelo produto i.

Sendo p_i^* o capital necessário para atender toda a demanda de determinado produto i, sabe-se que o mesmo é definido por:

$$p_i^* = d_i^* r_i cr_i \tag{A.3}$$

Porém, como não necessariamente precisa-se que toda a demanda seja atendida, ou até mesmo não se tem o capital total C suficiente para isto, tem-se que o capital que atende uma demanda qualquer é dado por:

$$p_i = d_i r_i cr_i \tag{A.4}$$

Com isto, pode-se definir f_1 e f_2 em função de p_i, ao passo em que se substitui, em ambas, d_i^* por d_i, sendo este último definido com base na Equação A.4:

$$d_i = \frac{p_i}{r_i cr_i} \tag{A.5}$$

Deste modo, aplicando a Equação A.5 na Equação A.1 e na Equação A.2, tem-se as equações finais referentes aos objetivos do problema de alocação de capital:

$$f_1 = \sum_{i=1}^{P} p_i \frac{lu_i C}{r_i cr_i} \tag{A.6}$$

$$f_2 = \sum_{i=1}^{P} p_i C \tag{A.7}$$

Anexos

ANEXO A – Dados utilizados nas instâncias de testes do GAP

A Tabela 14 traz os horários de chegada e partida, em minutos, bem como a quantidade de passageiros que desembarcarão e embarcarão das aeronaves analisadas no GAP apresentado neste trabalho.

Tabela 14 – Horários de chegada e partida e quantidade de passageiros das aeronaves.

aeronave (i)	a_i (min)	d_i (min)	qd_i	qa_i
1	0	65	167	128
2	0	70	178	125
3	0	70	138	207
4	5	75	178	239
5	20	80	181	155
6	20	90	76	121
7	40	100	173	92
8	40	110	144	175
9	45	105	181	176
10	55	115	148	190
11	65	135	28	39
12	80	160	36	91
13	85	155	128	126
14	90	160	134	177
15	100	170	97	93
16	100	160	204	140
17	105	165	246	246
18	110	195	169	213
19	115	180	203	108
20	115	200	209	169
21	120	190	147	153
22	125	190	156	235
23	130	200	189	118
24	130	195	261	112
25	135	220	195	87
26	150	220	207	169
27	150	225	148	121
28	155	270	138	30
29	160	235	106	104
30	165	245	187	29
31	175	240	132	137

Continua na próxima página.

Tabela 14 — Continuação da página anterior.

aeronave (i)	a_i (min)	d_i (min)	qd_i	qa_i
32	180	250	172	48
33	180	250	181	102
34	185	255	38	113
35	185	260	185	211
36	190	375	168	188
37	190	270	178	89
38	190	260	104	140
39	195	280	138	101
40	220	295	102	221
41	225	290	173	101
42	230	320	202	153
43	230	290	139	35
44	235	300	264	197
45	240	310	193	124
46	240	300	178	149
47	240	310	150	114
48	240	315	124	75
49	245	310	73	98
50	245	310	95	118
51	250	340	187	62
52	270	370	33	166
53	280	370	155	104
54	280	355	115	129
55	290	430	171	110
56	295	370	195	102
57	295	380	147	277
58	300	385	150	124
59	305	370	155	101
60	305	400	194	156
61	310	400	185	121

Fonte: adaptado de Zhang, Xue e Jiang (2017) e ANAC (2017).

A Tabela 15 mostra as distâncias entre os *gates* considerados no GAP e as áreas de *check-in* e coleta de bagagem, em metros.

Tabela 15 – Distâncias entre *gates* e áreas de *check-in* e coleta de bagagem.

gate (j)	wd_j (m)	wa_j (m)
1	350	1235
2	310	1145
3	220	1055
4	310	965
5	400	875
6	490	785
7	580	695
8	670	605
9	760	515
10	850	425
11	940	335
12	1030	75
13	1120	165
14	1210	255
15	1300	345
apron ($m+1$)	1340	385

Fonte: adaptado de Zhang, Xue e Jiang (2017) e ANAC (2017).