UNIVERSIDADE FEDERAL DE OURO PRETO
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CIVIL
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA CIVIL

ALGORITMOS DE OTIMIZAÇÃO MULTIOBJETIVO PARA O PROBLEMA DE SEQUENCIAMENTO DE ATIVIDADES EM PROJETOS DE CONSTRUÇÃO METÁLICA

HELTON CRISTIANO GOMES

Tese apresentada ao Programa de Pós-Graduação do Departamento de Engenharia Civil da Escola de Minas da Universidade Federal de Ouro Preto, como parte integrante dos requisitos para obtenção do título de Doutor em Engenharia Civil, área de concentração: Construção Metálica.

Orientador: Prof. Dr. Francisco de Assis das Neves
Co-orientador: Prof. Dr. Marcone Jamilson Freitas Souza

Ouro Preto, Outubro de 2013.

G633a Gomes, Helton Cristiano.
Algoritmos de otimização multiobjetivo para o problema de sequenciamento de atividades em projetos de construção metálica [manuscrito] / Helton Cristiano Gomes. - 2013.
ix, 115f.: il. color.; grafs.; tabs.

Orientador: Prof. Dr. Francisco de Assis das Neves.
Coorientador: Prof. Dr. Marcone Jamilson Freitas Souza.

Tese (Doutorado) - Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia Civil. Programa de Pós Graduação em Engenharia Civil.
Área de concentração: Construção Metálica.

1. Construção metálica - Teses. 2. Otimização combinatória - Teses. 3. Sequências (Matemática) - Teses. 4. Projetos de engenharia – Teses. I. Neves, Francisco de Assis das. II. Souza, Marcone Jamilson Freitas. III. Universidade Federal de Ouro Preto. IV. Título.

CDU: 624.014.2:519.165

Catalogação: sisbin@sisbin.ufop.br

ALGORITMOS DE OTIMIZAÇÃO MULTIOBJETIVO PARA O PROBLEMA DE SEQUENCIAMENTO DE ATIVIDADES EM PROJETOS DE CONSTRUÇÃO METÁLICA

AUTOR: HELTON CRISTIANO GOMES

Esta tese foi apresentada em sessão pública e aprovada em 10 de outubro de 2013, pela Banca Examinadora composta pelos seguintes membros:

Prof. Dr. Francisco de Assis das Neves (Orientador / UFOP)

Prof. Dr. Marcone Jamilson Freitas Souza (Orientador / UFOP)

Prof. Dr. Ricardo Azoubel da Mota Silveira (UFOP)

Prof. Dr. Hélio José Corrêa Barbosa (UFJF)

Prof. Dr. Rodney Rezende Saldanha (UFMG)

AGRADECIMENTOS

Gostaria de agradecer, primeiramente, à Deus pelo dom da vida.

Ao meu PAI (Aloísio) e meus AVÓS (Miguelão e Dona Bibi), que onde quer que estejam sempre torcerão por mim.

À minha mãe Dona Conceição, pelo apoio, compreensão e incentivos constantes, mesmo nos momentos mais difíceis.

Aos meus irmãos, Bruno e Júnior, pela amizade e companherismo.

À Flávia, principal incentivadora de tudo o que eu faço. Obrigado Linda! E à sua família pelo carinho que têm por mim.

Aos Professores Assis e Marcone pelas preciosas orientações, imprescindíveis para a realização deste trabalho. Muito obrigado pela amizade e confiança.

À UFOP pela oportunidade e por me dar condições para vencer mais essa etapa. Instituição do qual tenho o maior orgulho em ter estudado.

Aos professores do PROPEC pelos conselhos e ensinamentos que de uma forma ou de outra contribuíram para a realização deste trabalho.

À UFV pelo incentivo e apoio na realização dos meus estudos.

Aos meus bons e velhos amigos pelo companheirismo e força.

À Róvia pela atenção e por seu eficiente trabalho na secretaria do PROPEC.

RESUMO

Com o atual crescimento do mercado imobiliário, os recursos produtivos tendem a se tornar escassos e caros na construção civil. Devido a esse fato, a melhor utilização dos recursos produtivos se tornou de extrema importância para o sucesso desse tipo de empreendimento.

Outro fato importante é a crescente utilização do aço na construção civil, substituindo materiais convencionais como o concreto. Esse fato se deve às vantagens estéticas e de qualidade que esse tipo de construção vem apresentando em diversos tipos de projetos. Porém, além dessas vantagens proporcionadas pela utilização de sistemas construtivos em aço, a redução do tempo e do custo de construção e o aumento da produtividade são fatores-chave para o seu sucesso. No entanto, para se alcançar esses fatores, as obras precisam ser muito mais controladas, o que significa projetos mais bem elaborados onde a tecnologia está sendo um diferencial para as empresas que investem nela.

A falta e/ou mau planejamento e orientação no gerenciamento de projetos têm sido os principais responsáveis por problemas que ocorrem na construção civil. Um correto gerenciamento de projetos é capaz de propiciar a redução de prazos e custos, a melhor utilização dos recursos produtivos, a minimização de riscos e a redução de erros no processo produtivo.

Diversas ferramentas podem ser utilizadas pela engenharia no auxilio à tomada de decisões relativas ao gerenciamento de projetos, dentre elas destaca-se a otimização, ainda pouco aplicada na construção civil. Vários problemas de otimização relacionados a projetos, que se enquadram em diversas aplicações reais, podem ser encontrados na literatura. Um importante exemplo é o problema de sequenciamento de atividades em projetos com restrições de recursos e de precedência (PSAPRRP), uma vez que o correto sequenciamento das atividades de um projeto resulta em um melhor aproveitamento dos recursos disponíveis e, consequentemente, ganho de produtividade e tempo.

Neste trabalho, o PSAPRRP é abordado como um problema de otimização multiobjetivo, tendo como meta a minimização de dois critérios: a data de finalização do projeto e o somatório dos custos associados às datas de início de execução das atividades. Para a resolução do problema, são propostos cinco algoritmos multiobjetivos, baseados nos métodos *Multi-objective* GRASP (GMO), *Multi-objective Variable Neighborhood Search* (MOVNS) e *Pareto Iterated Local Search* (PILS). Os algoritmos propostos utilizam estratégias baseadas no conceito de dominância de Pareto para realizar a busca de soluções e determinar um conjunto de soluções não-dominadas próximo ao conjunto Pareto-ótimo, permitindo aos projetistas a escolha de uma solução que satisfaça seus interesses, tornando o projeto mais planejado e controlado.

Os conjuntos de soluções não-dominadas obtidos pelos algoritmos, para um conjunto de instâncias adaptadas da literatura, são comparados utilizando quatro métricas de avaliação de desempenho: medidas de distância, diferença de hipervolume, *epsilon* e taxa de erro. Foram realizados, também, experimentos estatísticos para comprovar a existência de diferença significativa entre os algoritmos propostos com relação às métricas utilizadas.

Por fim, com o intuito de exemplificar a aplicação dos cinco algoritmos, é proposto um exemplo fictício e simplificado de um projeto de construção civil utilizando estruturas metálicas. Com base nos resultados obtidos pelos algoritmos para dois cenários do exemplo, é apresentada uma análise acerca da influência da disponibilidade de recursos com relação aos objetivos adotados.

Palavras-chave: Construção Metálica, Gerenciamento de Projetos, Sequenciamento de Atividades, Otimização Multiobjetivo, Métodos Metaheurísticos.

ABSTRACT

With the current real estate market growth, the productive resources tend to become scarce and expensive in civil construction. Due to this fact, the best use of productive resources has become extremely important for the success of this type of enterprise.

Another important fact that has been observed is the increasing use of steel in civil construction, replacing conventional materials like concrete. This fact is due to the aesthetic and quality advantages that this type of construction has been showing in various types of projects. However, besides these advantages provided by the use of steel construction systems, the reduction of the duration and cost of construction and the increased productivity are key factors for its success. However, to achieve these factors, the works need to be much more controlled, which means projects better elaborate where technology is a differential for companies that invest in it.

The lack of planning and/or bad planning and guidance on project management have been the main responsible for problems that happen on civil construction. Correct project management is capable of providing reduction of duration and costs, better utilization of productive resource, minimization of risks, and error reduction on the production process.

Several tools can be used by engineering aiding the decision making related to project management, within which optimization is emphasized; this is seldom applied in civil construction. Several optimization problems related to projects, which has a wide diversity of real applications, can be found in literature. An important example is the resource-constrained project scheduling problem with precedence relation (RCPSPRP), considering that the correct project activities sequencing results in a better use of the available resources, and consequently, a gain in productivity and time.

In this work the RCPSPRP is addressed as a multi-objective optimization problem and aims at minimizing two criteria: the makespan and the total weighted start time of the activities. To solve the problem, five multi-objective algorithms are analyzed, based on Multi-objective GRASP (*MOG*), Multi-objective Variable Neighborhood Search (*MOVNS*) and Pareto Iterated Local Search (*PILS*) methods. The proposed algorithms use strategies based on the concept of Pareto Dominance to search for solutions and determine the set of non-dominated solutions close to the Pareto-optimal front, allowing the project manager to choose a solution that fulfills his interests by making the project better planned and controlled.

The set of non-dominated solutions attained by the algorithms for a set of instances adapted from literature are compared using four multi-objective performance measures: distance metrics, hypervolume indicator, *epsilon* metric and error ratio. Statistical experiments were also conducted to prove the existence of a significant difference between the proposed algorithms regarding the used metrics.

Finally, in order to illustrate the application of the five algorithms, a fictitious and simplified example of a civil construction project, using steel structures, is proposed. Based on the results obtained by the algorithms for two scenarios of the proposed example, an analysis about the influence of resource availability with respect to adopted objectives is presented.

Keywords: Steel Construction, Project Management, Scheduling, Multi-objective Optimization, Metaheuristic Methods.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AGs – Algoritmos Genéticos

ANOVA – Teste Estatístico de Análise de Variância

BT – Busca Tabu

BTMO – Busca Tabu Multiobjetivo

GMO – GRASP Multiobjetivo

GMOVNS – GRASP + VNS Multiobjetivo

GRASP – *Greed Randomized Adaptive Procedure*

ILS – *Iterated Local Search*

MDS – Método da Mínima Diferença Significativa (Método de Fisher)

MOVNS – VNS Multiobjetivo

MOVNS_I – VNS Multiobjetivo com Intensificação

MPGS – Método Paralelo de Geração de Sequenciamentos

MSGS – Método Serial de Geração de Sequenciamentos

NSGA-II – *Nondominated Sorting Genetic Algorithm II*

OC – Otimização Combinatória

OM – Otimização Multiobjetivo

PILS – Pareto *Iterated Local Search*

PSA – Pareto *Simulated Annealing*

PSAP – Problema de Sequenciamento de Atividades em Projetos

PSAPRRP – Problema de Sequenciamento de Atividades em Projetos com Restrições de Recursos e de Precedência

PSPLib *– Project Scheduling Problem Library*

SA *– Simulated Annealing*

SPEA-II – *Strength Pareto Evolutionary Algorithm II*

VNS – *Variable Neighborhood Search*

Capítulo 1

1. INTRODUÇÃO

1.1. Considerações Gerais e Objetivos

Com o atual crescimento do mercado imobiliário, os recursos produtivos, como, por exemplo, a mão de obra e a matéria-prima, tendem a se tornar escassos e caros na construção civil. Esse fato está tornando cada vez mais atrativa a utilização de novas tecnologias na busca pela redução de erros e falhas no processo construtivo.

A falta de projetos bem elaborados tem sido uma das principais responsáveis por erros e falhas que surgem na construção civil. Grande parte do retrabalho e dos desperdícios que ocorrem nesse setor são consequências da falta e/ou mau planejamento e orientação no gerenciamento dos projetos. O gerenciamento de projetos tem por objetivo, dentre outros, propiciar a redução de prazos e custos, minimização de riscos e redução de erros no processo produtivo.

Outro fato importante, que tem se observado, é a crescente utilização do aço na construção civil, substituindo materiais convencionais como o concreto. A competitividade da construção metálica tem possibilitado a utilização do aço em diversos tipos de projetos, apresentando vantagens estéticas e de qualidade. Porém, além destas vantagens proporcionadas pela utilização de sistemas construtivos em aço, a redução do tempo e do custo de construção, a melhor utilização dos recursos produtivos e o aumento da produtividade são fatores-chave para o sucesso desse tipo de empreendimento. Entretanto, esses fatores não são possíveis de serem alcançados sem se ter obras muito mais controladas, o que significa projetos melhores onde a tecnologia está sendo um diferencial para as empresas que investem nela.

Diversas classes de ferramentas podem auxiliar a engenharia na tomada de decisões relativas ao gerenciamento de projetos, dentre as quais destaca-se a otimização. Porém, tal classe de ferramentas ainda é pouco aplicada em projetos de construção civil.

Vários problemas de otimização relacionados a projetos, que se enquadram em diversas aplicações reais, podem ser encontrados na literatura. Um importante exemplo é o problema de sequenciamento de atividades em projetos, onde podem ser consideradas ou não restrições de recursos, de tempo e de precedência entre as atividades. Um correto sequenciamento das atividades de um projeto resulta em um melhor aproveitamento dos recursos disponíveis e, consequentemente, ganho de tempo e produtividade.

O problema de sequenciamento de atividades em projetos com restrições de recursos e de precedência (PSAPRRP) consiste em, dados um conjunto de *n* atividades e outro, com *m* recursos recuperáveis e com disponibilidades pré-definidas, determinar a data de início de execução de cada uma das *n* atividades, assegurando que o nível de recursos e as relações de precedência não sejam violados. A execução das atividades possui uma duração pré-determinada e demanda por recursos. O principal objetivo ao sequenciar as atividades é completar o projeto dentro do orçamento e do prazo previstos, satisfazendo, assim, a todos os seus participantes.

Considerando que os envolvidos em projetos frequentemente buscam finalizá-los o mais rápido possível com o mínimo custo e com máxima qualidade, o PSAPRRP é claramente um problema de otimização multiobjetivo (OM). Apesar de vários autores o considerarem um problema cuja resolução envolve diversos e conflitantes objetivos, poucos trabalhos têm sido desenvolvidos utilizando este enfoque. Em problemas de OM que envolvem objetivos conflitantes entre si, a melhoria de algum(uns) dele(s) pode

causar, consequentemente, a piora de outro(s). Essa situação ocorre em muitos problemas encontrados no mundo real.

A OM oferece vantagens em relação à otimização mono-objetivo devido ao fato de gerar soluções com diferentes contrapartidas entre distintos objetivos. Contudo, é necessário que o tomador de decisões (projetista) escolha a solução que melhor atenda às necessidades do projeto. Sendo assim, é muito importante a escolha de uma técnica que determine o conjunto de soluções mais adequado para auxiliar o projetista.

Os principais métodos de resolução de problemas de OM podem ser classificados em dois grupos: os métodos clássicos e os metaheurísticos. Nos métodos clássicos, a definição dos critérios de geração de soluções ocorre antes da execução da mesma. Já a classe de métodos metaheurísticos envolve o mapeamento de soluções viáveis, direcionando para soluções Pareto-ótimas.

Visando auxiliar os projetistas (tomadores de decisões) da construção metálica no gerenciamento de projetos de obras, tornando-as mais planejadas e controladas, este trabalho tem como objetivo desenvolver algoritmos eficientes para a resolução do PSAPRRP formulado como um problema de OM. Neste trabalho, o PSAPRRP é abordado tendo como meta a minimização de dois objetivos conflitantes: a data de finalização do projeto (*makespan*) e o somatório dos custos associados às datas de início de execução das atividades.

Devido ao fato de o PSAPRRP ser de difícil resolução no caso geral, é importante ressaltar que, para problemas com dimensões elevadas como nos casos reais, os métodos clássicos podem ser menos eficientes computacionalmente. Visto isso e, em vista da pouca aplicação de métodos metaheurísticos ao PSAPRRP multiobjetivo, são propostos neste trabalho cinco algoritmos multiobjetivos para sua resolução: um *Multi-objective* GRASP (GMO), denominado GMO_PSAP, um *Multi-objective* VNS (MOVNS), denominado MOVNS_PSAP, um *Multi-objective* GRASP utilizando o VNS como busca local, denominado GMOVNS_PSAP, um *Multi-objective* VNS com intensificação, denominado MOVNS_I_PSAP e um Pareto ILS (PILS), denominado PILS_PSAP.

Para avaliar a eficiência dos algoritmos implementados, os resultados obtidos através da utilização de adaptações de instâncias encontradas na literatura foram comparados através de quatro métricas de avaliação de desempenho: medidas de distância, diferença de hipervolume, *epsilon* e taxa de erro. Foram realizados, também, experimentos estatísticos com o intuito de verificar se há diferença significativa entre os algoritmos com relação às métricas utilizadas.

Por fim, com o propósito de exemplificar a aplicação dos algoritmos, é proposto um exemplo fictício e simplificado de um projeto de construção civil utilizando estruturas metálicas. Com base nos resultados obtidos pelos algoritmos para dois cenários do exemplo, é apresentada uma análise acerca da influência da disponibilidade de recursos com relação aos objetivos adotados.

Vale a pena ressaltar, também, que o desenvolvimento deste trabalho contribuirá para a divulgação de técnicas de Pesquisa Operacional nas empresas de construção metálica, possibilitando, dessa forma, a melhoria nos seus processos produtivos.

1.2. Organização do Trabalho

O restante desta tese está organizado como segue. O Capítulo 2 descreve as principais características do processo construtivo em aço, bem como suas principais atividades. Este capítulo expõe, também, a importância de um bom gerenciamento de projetos na construção civil, destacando os erros e falhas que podem surgir em uma obra, causados por um projeto mal elaborado.

O Capítulo 3 apresenta o PSAPRRP, um esquema de classificação, modelos matemáticos e métodos de resolução para o mesmo.

O Capítulo 4 descreve a OM, o conceito de Ótimo de Pareto e métodos clássicos e metaheurísticos de resolução de problemas de OM. Tal capítulo apresenta, também, algumas aplicações da OM encontradas na literatura e exemplos de aplicação de dois métodos clássicos ao PSAPRRP.

O Capítulo 5 dedica-se à apresentação da formulação multiobjetivo utilizada para o PSAPRRP e dos cinco algoritmos propostos para a resolução do mesmo.

O Capítulo 6 descreve as instâncias e as métricas de avaliação de desempenho utilizadas. No Capítulo 6 são apresentados, também, os resultados obtidos pelos cinco algoritmos, descritos no capítulo 5, para as métricas, bem como os resultados obtidos pelos experimentos estatísticos.

No Capítulo 7 é apresentada a aplicação dos cinco algoritmos propostos em um exemplo fictício de um projeto de construção civil utilizando estruturas metálicas. É apresentada, também, uma análise acerca da influência da disponibilidade de recursos com relação aos objetivos adotados, utilizando dois cenários para o exemplo.

Finalizando o trabalho, no Capítulo 8, algumas considerações e conclusões referentes ao trabalho são estabelecidas. São fornecidas, também, algumas sugestões para o desenvolvimento de trabalhos futuros.

Capítulo 2

2. GERENCIAMENTO DE PROJETOS DE CONSTRUÇÃO METÁLICA

2.1. Processos Construtivos com Estruturas Metálicas

Segundo Inaba (2009), desde o século XVIII, quando se iniciou a utilização de estruturas metálicas na construção civil, o aço tem possibilitado aos arquitetos, engenheiros e construtores soluções arrojadas, eficientes e de alta qualidade. A competitividade da construção metálica tem possibilitado a utilização do aço em obras como edifícios de escritórios e apartamentos, residências, habitações populares, pontes, passarelas, viadutos, galpões, supermercados, *shopping centers*, lojas, postos de combustíveis, aeroportos e terminais rodoferroviários, ginásios e arenas esportivas, torres de transmissão, etc.

Ainda de acordo com Inaba (2009), nos dias atuais é visível a crescente utilização do aço em substituição a materiais convencionais como o concreto. Segundo o autor, o sistema construtivo em aço apresenta vantagens significativas sobre o sistema construtivo convencional, em vista dos seguintes aspectos:

- **Liberdade no projeto de arquitetura** - a tecnologia do aço confere aos arquitetos total liberdade criadora, permitindo a elaboração de projetos arrojados e de expressão arquitetônica marcante;
- **Maior área útil** - as seções dos pilares e vigas de aço são substancialmente mais esbeltas do que as equivalentes em concreto, resultando em melhor aproveitamento do espaço interno e aumento da área útil, fator muito importante principalmente em garagens;
- **Flexibilidade** - a estrutura metálica mostra-se especialmente indicada nos casos onde há necessidade de adaptações, ampliações, reformas e mudança de ocupação de edifícios. Além disso, torna mais fácil a passagem de utilidades como água, ar condicionado, eletricidade, esgoto, telefonia, informática, etc.;
- **Compatibilidade com outros materiais** - o sistema construtivo em aço é perfeitamente compatível com qualquer tipo de material de fechamento, tanto vertical como horizontal, admitindo desde os mais convencionais (tijolos e blocos, lajes moldadas *in loco*) até componentes pré-fabricados (lajes e painéis de concreto, painéis *dry-wall*, etc.);
- **Menor prazo de execução** - a fabricação da estrutura em paralelo com a execução das fundações, a possibilidade de se trabalhar em diversas frentes de serviços simultaneamente, a diminuição de formas e escoramentos e o fato da montagem da estrutura não ser afetada pela ocorrência de chuvas, pode levar a uma redução de até 40% no tempo de execução quando comparado com os processos convencionais;
- **Racionalização de materiais e mão de obra** – em uma obra por meio de processos convencionais, o desperdício de materiais pode chegar a 25% em peso. A estrutura metálica possibilita a adoção de sistemas industrializados, fazendo com que o desperdício seja sensivelmente reduzido;
- **Alívio de carga nas fundações** - por serem mais leves, as estruturas metálicas podem reduzir em até 30% o custo das fundações;

- **Garantia de qualidade** - a fabricação de uma estrutura metálica ocorre dentro de uma indústria e conta com mão de obra altamente qualificada, o que dá ao cliente a garantia de uma obra com qualidade superior devido ao rígido controle existente durante todo o processo industrial;
- **Antecipação do ganho** - em função da maior velocidade de execução da obra, haverá um ganho adicional pela ocupação antecipada do imóvel e pela rapidez no retorno do capital investido;
- **Organização do canteiro de obras** - como a estrutura metálica é totalmente pré-fabricada, há uma melhor organização do canteiro devido, entre outros fatores, à ausência de grandes depósitos de areia, brita, cimento, madeiras e ferragens, reduzindo também o inevitável desperdício desses materiais. O ambiente limpo, com menor geração de entulho, oferece ainda melhores condições de segurança ao trabalhador contribuindo para a redução dos acidentes na obra;
- **Reciclabilidade** - o aço é 100% reciclável e as estruturas podem ser desmontadas e reaproveitadas;
- **Preservação do meio ambiente** - a estrutura metálica é menos agressiva ao meio ambiente, pois, além de reduzir o consumo de madeira na obra, diminui a emissão de material particulado e poluição sonora geradas pelas serras e outros equipamentos destinados a trabalhar a madeira;
- **Precisão construtiva** - enquanto nas estruturas de concreto a precisão é medida em centímetros, numa estrutura metálica a unidade empregada é o milímetro. Isso garante uma estrutura perfeitamente aprumada e nivelada, facilitando atividades como o assentamento de esquadrias, instalação de elevadores, bem como redução no custo dos materiais de revestimento.

Segundo Inaba (2009), além das vantagens relacionadas à estética e à qualidade proporcionadas pela utilização de sistemas construtivos em aço, a redução do tempo e do custo de construção, a melhor utilização dos recursos produtivos e o aumento da produtividade passaram a ser fatores-chave para o sucesso de qualquer empreendimento. Porém, segundo Hendrickson (2008), não é possível reduzir prazos e custos e aumentar a produtividade sem ter uma obra bem planejada e controlada, o que significa projetos bem gerenciados, em que a tecnologia esta sendo um diferencial para as empresas que investem nela.

2.2. Gerenciamento de Projetos

De acordo com o Instituto de Gerenciamento de Projetos (PMI, do inglês *Project Management Institute*), um projeto pode ser definido como um esforço temporário empreendido para criar um produto ou serviço único. Segundo Keelling (2002), na maioria das vezes, isso implica em um prazo limitado, com uma data estipulada para conclusão. Hamm *et al*. (2009) definem um projeto como inúmeras atividades que devem ser eficientemente sequenciadas e realizadas para atender a diferentes, e às vezes conflitantes, objetivos, como tempo, custo e qualidade.

Ainda de acordo com o PMI, o gerenciamento de projetos consiste na aplicação de conhecimentos, habilidades e técnicas para a elaboração de atividades com a finalidade de alcançar um conjunto de objetivos pré-definidos em certo prazo e com certo custo e qualidade, através da utilização de recursos técnicos e humanos. Segundo o Instituto, no início dos anos de 1950 foi dado início à utilização de técnicas que envolvem o uso de *softwares* e ferramentas para aumentar o sucesso dessa gestão.

Segundo Odedairo e Oladokum (2011), o gerenciamento de projetos envolve o planejamento, o sequenciamento, o monitoramento e o controle das atividades do projeto, propiciando, dessa forma, o alcance dos objetivos propostos. De acordo com os autores, os principais objetivos a serem alcançados ao gerenciar um projeto são, dentre outros, a redução de prazos e custos, a minimização de riscos e de erros no processo produtivo e a melhoria na utilização dos recursos produtivos.

Grande parte do retrabalho e dos erros que ocorrem na construção civil são consequências da falta e/ou mau planejamento e orientação no gerenciamento dos projetos, conforme descrito na seção a seguir.

2.2.1. Falhas no Gerenciamento de Projetos de Construção Civil

Segundo estudos realizados, a maior parte dos problemas que ocorrem na construção civil, incluindo a construção metálica, tem sua origem na etapa de projetos. Cambiaghi (1992) concluiu que a falta de projetos adequados é a principal responsável pelos fatores que contribuem para erros e falhas na construção civil. Segundo Picchi (1993), uma parcela de 6% do custo de uma obra corresponde à elaboração de projetos não otimizados.

Maciel e Melhado (1996) mostraram que, dentre as causas de patologias nas edificações, deficiências e erros de projetos são responsáveis por 60% dos problemas patológicos, como pode ser visto na Tabela 1.

Tabela 1: Principais causas de problemas patológicos na construção civil

Origem do Problema	**Índice** (%)
Projeto	60,0
Construção	26,4
Equipamentos	2,1
Outros	11,5

Fonte: Maciel e Melhado (1996)

Devido ao crescimento do mercado imobiliário, a construção civil vem ganhando, cada vez mais, espaço na formação do Produto Interno Bruto (PIB) brasileiro e mundial. Com o crescimento da atividade de construção, os recursos produtivos, como a mão de obra e a matéria-prima, tendem a se tornar escassos e caros. De acordo Picchi (1993), devido a estes fatos a construção civil precisa se adequar, buscando formas mais eficientes de planejamento, execução e controle dos seus projetos.

2.3. Planejamento de Projetos de Construção Metálica

De acordo com Hendrickson (2008), o planejamento é uma fundamental e desafiante atividade no gerenciamento de um projeto de construção. Ele envolve a escolha da tecnologia a ser utilizada, a definição das atividades do projeto, bem como suas durações e inter-relações, e a estimativa dos recursos necessários para a execução de cada atividade. Segundo o autor, com base em um bom planejamento, a definição do orçamento e a determinação da sequência em que as atividades do projeto deverão ser executadas são muito mais precisas, evitando desperdícios de recursos e de tempo.

A escolha da tecnologia apropriada afeta diretamente os custos e a duração das atividades, sendo um "elemento crítico" para o sucesso do projeto. Segundo Hendrickson (2008), para se determinar a melhor tecnologia a ser utilizada, é necessário analisar a confiabilidade das tecnologias alternativas, bem como seus impactos nos custos e na duração do projeto.

Ao mesmo tempo em que a escolha da tecnologia é considerada, um passo paralelo no processo de planejamento é a definição das inúmeras atividades que

precisam ser concluídas dentro do projeto. De acordo com Hendrickson (2008), a definição correta das atividades de um projeto pode ser trabalhosa visto que o projeto de uma construção pode envolver um grande número de atividades. A definição das atividades do projeto permite a estimativa da duração e dos recursos requeridos por cada uma, visto que as suas execuções requerem tempo e recursos. Vale ressaltar que a duração de uma mesma atividade pode variar de um projeto para outro.

Hendrickson (2008) propõe uma abordagem simples para a estimativa das durações das atividades: utilizar os registros históricos de atividades específicas e calcular a média destes registros fazendo a estimativa da nova duração. Essas durações são amplamente utilizadas na elaboração de cronogramas.

Com a definição dos recursos necessários para cada atividade, pode-se determinar o total de recursos necessários para o projeto. Pode-se, também, identificar a necessidade de determinado recurso ao longo do projeto, evitando, assim, potenciais gargalos. Sendo assim, os custos de execução das atividades estão ligados aos custos de aquisição e/ou utilização dos recursos e, portanto, também variam de um projeto para outro.

Os principais recursos demandados pelas atividades em um projeto de construção metálica são: mão-de-obra (engenheiros, mestres, montadores, soldadores/maçariqueiros, pintores, ajudantes, etc.); matéria prima (aço); materiais (tintas, solventes, soldas, parafusos, conectores, etc.); equipamentos (furadeiras, parafusadeiras, caminhões, carretas, tratores, escavadeiras, etc.); e máquinas (de corte, de pintura, torno, fresa, calandra, guindastes, etc.).

Outra definição importante, em relação às atividades, são suas relações de precedência. Uma vez definidas as atividades, a relação entre elas deve ser especificada. As relações de precedência determinam que algumas atividades precisam ser realizadas em uma sequência particular, ou seja, uma atividade não pode ser iniciada enquanto suas atividades precedentes não forem finalizadas. Essas relações podem ser dadas por algum requerimento de integridade estrutural, regulamentos e outros requerimentos técnicos. Três situações devem ser bem avaliadas na especificação das relações de precedência:

- criar um ciclo de atividades precedentes resulta em um planejamento inviável;

  Para exemplificar, supondo três atividades A, B e C, um ciclo de atividades do tipo apresentado na Figura 1 precisa ser evitado. Ou seja, se A precede B e, B precede C, então C não pode preceder A.

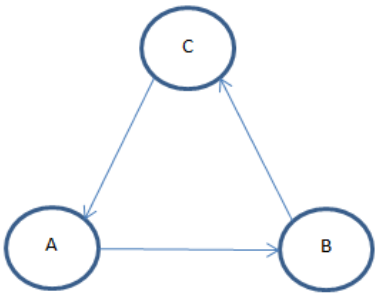

Figura 1: Exemplo de ciclo de atividades

- esquecer (desprezar) uma importante relação de precedência pode ser fatal;
- é importante perceber que as relações de precedência entre as atividades podem ser estabelecidas de formas diferentes e que cada uma tem uma implicação diferente no sequenciamento das atividades.

O processo construtivo utilizando estruturas metálicas envolve um grande número de atividades, que vão desde a escolha do tipo de aço a ser utilizado até a instalação dos itens de segurança. Nas seções a seguir são descritas as principais atividades da construção metálica.

2.3.1 Principais Atividades Envolvidas na Construção Metálica

De acordo com Bellei *et al*. (2008), uma obra com estruturas metálicas é o resultado de um conjunto de atividades que se inicia na elaboração do projeto (arquitetônico e estrutural), passa pelo detalhamento do mesmo (desenho de oficina), pela fabricação, limpeza e pintura das peças, seguida pelo transporte e montagem das mesmas e pela instalação de itens de proteção contra incêndio (se necessário).

Entretanto, as principais atividades envolvidas na construção civil utilizando estruturas metálicas podem ser agrupadas basicamente em duas etapas: a fabricação da estrutura metálica, que consiste nas atividades relativas à produção das peças a serem utilizadas na construção e a montagem da estrutura, que consiste nas atividades relacionadas à montagem das peças fabricadas. São apresentadas a seguir as principais atividades pertencentes a cada etapa.

A - Etapa de Fabricação

A etapa de fabricação parte de um projeto estrutural detalhado, no qual são definidas todas as peças que compõem a estrutura, inclusive todos os detalhes de encaixe e ligação. Como base neste projeto estrutural são fabricadas as peças, cujas principais atividades são:

- Traçagem: atividade que consiste em marcar a superfície do metal com as informações necessárias à fabricação;
- Corte: atividade destinada a deixar a peça no comprimento ou formato desejado;
- Acabamento e pré-deformação: atividade no qual faz-se um acabamento e uma pré-deformação das tiras obtidas no processo de corte das chapas, para compensar a deformação gerada pela soldagem;
- Solda: atividade no qual é feita a união definitiva entre as peças;
- Desempeno: atividade realizada após a soldagem para corrigir possíveis distorções;
- Dobra: atividade realizada para a fabricação de perfis ou chapas de ligação;
- Furação: atividade necessária quando as ligações entre as peças são feitas através de parafusos;
- Ponteamento: atividade que consiste na união provisória das peças menores sobre a principal, também chamada de montagem de fábrica ou pré-montagem;
- Processos auxiliares:
    - usinagem – atividade realizada quando necessita-se de um perfeito contato entre as superfícies das peças;
    - esmirilhamento – processo de desbaste para remoção de rebarbas e pontos de solda;
    - calandragem – processo de curvamento de chapas e perfis;
- Preparação da superfície: consiste nas atividades de remoção da crepa de laminação, dos respingos de solda, da ferrugem, das sujeiras, dos óleos, das graxas e de outros materiais contaminantes;
- Pintura: atividade cuja finalidade pode ser decorativa e/ou proteção contra a corrosão.

B - Etapa de Montagem

A etapa de montagem da estrutura metálica compreende a união das peças fabricadas utilizando parafusos e/ou soldas, com auxilio de ferramentas e equipamentos. Dentre as mais importantes atividades desta etapa pode-se destacar:

- Preparação das bases das colunas: atividades relacionadas à preparação das ligações das colunas e das bases;
- Nivelamento das colunas: atividades de posicionamento das colunas corrigindo-se eventuais desnivelamentos;
- Posicionamento dos componentes de desenvolvimento horizontal: atividades nos quais são posicionadas as vigas e as treliças;
- Estabilização do conjunto: atividades realizadas para que não ocorram perdas de estabilidade na estrutura;
- Ajustes: atividades de alinhamento, nivelamento e prumo da estrutura. Durante o processo de montagem, as ligações definitivas só serão executadas após o ajuste da estrutura;
- Execução das ligações definitivas: atividades de soldagem e/ou parafusamento definitivo das peças da estrutura.

Diversas outras atividades fazem parte da construção metálica como, por exemplo, a preparação do terreno (escavação, nivelamento, etc.) e da fundação, a definição e a instalação dos fechamentos verticais e horizontais (paredes e lajes), a montagem da cobertura (telhado) e de instalações (elétrica, hidráulica, de esgoto, etc.), a preparação e instalação dos pisos, a colocação das janelas e portas, as atividades de acabamento e retoques finais, a instalação de itens de segurança, dentre outras. As atividades que farão parte de um projeto específico devem ser definidas pelo projetista.

Com base na definição das atividades, juntamente com suas durações, necessidades de recursos e relações de precedência, pode-se determinar qual a melhor sequência em que essas devem ser realizadas. Segundo Adeli e Karin (2001), com esse sequenciamento os participantes do projeto podem monitorar e controlar o seu desenvolvimento. O correto sequenciamento das atividades de um projeto (SAP) resulta, também, em um melhor aproveitamento dos recursos disponíveis e, consequentemente, ganho de produtividade e tempo e redução de erros e falhas. Entretanto, a definição da melhor sequência em que as atividades devem ser executadas não é uma tarefa trivial devido às inúmeras sequências possíveis, sendo, dessa forma, necessária a utilização de algum tipo de ferramenta para a sua determinação.

Uma importante classe de ferramentas que pode auxiliar a engenharia no planejamento e controle de projetos de construção é a otimização. Segundo Geyer (2009), a otimização é uma poderosa ferramenta no suporte ao gerenciamento de projetos, mas ainda é pouco aplicada na construção civil. Ainda de acordo com o autor, a aplicação da otimização traz dois benefícios principais: o resultado obtido pode tanto levar a uma solução de melhora como a um conhecimento mais amplo sobre as possíveis soluções. Suas técnicas podem ser aplicadas nas mais diversas áreas de negócios.

Dentre os vários problemas de otimização relacionados a projetos que podem ser encontrados na literatura, destaca-se aqui o problema de sequenciamento de suas atividades (*Project Scheduling*). Segundo Oguz e Bala (1994), o problema de sequenciamento de atividades em projetos (PSAP) é importante e desafiador tanto para os profissionais responsáveis pela gestão de projetos como também para os

pesquisadores de áreas afins. De acordo com os autores, uma das razões de sua importância é por este ser um problema comum a um grande número de situações reais de tomada de decisão, tais como os problemas que surgem no gerenciamento de projetos na construção civil. O PSAP é desafiador, do ponto de vista teórico, por pertencer à classe de problemas de otimização combinatória NP-difíceis (Garey e Johnson, 1979).

No próximo capítulo são descritos, detalhadamente, o PSAP e alguns dos principais modelos e métodos de resolução encontrados na literatura para o mesmo.

Capítulo 3

3. O PROBLEMA DE SEQUENCIAMENTO DE ATIVIDADES EM PROJETOS

De acordo com Baker (1974), um problema de sequenciamento (*Scheduling Problem*) consiste na alocação de recursos ao longo do tempo para executar um conjunto de atividades. Segundo o autor, recursos são bens ou serviços, cuja disponibilidade é limitada ou não, e podem ser classificados em recuperáveis (ou renováveis) e não-recuperáveis (ou não-renováveis). Recursos recuperáveis são aqueles que, após serem utilizados na execução de uma atividade, podem ser utilizados em outras atividades. Para os recursos não-recuperáveis isso não acontece. Os exemplos mais comuns de recursos são: recuperáveis – máquinas, equipamentos e mão de obra; não-recuperáveis – matéria-prima.

As atividades podem ser definidas como trabalhos elementares cuja execução demanda um certo número de unidades de tempo e/ou recursos. Estas podem ser classificadas como *preemptivas* (quando sua execução pode ser interrompida e reiniciada algum tempo mais tarde) ou não *preemptivas* (uma vez iniciada, sua execução não pode ser interrompida e reiniciada). Em problemas de sequenciamento podem ser consideradas, também, relações de precedência entre as atividades, ou seja, a execução de uma atividade não pode ser iniciada enquanto as suas precedentes não forem finalizadas.

De acordo com Paula (2008), existem vários tipos de problemas de sequenciamento que se enquadram em diversas aplicações reais. Um caso específico é o PSAP, onde podem ser consideradas ou não restrições de recursos, de tempo e de precedência entre as atividades.

Um SAP é tradicionalmente definido como uma tabela com os horários ou datas (*timetable*) das execuções das atividades do projeto. Os recursos são alocados às atividades antes delas serem iniciadas e, então, o recurso é mantido ocupado pelo tempo de execução da mesma. O principal objetivo ao sequenciar as atividades é completar o projeto dentro do orçamento e do prazo previstos, satisfazendo, assim, a todos os seus participantes (Adeli e Karin, 2001).

Conforme descrito por Brucker *et al*. (1999), a grande variedade de PSAPs com características diferentes encontrada na literatura gerou a necessidade da criação de uma notação sistemática. Essa notação serviu como base para um esquema de classificação que é utilizado para facilitar a apresentação e a discussão do PSAP, além de permitir a imediata identificação das características do problema. Tal esquema de classificação é apresentado na seção a seguir.

3.1. Esquema de Classificação para Problemas de Sequenciamento de Atividades em Projetos

Graham *et al.* (1979) e Blazewicz *et al.*(1983) propuseram um esquema de classificação para problemas de sequenciamento em máquinas composto por três campos $\alpha \mid \beta \mid \gamma$. Demeulemeester e Herroelen (2002) estenderam esta classificação para o PSAP, propondo um esquema similar. Baseado em Demeulemeester e Herroelen (2002), o significado dos três campos é descrito nas seções a seguir.

3.1.1 Campo α: Características dos Recursos

O campo α representa as características dos recursos presentes no PSAP. Tal campo é composto por três parâmetros: α_1, α_2 e α_3.

O parâmetro $\alpha_1 \in \{^o, 1, m\}$ denota o número de tipos de recursos presentes no projeto, no qual:

- $\alpha_1 = {}^o$: nenhum tipo de recurso é considerado no PSAP;
- $\alpha_1 = 1$: um tipo de recurso é considerado;
- $\alpha_1 = m$: o número de tipos de recursos considerados é igual a *m*.

O parâmetro $\alpha_2 \in \{^o, 1, T, 1T, v\}$ representa as características dos tipos de recursos utilizados, onde:

- $\alpha_2 = {}^o$: ausência de especificação para qualquer tipo de recurso;
- $\alpha_2 = 1$: recursos recuperáveis,
- $\alpha_2 = T$: recursos não-recuperáveis;
- $\alpha_2 = 1T$: recursos recuperáveis e não-recuperáveis;
- $\alpha_2 = v$: parte dos recursos recuperáveis (ou não-recuperáveis) está disponível somente em períodos de tempo específicos.

Já o parâmetro $\alpha_3 \in \{^o, va\}$ descreve as características das disponibilidades dos recursos do PSAP, no qual:

- $\alpha_3 = {}^o$: recursos disponíveis em quantidades constantes;
- $\alpha_3 = va$: recursos disponíveis em quantidades variadas.

3.1.2 Campo β: Características das Atividades

O campo β especifica as características das atividades de um PSAP. Esse campo é composto por oito parâmetros: β_1, β_2, β_3, β_4, β_5, β_6, β_7 e β_8.

O parâmetro $\beta_1 \in \{^o, pmtn, pmtn\text{-}rep\}$ indica a possibilidade de as atividades serem interrompidas e depois reiniciadas (*preempção*), onde:

- $\beta_1 = {}^o$: a *preempção* não é permitida;
- $\beta_1 = pmtn$: preempções do tipo *preempt-resume* são permitidas;
- $\beta_1 = pmtn\text{-}rep$: preempções do tipo *preempt-repeat* são permitidas.

Preempções do tipo *preempt-resume* significam que a execução de uma atividade pode ser interrompida e reiniciada de onde parou. Já a preempção do tipo *preempt-repeat* significa que a execução de uma atividade pode ser interrompida, mas não pode ser reiniciada de onde parou, isto é, sua execução deve ser totalmente reiniciada.

O segundo parâmetro $\beta_2 \in \{^o, cpm, min, gpr\}$ descreve as relações de precedência entre as atividades, no qual:

- $\beta_2 = {}^o$: o problema não apresenta relações de precedência;
- $\beta_2 = cpm$: as atividades estão sujeitas a relações de precedências onde não é permitido intervalo entre elas;
- $\beta_2 = min$: as atividades estão sujeitas a relações de precedência no qual é imposto um valor mínimo para o intervalo entre elas;

- β_2 = *gpr*: as atividades estão sujeitas a relações de precedência onde é imposto um valor mínimo e um valor máximo para o intervalo entre elas.

O terceiro parâmetro $\beta_3 \in \{^o, r_j\}$ determina a data no qual a execução das atividades pode ser iniciada, onde:

- $\beta_3 = {}^o$: todas as atividades podem ser iniciadas na data zero;
- $\beta_3 = r_j$: cada atividade possui uma data mínima no qual poderá ocorrer o início de sua execução.

O parâmetro $\beta_4 \in \{^o, cont, d_j = d\}$ descreve a duração da execução das atividades do projeto, no qual:

- $\beta_4 = {}^o$: as execuções das atividades possuem durações inteiras arbitrárias;
- β_4 = *cont*: as execuções das atividades possuem durações contínuas arbitrárias;
- $\beta_4 = d_j = d$: a execução de todas as atividades possui duração igual a *d*.

O parâmetro $\beta_5 \in \{^o, \delta_j, \delta_n\}$ descreve prazos de finalização dentro do projeto, onde:

- $\beta_5 = {}^o$: não há nenhum prazo de finalização dentro do projeto;
- $\beta_5 = \delta_j$: prazos para finalização são impostos paras as atividades;
- $\beta_5 = \delta_n$: um prazo para finalização é imposto para o projeto.

O sexto parâmetro $\beta_6 \in \{^o, vr, disc, cont, int\}$ denota a natureza da demanda de recursos por parte das atividades do projeto, no qual:

- $\beta_6 = {}^o$: as atividades demandam por recursos em quantidades constantes, ou seja, a demanda por recursos é a mesma em qualquer período de tempo em que a atividade seja executada;
- β_6 = *vr*: as atividades demandam por recursos em quantidades variadas, ou seja, a demanda por recursos varia de acordo com o período de tempo em que a atividade será executada;

Para os casos em que a demanda das atividades por recursos são determinadas em função de algum fator, as seguintes configurações são utilizadas:

- β_6 = *disc*: a demanda por recursos é uma função discreta em relação à duração da execução da atividade;
- β_6 = *cont*: a demanda por recursos é uma função continua em relação à duração da execução da atividade;
- β_6 = *int*: a demanda por recursos de uma atividade é expressa por uma função.

O tipo e o número de possíveis modos de execução das atividades de um projeto são descritos pelo parâmetro $\beta_7 \in \{^o, mu\}$, onde:

- $\beta_7 = {}^o$: as atividades são executadas utilizando apenas um modo;
- β_7 = *mu*: as atividades possuem vários, pré-especificados, modos de execução.

O último parâmetro $\beta_8 \in \{^o, c_j, per, sched\}$ é utilizado para descrever a implicação financeira das atividades do projeto, no qual:

- $\beta_8 = ^o$: nenhum fluxo financeiro é especificado no problema;
- $\beta_8 = c_j$: são associados às atividades fluxos financeiros arbitrários;
- $\beta_8 = c^+_j$: são associados às atividades fluxos financeiros positivos;
- $\beta_8 = per$: fluxos financeiros periódicos são especificados para o projeto, isto é, pagamentos em intervalos regulares;
- $\beta_8 = sched$: tanto o montante quanto os prazos do fluxo financeiro têm que ser determinados.

3.1.3 Campo γ: Medidas de Desempenho

O terceiro campo γ é reservado para denotar o critério de otimalidade (medida de desempenho). As medidas de desempenho mais comuns são:

- $\gamma = C_{max}$: minimizar a data de finalização (*makespan*) do projeto;
- $\gamma = L_{max}$: minimizar o atraso no projeto;
- $\gamma = n_T$: minimizar o número de atividades atrasadas;
- $\gamma = av$: minimizar a disponibilidade de recursos para finalizar o projeto.

Os exemplos acima são utilizados para funções mono-objetivo. Problemas com múltiplos objetivos podem ser representados por $\gamma = multi$.

O foco deste trabalho é o PSAP sujeito a restrições de recursos e de precedência entre as atividades (PSAPRRP). O PSAPRRP envolve o sequenciamento de atividades sujeito a restrições de precedência sem intervalo entre elas e de recursos recuperáveis constantes. As atividades possuem um único modo de execução e duração inteira pré-fixada e, suas demandas pelos recursos são as mesmas em qualquer período de tempo. A preempção não é permitida. O critério de otimalidade mais utilizado é a minimização do *makespan* do projeto. O PSAPRRP, descrito na seção a seguir, é, então, denotado como $m, 1| cpm | C_{max}$.

3.2. O Problema de Sequenciamento de Atividades em Projetos com Restrições de Recursos e de Precedência

O PSAPRRP consiste em, dados um conjunto de *n* atividades e outro, com *m* recursos recuperáveis e com disponibilidades pré-definidas, determinar a data de início de execução de cada uma das *n* atividades, assegurando que o nível de recursos e as relações de precedência não sejam violados. A execução das atividades possui uma duração pré-determinada e demanda por recursos. O objetivo mais comum utilizado na resolução do PSAPRRP é a minimização do *makespan* do projeto.

O PSAPRRP motivou o estudo de diversos autores como Balas (1967), Brucker *et al.* (1998), Brucker *et al*. (1999), Brucker e Knust (2000), Carlier e Néron (2003), Liu e Wang (2007), Rogalska *et al.* (2008), Christodoulou (2009), Koné *et al.* (2009) e König e Beibert (2009), dentre outros.

Vários modelos matemáticos são apresentados na literatura para o PSAPRRP, como, por exemplo, os descritos por Brucker *et al*. (1999) e Koné *et al.* (2009). Genericamente, o PSAPRRP pode ser assim formulado:

Min S_{n+1} (1)

Sujeito a

$$S_j \geq S_i + p_i \quad \forall (i, j) \in E \tag{2}$$

$$\sum_{i \in P(t)} b_{ik} \leq B_k \quad \forall k \in R;\ \forall t \in H \tag{3}$$

$$S_i \geq 0 \quad \forall i \in A \tag{4}$$

em que:

A: conjunto de atividades, $A = \{1, ..., n\}$;

R: conjunto de recursos, $R = \{1, ..., m\}$;

E: conjunto de pares de atividades (i, j), no qual a atividade *i* é predecessora da atividade *j*;

H: horizonte de sequenciamento, $H = \{1, ..., T\}$ onde *T* é a data máxima (limite superior) proposta para a finalização do projeto;

S_i: variável de decisão representando a data de início da execução da atividade *i*;

S_{n+1}: data de finalização do projeto;

p_i: tempo de execução da atividade *i*;

b_{ik}: demanda da atividade *i* pelo recurso *k*;

B_k: disponibilidade do recurso *k*;

$P(t)$: conjunto de atividades em execução no instante de tempo *t*, $P(t) = \{i \in A | S_i \leq t < S_i + p_i\}$.

A função objetivo (1) indica a minimização da data de finalização do projeto (S_{n+1}). As restrições do tipo (2) estabelecem as relações de precedência entre as atividades, isto é, se $(i, j) \in E$ então o início da execução da atividade *j* só pode ocorrer após a finalização da execução da atividade *i* ($S_i + p_i$). As restrições (3) garantem que a disponibilidade de cada recurso não seja violada em cada período de tempo *t* ($t = 1,..., T$). As restrições (4) estabelecem o domínio das variáveis. A solução do PSAPRRP é dada por uma lista com as datas de início das execuções das atividades $S = \{S_1, S_2,..., S_n\}$.

O PSAPRRP pode ser representado, também, através de um grafo acíclico direcionado $G = \{A, E\}$, onde *A* (conjunto de nós) representa as atividades e *E* (conjunto de arcos) representa as relações de precedência. Nesse contexto, os arcos do conjunto *E* representam somente as inter-relações entre as atividades. As atividades são numeradas de 0 a $n+1$, onde as atividades 0 e $n+1$ são atividades artificiais representando, respectivamente, o início e o final do projeto. Para exemplificar a representação gráfica, considere o problema descrito na Tabela 2 (Koné *et al.*, 2009).

Tabela 2: Dados do problema

Atividades	1	2	3	4	5	6	7	8	9	10
Sucessoras	3	6, 7	4, 9	11	1	1	5, 8	10	4	9

Fonte: Koné *et al.* (2009)

No problema apresentado na Tabela 2, onde o projeto é composto por 10 atividades, a atividade 1 possui somente a atividade 3 como dependente de sua execução, ou seja, é sua sucessora. Já a atividade 2 tem como sucessoras as atividades 6

e 7. E assim sucessivamente. O grafo relativo ao problema descrito na Tabela 2 é apresentado na Figura 2.

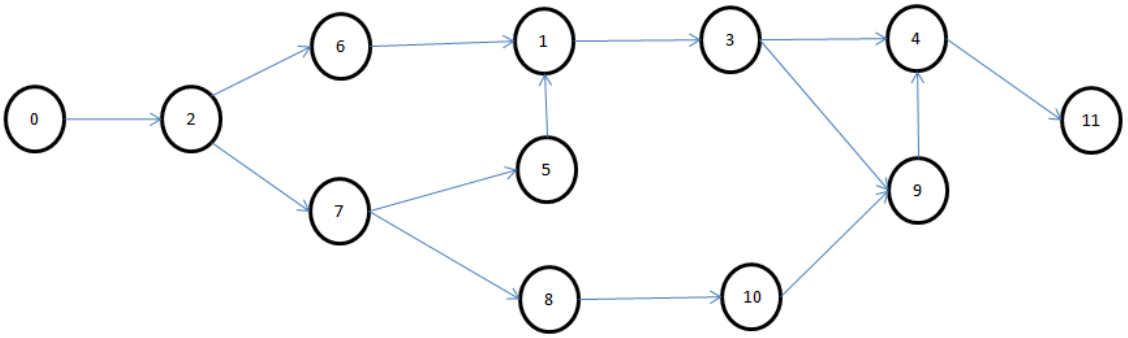

Figura 2: Grafo relativo ao problema

De acordo com Koné *et al.* (2009), existem problemas de sequenciamento que podem ser razoavelmente bem resolvidos, mas o PSAPRRP pertence à classe dos problemas de otimização realmente difíceis. Thomas e Salhi (1998) afirmam que a solução ótima do PSAP é difícil de determinar, especialmente para problemas de grande porte e com restrições de recursos e de precedência. Devido à complexidade do PSAPRRP, diversas formulações e métodos de resolução podem ser encontrados na literatura. Nas seções seguintes são apresentados três modelos, descritos por Koné *et al*. (2009), e alguns métodos de resolução para o PSAPRRP.

3.3. Modelos Matemáticos para o PSAPRRP

Em Koné *et al.* (2009) são apresentados diversos modelos matemáticos de programação linear inteira e inteira mista para o PSAPRRP, dentre os quais destacam-se os formulados com variáveis de decisão inteiras indexadas no tempo, propostos por Pritsker *et al.* (1969) e Christofides *et al.* (1987), e o formulado com variáveis de decisão contínuas no tempo proposto por Artigues *et al.* (2003). Tais modelos são descritos nas seções a seguir.

3.3.1. Modelos para o PSAPRRP com Variáveis Indexadas no Tempo

Os modelos de programação inteira, descritos por Koné *et al.* (2009) e propostos por Pritsker *et al.* (1969) e Christofides *et al.* (1987), apresentam variáveis de decisão inteiras e indexadas no tempo. A formulação pela indexação no tempo consiste na discretização do horizonte de planejamento H em períodos unitários t, onde cada período começa em t-1 e termina em t. Esse tipo de formulação apresenta uma importante desvantagem, a dimensão do modelo gerado. O número de variáveis binárias cresce proporcionalmente com T.

O modelo proposto por Pritsker *et al.* (1969) pode ser assim formalizado:

- **Dados do problema**
- A: conjunto de atividades, $A = \{1, ..., n\}$;
- R: conjunto de recursos, $R = \{1, ..., m\}$;
- E: conjunto de atividades sucessoras, $(i, j) \in E$ se a atividade j for sucessora de i;
- H: horizonte de sequenciamento, $H = \{1, ..., T\}$ onde T é a data máxima (limite superior) proposta para a finalização do projeto;
- ES_i: data mais cedo em que a atividade i pode ser iniciada;

- LS_i: data mais tarde em que a atividade *i* pode ser iniciada.

Na formulação proposta por Pritsker *et al.* (1969) são utilizadas, também, duas atividades "fictícias" (0 e *n*+1) para representarem, respectivamente, o início e o fim do projeto. Essas atividades possuem tempo de execução igual a zero e suas execuções não demandam por recursos.

- **Variável de decisão**

$$x_{it} = \begin{cases} 1, \textit{ se a atividade } i \textit{ for iniciada no período } t \\ 0, \textit{ caso contrário} \end{cases}$$

O modelo de programação linear inteira proposto por Pritsker *et al.* (1969) é apresentado pelas equações (5) a (11):

- **Modelo**

$$Min \quad \sum_{t=ES_{n+1}}^{LS_{n+1}} t\, x_{n+1,t} \tag{5}$$

$$s.\,a \quad \sum_{t=ES_j}^{LS_j} t\, x_{jt} \geq \sum_{t=ES_i}^{LS_i} t\, x_{it} + p_i \qquad \forall (i,\ j) \in E \tag{6}$$

$$\sum_{i=1}^{n} b_{ik} \sum_{\tau=\max(ES_i,\, t-p_i+1)}^{\min(LS_i,\, t)} x_{i\tau} \leq B_k \qquad \forall t \in H,\ \forall k \in R \tag{7}$$

$$\sum_{t=ES_i}^{LS_i} x_{it} = 1 \qquad \forall i \in A \cup \{n+1\} \tag{8}$$

$$x_{00} = 1 \tag{9}$$

$$x_{it} = 0 \qquad \forall i \in A \cup \{n+1\},\ t \in H \setminus [ES_i,\ LS_i] \tag{10}$$

$$x_{it} \in \{0,\ 1\} \qquad \forall i \in A \cup \{n+1\},\ t \in [ES_i,\ LS_i] \tag{11}$$

No modelo descrito, a função objetivo (5) consiste em minimizar a data de finalização do projeto. As restrições (6) garantem que as relações de precedência entre as atividades sejam obedecidas. O conjunto de restrições (7) assegura que o nível de recursos não seja violado. As restrições (8) garantem que cada atividade seja executada somente uma vez. A restrição (9) assegura que a atividade "fictícia" 0 seja iniciada no instante de tempo 0, representando o início do projeto. Já as restrições (10) e (11) determinam o domínio das variáveis. Tal modelo envolve $\sum_{i=1}^{n+1} (LS_i - ES_i)$ variáveis binárias e $|E| + (T+1)m + n + 1$ restrições.

Koné *et al.* (2009) apresentam, também, uma segunda formulação, proposta por Christofides *et al.* (1987), muito similar à descrita acima. As duas formulações se

diferem basicamente pela forma como são tratadas as restrições de precedência entre as atividades. Na formulação proposta por Christofides *et al.* (1987), as restrições de precedência (6) são substituídas pelo seguinte conjunto de restrições:

$$\sum_{\tau=t}^{LS_i} x_{i\tau} + \sum_{\tau=ES_j}^{\min(LS_j,\, t-p_i-1)} x_{j\tau} \leq 1 \qquad \forall (i,j) \in E,\ \forall t \in [ES_i,\, LS_i] \qquad (12)$$

As duas formulações apresentam o mesmo número de variáveis, só que a formulação proposta por Christofides *et al.* (1987) apresenta $m\sum_{i=1}^{n+1}(LS_i - ES_i)$ restrições a mais.

3.3.2. Modelo para o PSAPRRP com Variáveis Contínuas no Tempo

O modelo de programação inteira mista, com variáveis de decisão contínuas no tempo, proposto por Artigues *et al.* (2003), apresenta três tipos de variáveis:

- S_i: variáveis contínuas que definem a data de início de cada atividade *i*;
- $x_{ij} = \begin{cases} 1, \textit{ se a atividade } i \textit{ for executada antes da atividade } j \\ 0, \textit{ caso contrário} \end{cases}$
- f_{ijk}: variáveis contínuas que definem a quantidade de recurso *k* transferida da atividade *i* (no fim da sua execução) para a atividade *j* (no inicio da sua execução). Ou seja, são variáveis de fluxo.

Os conjuntos *A*, *R* e *E* são os mesmos definidos na Seção 3.3.1.

Os autores definem, também, que $\tilde{b}_{ik} = b_{ik}$, para todo $i \in A$, e $\tilde{b}_{0k} = \tilde{b}_{n+1k} = B_k$, representando que a atividade 0 funciona uma fonte de recursos e a atividade $n + 1$ um sumidouro.

O modelo de programação linear inteira mista proposto por Artigues *et al.* (2003) pode, então, ser assim formulado:

$$Min \quad S_{n+1} \qquad (13)$$

$$s.a \quad x_{ij} + x_{ji} \leq 1 \qquad \forall (i,\, j) \in (A \cup \{0,\, n+1\})^2,\ i < j \qquad (14)$$

$$x_{ik} \geq x_{ij} + x_{jk} - 1 \qquad \forall (i,\, j,\, k) \in (A \cup \{0,\, n+1\})^3 \qquad (15)$$

$$S_j - S_i \geq -M_{ij} + (p_i + M_{ij}) x_{ij} \qquad \forall (i,\, j) \in (A \cup \{0,\, n+1\})^2 \qquad (16)$$

$$f_{ijk} \leq \min(\tilde{b}_{ik},\, \tilde{b}_{jk}) x_{ij} \qquad \forall (i,\, j) \in (A \cup \{0\} \times A \cup \{n+1\}),\ \forall k \in R \qquad (17)$$

$$\sum_{j \in A \cup \{0,\, n+1\}} f_{ijk} = \tilde{b}_{ik} \qquad \forall i \in A \cup \{0,\, n+1\},\ \forall k \in R \qquad (18)$$

$$\sum_{i \in A \cup \{0,\, n+1\}} f_{ijk} = \tilde{b}_{jik} \qquad \forall j \in A \cup \{0,\, n+1\},\ \forall k \in R \qquad (19)$$

$$f_{n+1,0,k} = B_k \qquad \forall k \in R \qquad (20)$$

$$x_{ij} = 1 \qquad \forall (i,\ j) \in E \tag{21}$$

$$x_{ji} = 0 \qquad \forall (i,\ j) \notin E \tag{22}$$

$$f_{ijk} \geq 0 \qquad \forall (i,\ j) \in (A \cup \{0,\ n+1\})^2,\ \forall k \in R \tag{23}$$

$$S_0 = 0 \tag{24}$$

$$ES_i \leq S_i \leq LS_i \qquad \forall i \in A \cup \{n+1\} \tag{25}$$

$$x_{ij} \in \{0,\ 1\} \qquad \forall (i,\ j) \in (A \cup \{0,\ n+1\})^2 \tag{26}$$

Nesta formulação, M_{ij} é uma constante grande o suficiente para ser um limite superior válido para $S_i - S_j$, isto é, $M_{ij} \geq ES_i - LS_j$.

No modelo proposto por Artigues *et al.* (2003), a função-objetivo (13) consiste em minimizar a duração do projeto. As restrições (14) definem que para duas atividades distintas, ou *i* precede *j*, ou *j* precede *i*, ou *i* e *j* são executadas juntas. As restrições (15) expressam a transitividade das relações de precedência. O conjunto de restrições (16), chamadas restrições disjuntivas, vinculam as datas de início de execução das atividades *i* e *j* com a respectiva variável x_{ij}. As restrições (17) relacionam as variáveis de fluxo (f_{ijk}) e as variáveis x_{ij}. Se *i* precede *j*, o fluxo máximo de *i* para *j* é dado por $min(b_{ik}, b_{jk})$, e se *i* não precede *j*, o fluxo deve ser zero. As restrições (18) a (20) são as restrições de conservação de fluxo dos recursos. Os conjuntos de restrições (21) e (22) determinam as relações de precedência pré-existentes. A restrição (24) garante que o projeto seja iniciado na data zero. As restrições (23), (25) e (26) determinam o domínio das variáveis.

O modelo de Artigues *et al.* (2003) envolve $2|\overline{E}|$ variáveis binárias, onde $\overline{E}$ é o complemento do conjunto E ($\overline{E} = \{(i, j) | (i, j) \notin E\}$), e $m(n + 2)^2 + n + 1$ variáveis contínuas.

Para exemplificar a resolução do PSAPRRP através dos três modelos descritos, é utilizado o problema proposto em Koné *et al.* (2009), apresentado anteriormente. Os modelos foram implementados computacionalmente no *software* IBM ILOG CPLEX 12.1 e executados em um computador *Turion II Dual-Core* 2.20GHz e 4GB de RAM, sob sistema operacional *Windows* 7 *Home Premium* 64 Bits.

No problema proposto em Koné *et al.* (2009), para a execução das atividades do projeto são necessários dois tipos de recursos, cujas disponibilidades são, respectivamente, 5 e 3 unidades. O tempo de execução, em u. t., a demanda pelos recursos e as atividades sucessoras de cada atividade são descritas na Tabela 3.

Tabela 3: Dados do problema completo

Atividades	1	2	3	4	5	6	7	8	9	10
p_i	7	3	5	5	6	4	5	4	3	7
b_{i1}	0	2	3	3	2	1	1	1	1	3
b_{i2}	2	1	3	2	1	0	3	1	1	1
Sucessoras	3	6, 7	4, 9	11	1	1	5, 8	10	4	9

Fonte: Koné *et al*. (2009).

No problema apresentado na Tabela 3, a atividade 1 possui tempo de execução igual a 7 u. t., demanda 0 e 2 unidades pelos recursos 1 e 2, respectivamente, e somente a atividade 3 é dependente da sua execução, ou seja, é sua sucessora. Já a atividade 2 possui tempo de execução igual a 3 u. t., demanda 2 e 1 unidades, respectivamente, pelos recursos 1 e 2 e tem como atividades sucessoras a 6 e a 7. E assim sucessivamente. A execução de todas as atividades pode ser iniciada em qualquer instante de tempo *t*, dentro do horizonte de planejamento *H*, desde que suas predecessoras tenham sido finalizadas e $t \leq T - p_i$. A data final proposta para o horizonte de planejamento, ou seja, a data mais tarde no qual o projeto pode ser finalizado, foi T = 60 u. t.. A solução obtida para o problema descrito, através da resolução dos três modelos, pode ser vista na Tabela 4.

Tabela 4: Solução do problema

Atividaes	1	2	3	4	5	6	7	8	9	10
Data de Início	15	1	22	30	9	10	4	10	27	14

Na Tabela 4 são apresentadas as datas programadas para o início da execução das atividades correspondentes à solução obtida para o problema, ou seja, S = {15, 1, 22, 30, 9, 10, 4, 10, 27, 14}. Na solução, as 10 atividades têm o início de sua execução nos períodos de tempo 15, 1, 22, 30, 9, 10, 4, 10, 27 e 14, respectivamente, obtendo uma duração para o projeto igual a 35 u. t.. O Gráfico de *Gantt* representando a solução do problema é apresentado na Figura 3.

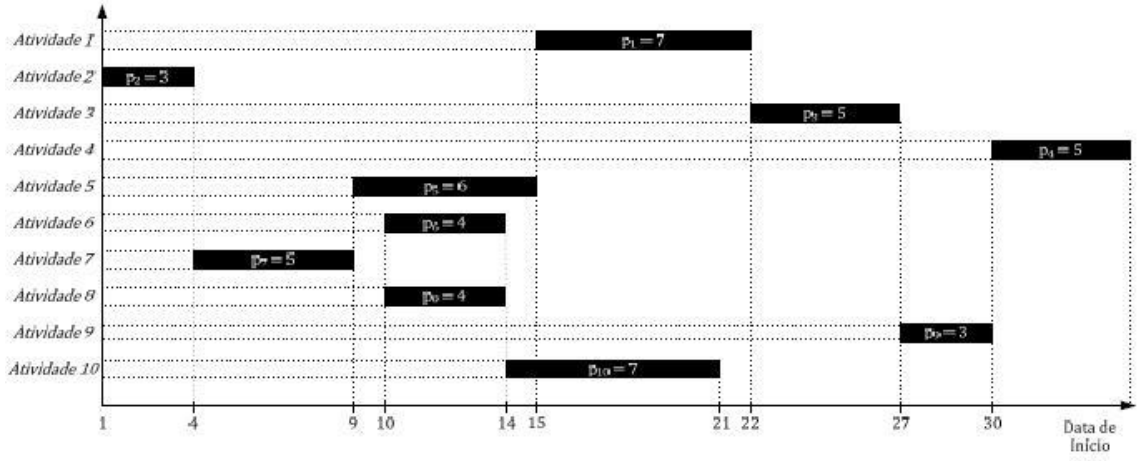

Figura 3: Gráfico de *Gantt* para o sequenciamento ótimo do problema

No Gráfico de *Gantt*, apresentado na Figura 3, é representado o sequenciamento no qual as atividades devem ser executadas, obtido na resolução do problema. A primeira atividade a ser executada é a 2, seguida pelas atividades 7, 5, 6 e 8, 10, 1, 3, 9 e 4. O projeto tem seu término programado para o período de tempo 35, data de finalização da execução da atividade 4.

Os tempos computacionais, em segundos, gastos por cada modelo na obtenção da solução do problema, bem como o número de variáveis e de restrições gerados são descritos na Tabela 5.

Tabela 5: Tempos Computacionais, Nº de Variáveis e Nº de Restrições

Modelo	**Tempo Computacional**	**Número de Variáveis**	**Número de Restrições**
Pritsker *et al.* (1969)	0,56	401	113
Christofides *et al.* (1987)	0,13	401	887
Artigues *et al.* (2003)	0,59	444	2319

Como pode ser visto na Tabela 5, para o problema utilizado, os três modelos apresentaram tempo computacional e número de variáveis muito próximos. Porém, apresentaram valores discrepantes para o número de restrições. O modelo com variáveis contínuas proposto por Artigues *et al.* (2003) apresentou um número bem maior de restrições, aproximadamente 20 vezes mais restrições que o modelo proposto por Pritsker *et al.* (1969) e 2,6 vezes mais que o modelo proposto por Christofides *et al.* (1987). O modelo de Artigues *et al.* (2003) também foi o que apresentou o número maior de variáveis.

Para problemas pequenos como o apresentado, os modelos exigiram baixo esforço computacional, mas devido ao PSAPRRP ser de difícil resolução no caso geral, como descrevem Koné *et al.* (2009) e Thomas e Salhi (1998), é interessante ressaltar que, para problemas com dimensões maiores como nos casos reais, as formulações apresentadas podem ser menos eficientes computacionalmente. Essa questão pode ser tratada e bem resolvida aplicando-se outras técnicas de otimização, diminuindo, dessa maneira, o esforço computacional.

Na seção seguinte é apresentada a descrição de algumas técnicas de otimização, encontradas na literatura, aplicadas na resolução do PSAPRRP.

3.4. Métodos Heurísticos Aplicados ao PSAPRRP

O PSAPRRP pertence à classe dos problemas combinatórios, a qual é caracterizada pelo crescimento fatorial do tempo computacional necessário para a consideração de todas as suas possíveis soluções em relação à dimensão do problema (Davis, 1973). Portanto, os métodos de resolução para o mesmo devem ser extraídos da otimização combinatória (OC). A OC trata de problemas que são caracterizados por um número finito de soluções viáveis possíveis. Embora, a princípio, a solução ótima de um problema com um número finito de soluções viáveis possa ser obtida por simples enumeração, na prática essa tarefa é, na maioria das vezes, impossível, ainda mais para problemas reais onde o número de soluções possíveis pode ser muito alto.

Os diversos métodos de resolução para problemas como o PSAPRRP, encontrados na literatura, podem ser divididos em dois grandes grupos: os métodos exatos, que buscam a solução ótima para os problemas; e os métodos aproximados ou heurísticos, que visam obter boas soluções.

Devido ao fato de o PSAPRRP pertencer à classe de problemas *NP*-difíceis, dificilmente serão encontrados métodos exatos eficientes, ou seja, métodos que gerem soluções ótimas em um tempo polinomial, para a resolução de problemas reais e de grandes dimensões. Hamm *et al.* (2009) afirmam que é computacionalmente impraticável a utilização de métodos exatos para a obtenção do sequenciamento ótimo para o PSAPRRP. Em seus experimentos, Alvarez-Valdez e Tamarit (1989) conseguiram resolver na otimalidade problemas com até 50 atividades, no máximo. Segundo Thomas e Salhi (1998), as heurísticas apareceram como a melhor forma de obter soluções boas para problemas combinatórios com elevadas dimensões, como é o caso do PSAPRRP. Portanto, o foco deste trabalho são os métodos heurísticos.

Um método heurístico é definido como uma técnica que procura boas soluções (próximas da ótima) a um custo computacional razoável, sem, no entanto, estar capacitada a garantir a otimalidade, bem como garantir quão próximo uma determinada solução está da solução ótima.

Foram desenvolvidos nas últimas décadas diversos métodos heurísticos para a resolução do PSAPRRP, obtendo-se soluções boas ou próximas da ótima em um tempo computacional razoável. De acordo com Kolisch (1996), os métodos heurísticos para o PSAPRRP podem ser divididos, basicamente, em três tipos: Programação baseada em

regras de prioridade; técnicas de busca local (metaheurísticas); e Algoritmos Evolucionários. Os três tipos de métodos heurísticos são apresentados nas seções a seguir.

3.4.1. Métodos Heurísticos de Programação Baseados em Regras de Prioridade

Os métodos heurísticos baseados em regras de prioridade ainda estão entre as mais importantes técnicas utilizadas para a resolução do PSAPRRP devido ao fato de serem intuitivos e de fácil implementação, além de exigirem baixo esforço computacional. A maioria dos métodos heurísticos baseados em regras de prioridade aplicados ao PSAPRRP pode ser descrita através do procedimento apresentado na Figura 4.

Método Heurístico Baseado em Regras de Prioridade

$s \leftarrow \phi$;

Enquanto (*houver alguma atividade não sequenciada*) **faça**
  Avance para o período mais próximo onde há algum recurso não alocado;
  Estabeleça a lista de atividades sequenciáveis;
    Se (*a lista estiver vazia*) **então**
      Volte para o passo anterior;
    Fim_se;
  Escolha a atividade sequenciável *t* que possui a maior prioridade;
  $s \leftarrow s \cup \{t\}$;

Fim_enquanto;
Retorna *s*;

Figura 4: Pseudocódigo de um Método Heurístico de Programação Baseado em Regras de Prioridade.

Para o procedimento descrito na Figura 4, uma atividade é considerada sequenciável se duas condições forem satisfeitas: todas as suas atividades predecessoras já estiverem sequenciadas e possuir pelo menos um modo de execução. Um modo de execução de uma atividade só é considerado possível se a quantidade de cada recurso demandado por ela for menor ou igual à disponibilidade dos recursos no período de tempo considerado.

Segundo Kolisch (1996), geralmente um método heurístico baseado em regras de prioridade é composto por uma regra de prioridade e um método de geração de sequenciamentos para a determinação de sequenciamentos viáveis. Algumas regras de prioridade e dois métodos de geração de sequenciamentos são descritos a seguir.

A - Regras de Prioridade

Em seu trabalho, Boctor (1993) define as seguintes regras como as principais regras de priorização de atividades utilizadas em heurísticas para o PSAPRRP:

(1) Menor Folga Total (FT_i);
(2) Menor data mais tarde possível para início de execução (LS_i);
(3) Maior número de atividades sucessoras;
(4) Maior tempo de execução (p_i);
(5) Maior custo de execução.

Define-se Folga Total de uma atividade *i* (FT_i) como sendo o espaço de tempo, além da duração prevista para a atividade (p_i), medido entre a data de término da execução da atividade ($S_i + p_i$) e sua respectiva data mais tarde de término ($LS_i + p_i$), supondo que a execução da atividade seja iniciada na sua data mais cedo possível de

início ($S_i = ES_i$).

$$FT_i = (LS_i + p_i) - (ES_i + p_i) \text{ ou } FT_i = LS_i - ES_i$$

Se $FT_i = 0$, isto quer dizer que a atividade *i*, iniciando a execução em sua data mais cedo possível (ES_i) e sendo finalizada na data mais tarde possível ($LS_i + p_i$), ocupa todo o tempo disponível do projeto.

Os dois principais tipos de métodos de geração de sequenciamento encontrados na literatura são o método serial (*Serial Method*) e o método paralelo (*Parallel Method*), apresentados a seguir.

B - Método Serial de Geração de Sequenciamentos

O método serial, proposto por Kelley (1963), consiste em *N* etapas onde, em cada uma, uma atividade é selecionada para ser sequenciada. O pseudocódigo para o método serial de geração de sequenciamentos (MSGS) é descrito na Figura 5.

Método Serial de Geração de Sequenciamentos

$n \leftarrow 1$;
$S_n \leftarrow \phi$;
Enquanto ($|S_n| < N$) **faça**
  Determine D_n e K'_{kt};
  $j^* \leftarrow min_{j \in Dn} \{j \mid v(j) = inf_{i \in Dn} \{v(i)\}\}$;
  $FC_{j*} \leftarrow min \{F_i + p_i \mid i \in P_{j*}\}$;
  $F_{j*} \leftarrow min \{t \mid FC_{j*} \leq t \leq LS_{j*}\}$, com $\{b_{j*k} \leq K'_{kt}\}$;
  $S_{n+1} \leftarrow S_n \cup \{j*\}$;
  $n \leftarrow n + 1$;
Fim_enquanto;
Retorna S_n;

Figura 5: Pseudocódigo do MSGS

No pseudocódigo descrito na Figura 5, tem-se:

p_j = duração da execução da atividade *j*;
D_n = conjunto de decisão;
$D_n = \{j \mid j \notin S_n, P_j \subseteq S_n\}$;
F_j = data de início da execução da atividade *j*;
FC_j = data mais cedo possível para o início da execução da atividade *j*;
LS_j = data mais tarde possível para o início da execução da atividade *j*;
S_n = conjunto de atividades sequenciadas;
P_j = conjunto de atividades antecessoras da atividade *j*;
$v(j)$ = valor da prioridade da atividade *j*; $\{j \in D_n\}$;
I_j = data de finalização da execução da atividade *j*;
$I_j = F_j + p_j$;
K'_{kt} = quantidade de recurso *k* disponível na data *t*;

$$K'_{kt} = \sum_{j \in A_t} B_k - b_{jk}$$

B_k = disponibilidade de recurso *k*;
b_{jk} = quantidade de recurso *k* utilizado pela atividade *j* por unidade de tempo.

No MSGS dois conjuntos de atividades disjuntos estão associados a cada etapa: no conjunto S_n encontram-se as atividades já sequenciadas e no conjunto D_n as não sequenciadas, cujas antecessoras já encontram-se em S_n. A cada etapa, uma atividade *j*

$\in D_n$ é selecionada, baseada em alguma regra de prioridade, e sequenciada para ter seu início na data mais cedo que satisfaça as restrições de precedência e de recursos. No caso de duas atividades possuírem a mesma prioridade, será selecionada a de menor número. Depois os conjuntos S_n e D_n são atualizados e este processo se repete até que todas as atividades estejam sequenciadas, ou seja, quando $n = N$.

C - Método Paralelo de Geração de Sequenciamentos

O método paralelo de geração de sequenciamentos (MPGS) consiste em sequenciar um conjunto de atividades em cada uma das N etapas do método. A característica que diferencia este método é que a cada etapa n é associada uma data de sequência t_n, onde $t_m \leq t_n$ para $m < n$. Por causa desta data de sequência, o conjunto de atividades sequenciadas é dividido em dois subconjuntos: o conjunto completo C_n, que contém as atividades já sequenciadas cujas execuções terminam até a data de sequência e, o conjunto ativo A_n, que contém as atividades já sequenciadas, mas suas execuções não terminaram até a data de sequência t_n.

O conjunto de decisão D_n, diferentemente do MSGS, contém também todas as atividades não sequenciadas que estão disponíveis para o sequenciamento, em termos das restrições de recursos e de precedência. O sequenciamento parcial de cada etapa é feito pelas atividades que estão nos conjuntos C_n e A_n. O pseudocódigo para o MPGS é apresentado na Figura 6.

Método Paralelo de Geração de Sequenciamentos

$n \leftarrow 1$;
$t_n \leftarrow 0$;
$D_n \leftarrow \{1\}$;
$A_n \leftarrow C_n \leftarrow \phi$;
$K'_k \leftarrow B_k \quad \forall k \in R$;
Vá para o **Passo (2)**;
Enquanto ($| A_n \cup C_n | < N$) **faça**
Passo (1)
  $t_n \leftarrow min \{F_n \mid j \in S_{n-1}\}$;
  $A_n \leftarrow A_{n-1} \mid \{j \mid j \in A_{n-1}, F_n = t_n\}$;
  $C_n \leftarrow C_{n-1} \cup \{j \mid j \in A_{n-1}, F_n = t_n\}$;
  Determine $K'_k \ \forall k \in R$ e D_n;
Passo (2)
  $j^* \leftarrow min_{j \in Dn} \{j \mid v(j) = inf_{i \in Dn} \{v(i)\}\}$;
  $F_{j^*} \leftarrow t_i$;
  $A_n \leftarrow A_i \cup \{j^*\}$;
  Determine $K'_k \ \forall k \in R$ e D_n;
  Se ($D_n \neq \phi$) **então**
    Vá para o **Passo (2)**;
  senão $n \leftarrow n++$;
  Fim_se;
Fim_enquanto;
Retorna A_n;

Figura 6: Pseudocódigo do MPGS

No pseudocódigo apresentado na Figura 6, tem-se:

A_n = conjunto de atividade ativas na data t_n;
C_n = conjunto de atividades completadas até a data t_n;
p_j = duração da atividade j;
D_n = conjunto de decisão;

$$D_n = \{j \mid j \notin \{A_n \cup C_n\}\ P_j \subseteq C_n, K'_{jk} \Leftarrow K_n\ \forall\, k \in R\}$$

F_j = data programada para o inicio da execução da atividade j;
P_i = conjunto de atividades antecessoras da atividade i;
$v(j)$ = valor da prioridade da atividade j;
K'_{kt} = quantidade de recurso k disponível na data t;

$$K'_{kt} = \sum_{j \in A_t} B_k - b_{jk}$$

B_k = Disponibilidade de recurso k;
b_{jk} = quantidade de recurso k utilizado pela atividade j por unidade de tempo.

No MPGS, cada etapa é realizada em dois passos: (1) a data de sequência nova é determinada e as atividades com data de finalização de sua execução igual à nova data de sequência são removidas de A_n e adicionadas à C_n; (2) uma atividade de D_n é selecionada de acordo com a regra de prioridade definida e sequenciada para iniciar na data de sequência corrente. Depois, esta atividade é removida de D_n e adicionada a C_n. O passo (2) é repetido até que o conjunto D_n esteja vazio, ou seja, quando todas as atividades estiverem sequenciadas. O MPGS para quando todas as atividades pertencerem a A_n.

Autores como Alvarez-Valdes e Tamarit (1989) e Kolisch (1996) apresentaram estudos relacionados aos dois métodos de geração de sequenciamento descritos. Segundo os autores, estes métodos têm sido muito utilizados na determinação de soluções iniciais para técnicas de busca local (metaheurísticas) destinadas à resolução do PSAPRRP.

3.4.2. Técnicas de Busca Local

As técnicas de busca local para OC são métodos que realizam uma série de iterações, melhorando a solução de um determinado problema através de movimentos dentro de uma estrutura de vizinhança. Este tipo de técnica tem sido muito utilizado na determinação de boas soluções, dentro de um tempo computacional razoável, para problemas combinatórios.

Denomina-se movimento, uma modificação que transforma uma solução s em outra, s', que esteja em sua vizinhança. O conjunto de soluções que pode ser obtido desta forma, a partir de uma dada solução s, é chamado de vizinhança de s e denotado por $N(s)$. O conceito de vizinhança é baseado na possibilidade de utilizar um método que perturbe uma solução encontrada, de modo a obter uma ou mais diferentes soluções, eventualmente de melhora, para um problema.

As principais técnicas de busca local encontradas na literatura para o PSAPRRP são a Busca Tabu, o *Simulated Annealing*, os Algoritmos Genéticos e a Colônia de Formigas. Mas outras técnicas também podem ser utilizadas como, por exemplo, o GRASP, o VNS e o ILS.

A - Busca Tabu

A Busca Tabu (BT), proposta por Glover (1986) e baseada na inteligência artificial, é um procedimento adaptativo dotado de uma estrutura de memória que aceita movimentos de piora para escapar de ótimos locais. Esta estratégia pode, no entanto, fazer com que o algoritmo cicle, isto é, retorne a uma solução já gerada anteriormente. Para evitar que isso ocorra, é utilizada uma lista tabu *LT*, a qual armazena movimentos proibidos dando-os o *status* de tabu. A *LT* clássica armazena os movimentos reversos aos últimos |*LT*| movimentos realizados (onde |*LT*| é um parâmetro da BT) e funciona como uma fila de tamanho fixo, isto é, quando um novo movimento é armazenado em *LT*, o mais antigo sai. Portanto, na exploração de $N(s)$, ficam excluídos da busca as soluções s' que são obtidas por movimentos *m* armazenados em *LT*. A *LT* se por um lado reduz o risco de ciclagem, por outro também pode proibir movimentos para soluções que ainda não foram geradas. Assim, utiliza-se, também, uma *função de aspiração*, que é um mecanismo que retira, sob certas circunstâncias, o *status* tabu de um movimento. Para cada possível valor *v* da função objetivo, está associado um nível de *aspiração* $A(v)$. Então, uma solução $s' \in N(s)$ pode ser gerada se $f(s') < A(f(s))$, mesmo que este movimento *m* seja tabu. O pseudocódigo do caso geral do procedimento BT, para problemas de minimização, é descrito na Figura 7.

Procedimento BT
Entrada: $f(.)$, $N(.)$, $A(.)$, $|V|$, f_{min}, $|LT|$, BT_{max}, s
Saída: s
$s^* \leftarrow s$;
$Iter \leftarrow 0$;
$MelhorIter \leftarrow 0$;
$LT \leftarrow \phi$;
Inicialize a *função de aspiração A*;
Enquanto ($f(s) > f_{min}$ e $Iter - MelhorIter < BT_{max}$) **faça**
  $Iter \leftarrow Iter + 1$;
  Seja $s' \leftarrow s \oplus m$ o melhor elemento de $V \subseteq N(s)$ tal que o movimento *m* não seja tabu ($m \notin LT$) ou s' atenda a condição de aspiração ($f(s') < A(f(s))$);
  Atualize a lista tabu *LT*;
  $s \leftarrow s'$;
  Se ($f(s) < f(s^*)$) **então**
    $s^* \leftarrow s$;
    $MelhorIter \leftarrow Iter$;
  Fim_se;
  Atualize a *função de aspiração A*;
Fim_Enquanto;
$s \leftarrow s^*$;
Retorna s;

Figura 7: Pseudocódigo da Metaheurística Busca Tabu

A BT começa com a geração de uma solução inicial *s* e explora, a cada iteração, um subconjunto $V \subset N(s)$. A solução $s' \in V$ com melhor valor, segundo a função $f(.)$ torna-se a nova solução corrente mesmo que s' seja pior do que *s*, ou seja, $f(s') > f(s)$ para um problema de minimização. Duas regras são geralmente utilizadas para finalizar o procedimento. Na primeira, o método para quando é atingido um certo número de iterações (BT_{max}) sem melhora no valor da melhor solução. Na segunda, o método para quando o valor da melhor solução chega a um limite inferior conhecido (ou próximo

dele).

Thomas e Salhi (1998) propuseram uma metodologia baseada na metaheurística BT para a resolução do PSAPRRP, tendo como objetivo a minimização do *makespan* do projeto. Para o método, os autores definem três tipos de movimentos possíveis: troca de atividades e inserção de atividade tipo 1 e tipo 2.

Com base em uma solução qualquer, o movimento troca de atividades requer a escolha aleatória de duas atividades para serem trocadas de posição no sequenciamento. A escolha das atividades é fundamentada em três condições:

Dada uma solução anteriormente obtida, as atividades i e j são viáveis para serem trocadas de posição se:

(a) $S_i \neq S_j$;
(b) $p_i, p_j > 0$;
(c) $j \notin C_i$.

onde C_i é o conjunto de atividades pelo qual, através de algum caminho de atividades, a atividade i pode chegar à atividade j. Se as três condições são satisfeitas, então:

$$S_i^{novo} = S_j \text{ e } S_j^{novo} = S_i$$

Em (a), para a escolha das atividades a serem trocadas de posição no sequenciamento, considera-se somente atividades com datas de início de execução diferentes. Trocar atividades com mesma data de inicio de execução não afetará a duração do projeto. A condição (b) garante que as atividades artificiais 0 (início do projeto) e n+1 (fim do projeto) não sejam consideradas para troca. A condição (c) é utilizada para manter a viabilidade em termos das relações de precedência.

O movimento inserção de atividade requer que uma atividade i seja deslocada para outra posição dentro do sequenciamento. Duas possibilidades são consideradas para este movimento. No movimento inserção de atividade tipo 1, a data de início de execução da atividade i é inserida na data de início de execução de uma outra atividade j. No movimento tipo 2, a data de início de execução da atividade i é inserida na data de finalização de execução de uma outra atividade j.

Uma atividade i é viável para a inserção do tipo 1 se as condições (a), (b) e (c) são atendidas e, então:

$$S_i^{novo} = S_j$$

Similarmente, uma atividade i é viável para a inserção do tipo 2 se as condições (b) e (c) forem atendidas e, então:

$$S_i^{novo} = S_j + p_j$$

Utilizando o movimento troca de atividades, a vizinhança de uma solução é composta por $n(n-1)/2$ soluções e, utilizando os movimentos inserção de atividade do tipo 1 e 2, $(n-1)^2$ soluções.

Thomas e Salhi (1998) denotaram os tipos de movimentos por k (k = 1, 2 e 3), onde k = 1 representa o movimento troca de atividades, k = 2 representa o movimento inserção de atividade do tipo 1 e k = 3 representa o movimento inserção de atividade do tipo 2.

Os autores definem inicialmente o *status* tabu $tab(i, j, k) = 0$ para todos os possíveis pares de atividades i e j e k = 1, 2 e 3. Um movimento do tipo k, envolvendo duas atividades i e j, é considerado tabu e, portanto, proibido de ser realizado, se $tab(i, j, k) > iter$, onde $iter$ é o contador de iterações. O *status* tabu é atualizado através da seguinte fórmula:

$$tab(i, j, k) = iter + |LT|$$

em que $|LT|$ é o tamanho pré-definido para a *LT*.

Para comparar os resultados gerados pela BT, Thomas e Salhi (1998) utilizaram soluções ótimas e limites inferiores obtidos por Heurísticas Lagrangeanas, comprovando, assim, a eficiência do método.

Klein (2000) também desenvolveu uma técnica BT, denominada Busca Tabu Reativa, para o PSAPRRP com disponibilidade de recursos variantes no tempo. O método proposto é baseado no MSGS e a estrutura de vizinhança utilizada pelo autor consiste na troca de posição no sequenciamento de duas atividades, incluindo, também, a troca de seus sucessores e predecessores, garantindo, dessa forma, sequenciamentos viáveis.

B - Simulated Annealing

A metaheurística *Simulated Annealing* (*SA*), proposta por Kirkpatrick *et al.* (1983), é uma técnica de busca local probabilística, fundamentada em uma analogia com a termodinâmica ao simular o resfriamento de um conjunto de átomos aquecidos, operação conhecida como recozimento (*Annealing*). O *SA* começa sua busca a partir de uma solução inicial qualquer e seu procedimento principal, descrito na Figura 8, consiste em um *loop* que gera aleatoriamente, a cada uma das SA_{max} iterações, um único vizinho *s´* da solução corrente *s*.

```
Procedimento SA
Entrada: f(.), N(.), α , SA_max, T_0, s
Saída: s
s* ← s;
IterT ← 0;
T ← T_0;
Enquanto (T > 0) faça
   Enquanto (IterT < SA_max) faça
      IterT ← IterT + 1;
      Gere um vizinho qualquer s´ ∈ N(s);
      Δ = f(s´) – f(s);
      Se ( Δ < 0) então
         s ← s´;
         Se (f(s´) < f(s*)) então
            s* ← s´;
         senão Determine x ∈ [0, 1];
            Se ( x < e^(−Δ/T) ) então
               s ← s´;
            Fim_se;
         Fim_se;
      Fim_se
   Fim_enquanto;
      T ← α x T;
      IterT ← 0;
Fim_enquanto;
s ← s*;
Retorna s;
```

Figura 8: Pseudocódigo da Metaheurística *Simulated Annealing*

Considerando um problema de minimização, seja Δ a variação no valor da função objetivo ao mover-se de uma solução *s* para uma solução *s´*, isto é, $\Delta = f(s´) - f(s)$. O *SA* aceita o movimento e a solução *s´* passa a ser a solução corrente se $\Delta < 0$ ($f(s´) < f(s)$), ou seja, *s´* é melhor do que *s*. Caso $\Delta \geq 0$ a solução *s´* também pode ser aceita, mas neste caso, com uma probabilidade $e^{-\Delta/T}$, onde *T* é um parâmetro do *SA*, denominado temperatura e que regula a probabilidade de aceitação ou não de uma solução de piora. A temperatura assume inicialmente um valor elevado T_0 e, após um número fixo de iterações, é gradativamente reduzida por uma razão de resfriamento (α), tal que $T_k \leftarrow \alpha \times T_{k-1}$, sendo 0 < α < 1. À medida que *T* aproxima-se de zero, diminui-se a probabilidade de aceitação de soluções de piora ($T \to 0 \Rightarrow e^{-\Delta/T} \to 0$). O procedimento para quando a temperatura aproxima-se de zero e nenhuma solução de piora é mais aceita, isto é, quando o sistema torna-se estável.

Segundo Kirkpatrick *et al.* (1983), a velocidade de convergência para boas soluções do *SA* depende da definição de T_0, de α e da taxa de resfriamento ($\Delta t = t - (\alpha * t)$). Os valores ideais para estes parâmetros não podem ser determinados antecipadamente, eles dependem do problema específico que se esta querendo resolver e devem ser ajustados empiricamente.

König e Beibert (2009) utilizaram um algoritmo *SA* integrado ao conceito de simulação baseada em restrições para a resolução do PSAPRRP. Na formulação utilizada, os autores consideram como restrições a dependência tecnológica de construção (precedências), a capacidade da obra (equipamentos e mão de obra), a disponibilidade de matéria-prima e os aspectos espaciais (área de trabalho).

De acordo com as restrições de precedência, König e Beibert (2009) definiram níveis de execução para as atividades. O nível de execução de uma atividade depende de seus predecessores e é determinado pelo comprimento máximo dentre os possíveis caminhos para se chegar à mesma, partindo da atividade 0. Utilizando o problema com 10 atividades descrito na seção 3.2, os níveis estabelecidos para as atividades seriam os apresentados na Figura 9.

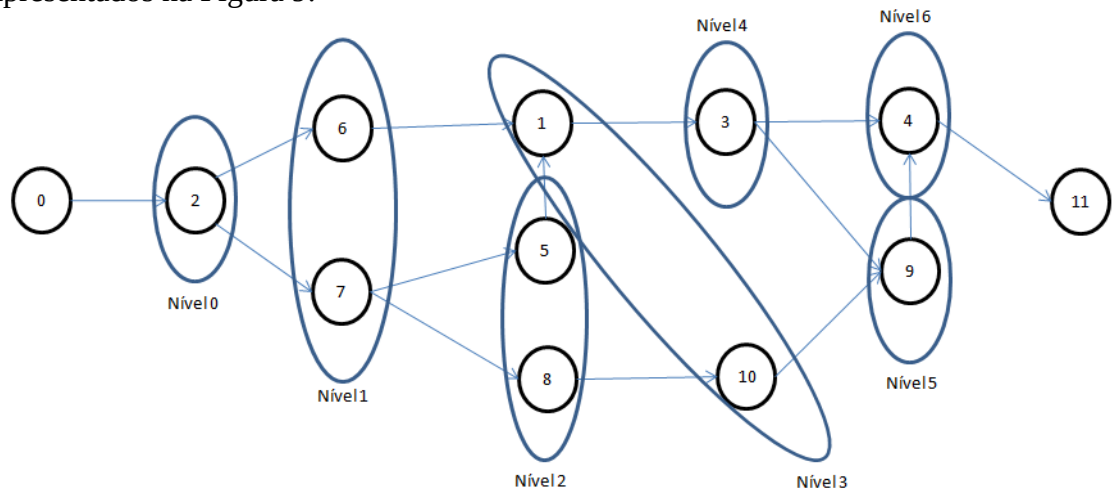

Figura 9: Grafo com níveis de atividades

Na determinação do valor do comprimento dos caminhos no grafo, não é levado em consideração o arco que sai do nó zero, pois ele é utilizado somente para representar o início do projeto. Os demais arcos possuem comprimento igual a um. Então, como pode ser visto na Figura 9, o nível zero é composto somente pela atividade 2, pois o comprimento do caminho máximo para chegar de zero até ela é 0. O nível 1 é composto pelas atividades 6 e 7, já que o comprimento do caminho máximo de zero até elas é 1. E

assim por diante. No caso em que os recursos necessários para a execução das atividades possuem disponibilidades ilimitadas, a solução ótima para o problema é obtida sequenciando simultaneamente todas as atividades pertencentes ao mesmo nível de execução.

Segundo os autores, a definição de uma estrutura de vizinhança apropriada também é muito importante na utilização do *SA*. Sendo assim, com base nos níveis de execução das atividades, os autores definiram uma única estrutura. O movimento utilizado por König e Beibert (2009) para a obtenção de uma nova solução consiste na troca aleatória na ordem de sequenciamento de duas atividades pertencentes ao mesmo nível de execução. Com base no problema descrito na Figura 9, a atividade 6 só poderia ser trocada de posição com a atividade 7, a 5 com a 8 e a 1 com a 10.

No *SA* proposto por König e Beibert (2009), a probabilidade que determina a aceitação ou não de uma nova solução é dada pela função $p(cc, cn, t)$, que depende do valor da solução corrente (cc), do valor da nova solução (cn) e da temperatura corrente (t). De acordo como definido por Kirkpatrick *et al.* (1983), os autores utilizaram a seguinte função de probabilidade:

$$p(cc,cn,t)=\begin{cases}1, \textit{ se } cn < cc \\ e^{((cc-cn)/t)}, \textit{ caso contrário}\end{cases}$$

A temperatura também é reduzida como proposto por Kirkpatrick *et al.* (1983), $t_k = \alpha \times t_{k-1}$.

C - Algoritmos Genéticos

Os Algoritmos Genéticos (AGs) são metaheurísticas que fundamentam-se em uma analogia com os processos naturais de evolução nos quais, dada uma população, os indivíduos com características genéticas melhores têm maiores chances de sobrevivência e de produzirem filhos cada vez mais aptos, enquanto indivíduos menos aptos tendem a desaparecer. Referencia-se Goldberg (1989) e Reeves (1993) para um maior detalhamento do método.

Nos AGs, cada cromossomo (indivíduo da população) está associado a uma solução do problema e cada gene está associado a uma componente da solução. Um mecanismo de reprodução, baseado em processos evolutivos, é aplicado sobre a população com o objetivo de explorar o processo de busca e encontrar melhores soluções para o problema. O pseudocódigo de um AG é descrito na Figura 10.

```
Procedimento AG
t ← 0;
Gere a população inicial P(t);
Avalie P(t);
Enquanto (Critério de Parada = Falso) faça
   t ← t++;
   Gere P(t) a partir de P(t - 1);
   Avalie P(t);
   Defina a população sobrevivente;
Fim_enquanto;
Retorna P(t);
```

Figura 10: Pseudocódigo da Metaheurística Algoritmos Genéticos

O AG inicia sua busca através de uma população gerada aleatoriamente, a qual é chamada de população no tempo 0 ($P(0)$). Como pode ser visto na Figura 10, o procedimento principal do AG é um *loop* que cria uma população no tempo t +1 a partir

de uma população no tempo *t*, onde cada população é composta por *n* cromossomos. Para a criação de uma nova população no tempo *t*+1, os indivíduos da população no tempo *t* passam por uma fase de reprodução, a qual consiste em selecionar indivíduos para operações de recombinação e/ou mutação.

Na operação de recombinação, os genes de dois cromossomos pais são combinados gerando cromossomos filhos (normalmente dois), de forma que, para cada cromossomo filho, haja um conjunto de genes de cada um dos cromossomos pais. A operação de mutação consiste em alterar aleatoriamente uma parte dos genes de cada cromossomo. Ambas as operações são realizadas com certa probabilidade.

Gerada a nova população no tempo *t*+1, define-se a população sobrevivente, isto é, os *n* cromossomos que integrarão a nova população. Para definir a população sobrevivente, cada solução é avaliada por uma função de aptidão, determinando, dessa forma, os melhores cromossomos.

O método termina, geralmente, quando um certo número de populações é gerado ou quando a melhor solução encontrada atinge um certo nível de aptidão ou ainda, quando não há melhora após um certo número de iterações.

Segundo Deiranlou e Jolai (2009), uma das primeiras tentativas de se aplicar o AG a um problema de sequenciamento foi feito por Davis (1985). A partir daí, vários outros autores utilizaram AGs na resolução de problemas deste tipo.

Deiranlou e Jolai (2009) propuseram um novo AG para a resolução do PSAPRRP, denominado DHGA (do inglês, *Development of Hartmann Genetic Algorithm*), onde o objetivo é minimizar o *makespan* do projeto. O AG proposto pelos autores inicia-se com a geração de uma população inicial. O AG determina, então, o valor da função de aptidão para os indivíduos da população inicial e, depois, divide aleatoriamente a população em pares de indivíduos. Para cada par de indivíduos resultante, aplica-se o operador de recombinação para produzir dois novos indivíduos. Subsequentemente, aplica-se o operador de mutação aos novos indivíduos gerados e, após calcular o valor da função de aptidão de cada individuo novo, adiciona-os à população corrente. Então, aplica-se um operador de seleção, que seleciona os indivíduos mais aptos (maiores valores da função de aptidão), reduzindo a população ao tamanho anterior. Finalmente, as probabilidades dos operadores de recombinação e de mutação são ajustadas.

Agarwal *et al.* (2011) desenvolveram um algoritmo hibrido combinando os AGs e as Redes Neurais (RN), denominado algoritmo Neurogenético, para a resolução do PSAPRRP. Para a geração de sequenciamentos viáveis, foram utilizados tanto o MSGS como o MPGS. No algoritmo proposto por Agarwal *et al.* (2011), o processo de busca baseia-se no AG para a busca global e nas RN para busca local. Para isso, os autores intercalam iterações de busca do AG com iterações de busca das RN. A ideia dos autores é a de que após algumas iterações do AG sejam geradas populações com boas soluções e que estas estejam distribuídas globalmente no espaço de soluções, isto é, sejam pertencentes a diferentes vizinhanças de pesquisa. Daí, utilizaram estas boas soluções como soluções iniciais para as RN aplicarem uma busca local intensiva, gerando, dessa forma, soluções de melhora impossíveis de serem geradas aplicando somente o AG ou as RN. As soluções de melhora obtidas pelas RN também são introduzidas à população corrente do AG, melhorando, assim, as populações das futuras iterações.

D - Colônia de Formigas

A metaheurística Colônia de Formigas (CF), proposta por Dorigo (1992) para a resolução de problemas combinatórios, é um método inspirado no comportamento de

colônias reais de formigas. O método simula o comportamento de algumas espécies de formigas que cooperam-se por um meio de comunicação muito simples. Ao se deslocarem do ninho até a fonte de alimento, as formigas deixam um rastro (substância denominada feromônio) que é utilizado para se comunicarem quimicamente, criando caminhos (ou trilhas), guiando, dessa forma, as outras formigas. Segundo Chirstodoulou (2009), a característica essencial da metaheurística CF é a combinação de informações conhecidas a priori sobre a estrutura de uma solução promissora com as informações obtidas sobre a estrutura de uma boa solução gerada. Chirstodoulou (2009) propôs uma metodologia baseada na CF para a resolução do PSAP, mas sem considerar restrições de recursos.

Merkle *et al.* (2002) apresentaram a primeira aplicação da metaheurística CF para o PSAPRRP. No trabalho dos autores, uma formiga, que representa uma solução, corresponde a uma aplicação do MSGS, onde a próxima atividade a ser sequenciada é selecionada utilizando uma regra de prioridade baseada no valor ponderado da data mais tarde possível para o início da execução das atividades e, nos feromônios que representam as informações de formigas anteriores. Um feromônio τ_{ij} descreve o quanto é promissor sequenciar a atividade *j* como a *i*-ésima atividade na solução.

Tseng e Chen (2006) propuseram uma metaheurística híbrida combinando a CF com o AG, denominada ANGEL, para a resolução do PSAPRRP. O procedimento ANGEL inicia-se utilizando a CF para a geração de um conjunto de soluções que será utilizado como população inicial para o AG. Daí, o AG é executado e o feromônio da CF é atualizado sempre que for encontrada uma solução de melhora. Quando o AG termina a sua busca, a CF é utilizada novamente com o novo feromônio. Dessa forma, a CF e o AG se alternam cooperativamente na busca ao longo do espaço de soluções.

E - GRASP

A metaheurística GRASP (*Greedy Randomized Adaptive Search Procedure*), proposta por Feo e Resende (1995), é um procedimento iterativo com múltiplos reinícios, onde cada iteração é composta por duas fases: uma fase construtiva, na qual uma solução *s* é gerada e, uma fase de busca local (refinamento), na qual um ótimo local na vizinhança da solução *s* é obtido. A melhor solução encontrada ao longo de todas as iterações do método é retornada como resultado. O número máximo de iterações é um parâmetro de entrada do método e é denotado por $GRASP_{max}$. O pseudocódigo com a estrutura básica do método GRASP é descrito na Figura 11.

Procedimento GRASP
Entrada: *f*(.), *g*(.), *N*(.), $GRASP_{max}$, *s*
Saída: *s*
$f^* \leftarrow \infty$;
Para (*Iter* = 1 até $GRASP_{max}$) **faça**
  Construcao(*g*(.), α, *s*);
  BuscaLocal(*f*(.), *N*(.), *s*);
  Se (*f*(*s*) < *f**) **então**
    $s^* \leftarrow s$;
    $f^* \leftarrow f(s)$;
  Fim_se;
Fim_para;
$s \leftarrow s^*$;
Retorna *s*;

Figura 11: Pseudocódigo da Estrutura básica do método GRASP

Na fase construtiva do método, uma solução é iterativamente construída. A cada iteração desta fase, os próximos elementos candidatos a serem incluídos na solução são armazenados em uma lista *LC* de candidatos, seguindo um critério de ordenação pré-definido. O processo de seleção de um elemento pertencente à *LC* é baseado em uma função adaptativa gulosa *g* que estima o benefício ao escolher cada um dos elementos. A fase construtiva é adaptativa devido ao fato de os benefícios associados à escolha de cada elemento serem atualizados a cada iteração desta fase, refletindo as mudanças oriundas da escolha do elemento anterior. A componente probabilística do método reside no fato de que cada elemento é selecionado aleatoriamente a partir de um subconjunto restrito formado pelos melhores elementos que compõem *LC*. Este subconjunto é denominado lista de candidatos restrita *LCR*. Esta técnica de seleção permite que diferentes soluções possam ser geradas em cada iteração do método. O método GRASP procura conjugar bons aspectos dos algoritmos puramente gulosos com aqueles dos procedimentos aleatórios de construção de soluções.

A Figura 12 descreve o pseudocódigo da fase construtiva do método GRASP, onde $\alpha \in [0, 1]$ é um parâmetro que controla o nível de gulosidade e aleatoriedade do procedimento *Construcao*. Para $\alpha = 0$ o procedimento gera soluções totalmente gulosas e para $\alpha = 1$ gera soluções puramente aleatórias. O parâmetro α, que determina o tamanho da *LCR*, é, basicamente, o único parâmetro do método a ser ajustado na sua implementação.

Procedimento *Construcao*
Entrada: $g(.)$, α, s
Saída: s
$s \leftarrow \phi$;
Inicialize a lista *LC* de candidatos;
Enquanto ($LC \neq \phi$) **faça**
  $g(t_{min}) \leftarrow min\{g(t) \mid t \in LC\}$;
  $g(t_{max}) \leftarrow max\{ g(t) \mid t \in LC \}$;
  $LCR \leftarrow \{ t \in LC \mid g(t) \leq g(t_{min}) + \alpha(g(t_{max}) - g(t_{min}))\}$;
  Selecione, aleatoriamente, um elemento $t \in LCR$;
  $s \leftarrow s \cup \{t\}$;
  Atualize o conjunto *LC*;
Fim_enquanto;
Retorna *s*;

Figura 12: Pseudocódigo da fase construtiva do método GRASP

Assim como em muitas técnicas determinísticas, as soluções geradas pela fase construtiva do método GRASP não são localmente ótimas com respeito à definição de vizinhança adotada. Daí a importância da fase de busca local, a qual tem como objetivo melhorar a solução construída. O pseudocódigo de um procedimento básico de busca local para um problema de minimização, utilizando uma estrutura de vizinhança $N(.)$, é descrito na Figura 13.

```
Procedimento BuscaLocal
Entrada: f(.), N(.), s
Saída: s
V ← {s´ ∈ N(s) | f(s´) < f(s)};
Enquanto (|V| > 0) faça
   Determine uma solução s´ ∈ V;
   s ← s´ ;
   V ← {s´ ∈ N(s) | f(s´) < f(s)};
Fim_enquanto;
Retorna s;
```

Figura 13: Pseudocódigo da fase de busca local do método GRASP

A eficiência da busca local depende da qualidade da solução construída na primeira fase do método. O procedimento construtivo tem, então, um papel importante na busca local, uma vez que as soluções construídas devem constituir bons pontos de partida para a busca local, tornando-a mais rápida.

F – VNS

A metaheurística *Variable Neighborhood Search* - VNS (MLADENOVIC e HANSEN, 1997) é um método de busca local que consiste em explorar o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança. O pseudocódigo do procedimento VNS é apresentado na Figura 14, onde r representa o número de diferentes estruturas de vizinhança utilizadas.

```
Procedimento VNS
Entrada: r
Saída: s
Determine uma solução inicial s0;
s ← s0;
Enquanto (Critério de Parada = Falso) faça
   k ← 1;
   Enquanto (k ≤ r) faça
      BuscaLocal(s) ( ∀ s´ ∈ Nk(s));
      Se (f(s´) < f(s)) então
         s ← s´;
         k ← 1;
      senão k ← k++;
      Fim_se;
   Fim_enquanto;
Fim_enquanto;
Retorna s;
```

Figura 14: Pseudocódigo da estrutura básica do VNS

O VNS parte de uma solução inicial qualquer s, a qual é submetida a um procedimento de busca local. No processo de busca local, determina-se uma estrutura de vizinhança inicial ($N_1(s)$) e, se alguma solução vizinha $s´$ gerada for melhor que a solução corrente s, a busca se reinicia com $s´$ como solução corrente, utilizando a mesma estrutura de vizinhança. Caso contrário, continua-se a busca utilizando a próxima estrutura de vizinhança ($N_{k+1}(s)$). Este procedimento é encerrado quando algum critério de parada for atingido, retornando, assim, a melhor solução encontrada ao longo da busca.

G - ILS

A metaheurística *Iterated Local Search* – ILS, proposta por Lourenço *et al*. (2002), é baseada na simples ideia de que um procedimento de busca local pode ser melhorado gerando-se novas soluções de partida, as quais são obtidas através de perturbações em uma solução ótima local. Na aplicação do ILS devem ser especificados um procedimento para gerar uma solução inicial s_0 para o problema, um procedimento de busca local que retornará uma solução melhorada *s´´*, um procedimento de perturbação, que modifica a solução corrente *s* guiando a uma solução intermediária *s´* e, um critério de avaliação que decide a qual solução a próxima perturbação será aplicada. O pseudocódigo básico do ILS é apresentado na Figura 15.

Procedimento ILS
Entrada: Critério de Parada
Saída: *s*

Determine uma solução inicial s_0 ;
s ← *BuscaLocal*(s_0);
Enquanto (Critério de Parada = *Falso*) **faça**
  s´ ← *Perturbacao*(*s*);
  s´´ ← *BuscaLocal*(*s´*);
  s ← *CriterioAceitacao*(*s*, *s´´*);
Fim_enquanto;
Retorna *s*;

Figura 15: Pseudocódigo da metaheurística ILS

O ILS inicia-se com a geração de uma solução inicial s_0, a qual é aplicada uma busca local, obtendo uma solução ótima local *s*. De modo geral, qualquer método de busca local pode ser usado, mas o desempenho do ILS com respeito à qualidade da solução final e a velocidade de convergência dependem fortemente do método escolhido. Após a primeira busca local é aplicada uma perturbação à solução *s* gerando uma nova solução de partida *s´*. O mecanismo de perturbação deve ser capaz de escapar do ótimo local corrente e permitir que a busca local explore diferentes soluções e, ao mesmo tempo, guardar características do ótimo local corrente e evitar o reinício aleatório. Em seguida é feita uma busca local na vizinhança da solução *s´*, gerando uma solução *s´´*, a qual é submetida a um critério de aceitação. O critério de aceitação é usado para decidir de qual solução se continuará a exploração, bem como qual será a perturbação a ser aplicada. Este procedimento é realizado até algum critério de parada ser atingido, onde é retornada a melhor solução encontrada durante sua execução.

Um aspecto importante do critério de aceitação e da perturbação é que eles induzem aos procedimentos de intensificação e diversificação. A intensificação da busca ao redor da melhor solução encontrada é obtida, por exemplo, pela aplicação de "pequenas" perturbações sobre ela. A diversificação, por sua vez, pode ser realizada aceitando-se quaisquer soluções *s´´* e aplicando "grandes" perturbações na solução ótima local.

Segundo Lourenço *et al*. (2002), o ILS é um método de busca local que procura focar a busca não no espaço completo de soluções, mas em um pequeno subespaço definido por soluções que são ótimas locais de determinado procedimento de otimização.

3.4.3. Algoritmos Evolucionários

Nas décadas de 50 e 60, vários pesquisadores estudaram os sistemas evolucionários naturais com a ideia de que a evolução poderia ser utilizada como uma ferramenta para resolver problemas de engenharia. Desses estudos surgiram os Algoritmos Evolucionários (AEs), fundamentados no princípio de que os indivíduos mais adaptados ao meio ambiente possuem maior probabilidade de sobrevivência. Os AGs aparecem como principal exemplo de AE. A estrutura básica de um AE pode ser vista na Figura 16.

Procedimento AEs
Gere uma população inicial $P_0 = X_1, X_2, ..., X_p$;
$t \leftarrow 0$;
Enquanto (Critério de Parada = *Falso*) **faça**
Etapa de Avaliação
Determine a aptidão $f(X_i)$ para cada indivíduo X_i, $i = 1, 2, ..., p$;

Etapa de Seleção
Selecione indivíduos para S_t (grupo de recombinação); (O mesmo indivíduo de P_t pode aparecer várias vezes em S_t)

Etapa de Reprodução
Formar pares de indivíduos em S_t e para cada par de indivíduos:
Recombine o par com probabilidade p_{cross};
Copie o descendente resultante para S_{t+1}; (O par de cromossomas é eliminado e substituído pelo seu descendente)
Copie os pares de indivíduos para o conjunto S_{t+1} com probabilidade 1 - p_{cross};
Para (cada indivíduo de S_{t+1}) **faça**
Aplique a mutação no indivíduo com probabilidade p_{mut};
Copie os indivíduos transformados para P_{t+1};
Copie o indivíduo em P_{t+1} com probabilidade 1-p_{mut};
Fim_para;
$t \leftarrow t++$;
Retorne para a **Etapa de Avaliação**;
Fim_enquanto;
Retorna P_t;

Figura 16: Pseudocódigo de um Algoritmo Evolucionário

Os AEs buscam a melhor solução através da melhoria contínua de uma população formada por soluções potenciais denominadas indivíduos ou cromossomos. Cada evolução de uma população é denominada geração, a qual consiste de um ciclo de: Avaliação, realizada através de uma função de aptidão; Seleção, baseada na determinação da aptidão dos indivíduos; e Reprodução, que consiste na criação de descendentes através de Recombinações e Mutações de indivíduos selecionados.

Rogalska *et al.* (2008) utilizaram um AE Híbrido (AEH) para a otimização do tempo/custo no PSAPRRP. O AEH utilizado pelos autores foi proposto, originalmente, por Bozejko e Wodecki (2006) como um método geral para a resolução de problemas de otimização discreta. O AEH utilizado por Rogalska *et al.* (2008) inicia-se com a geração de uma população P_0, que pode ser criada aleatoriamente. O melhor elemento de P_0 é adotado como solução sub-ótima (s*). Definido que as populações possuem um número fixo de elementos (n), uma nova população P_{i+1} é gerada da seguinte forma: para a população corrente P_i, um conjunto LM_i com as melhores soluções pertencentes a P_i é

fixado. Elementos com o mesmo valor das melhores soluções também são fixados, formando, assim, um conjunto de elementos denotado por FS_{i+1}. Cada elemento da nova população P_{i+1} é gerado através da recombinação de elementos do conjunto FS_{i+1}. Alguns elementos de P_i, não fixados em FS_{i+1}, são designados aleatoriamente para as posições restantes em P_{i+1}, gerando, dessa forma, uma população com n elementos. Se existir algum indivíduo β pertencente a P_{i+1} tal que $f(\beta) < f(s^*)$, então β é adotada como s^*. O AEH para quando um determinado número de gerações é criado.

De acordo com Martínez *et al.*(2009), muitos problemas relacionados a projetos em engenharia podem ser formulados como um problema de otimização multiobjetivo (OM), como é o caso do PSAPRRP. Ballestín e Blanco (2011) afirmam que o PSAPRRP é claramente um problema de OM, visto que os gestores de projetos frequentemente buscam finalizá-los o mais rápido possível com o mínimo custo e com máxima qualidade. Apesar de vários autores o considerarem um problema cuja resolução envolve diversos e conflitantes objetivos, poucos trabalhos têm sido desenvolvidos utilizando este enfoque.

Segundo Martínez *et al.*(2009), a OM oferece vantagens em relação à otimização mono-objetivo porque gera soluções com diferentes contrapartidas entre os distintos objetivos do problema abordado.

Devido a estes fatos, neste trabalho o PSAPRRP é abordado como um problema de OM, considerando dois objetivos conflitantes: tempo e custo. Uma descrição da OM, bem como as principais técnicas de resolução de problemas de OM encontrados na literatura, é apresentada no capítulo seguinte. É apresentada, também, uma abordagem multiobjetivo e algumas aplicações da OM para o PSAPRRP.

Capítulo 4

4. OTIMIZAÇÃO MULTIOBJETIVO

Muitos problemas encontrados no mundo real apresentam inúmeros objetivos a serem otimizados, que são, na maioria das vezes, conflitantes entre si. Ou seja, a melhoria de algum(uns) dele(s) pode causar, consequentemente, a piora de outro(s). Problemas com essa característica são denominados problemas de OM. Um problema de OM pode ser formulado da seguinte maneira:

$$Min\ (Max) \qquad z = f(x) = [f_1(x), f_2(x), ..., f_r(x)]$$

Sujeito a

$$x \in X^*$$

em que:

- $f_1(x)$, $f_2(x)$, ..., $f_r(x)$ são as funções-objetivo;
- $x = (x_1, x_2, ..., x_n)$ é o vetor de variáveis de decisão;
- X^* é o conjunto de soluções viáveis do problema. X^* é determinado levando em consideração as restrições do problema, sendo Z^* seu conjunto imagem, denominado espaço objetivo viável.

A resolução de um problema de OM consiste na obtenção de um conjunto de variáveis (soluções) que satisfaça algumas restrições e otimize uma função constituída por diversos termos ou funções-objetivo. Sendo assim, é necessário que o tomador de decisão escolha a variável (solução) que melhor atenda às suas necessidades.

De acordo com Martínez *et al.*(2009), um problema de OM é resolvido, geralmente, por meio da construção de um diagrama de Pareto, onde cada ponto do diagrama representa uma solução do espaço de soluções viáveis do problema. Tal diagrama auxiliará o tomador de decisão na escolha da solução mais adequada. Contudo, a obtenção do correto diagrama de Pareto não é uma tarefa trivial, ou até mesmo impossível de se realizar, pois essa tarefa depende da complexidade das funções-objetivo a serem otimizadas, das restrições do problema e, em particular, do tipo de método de resolução utilizado. Portanto, é muito importante a escolha de uma técnica que determine o conjunto de soluções mais adequado para auxiliar o tomador de decisão.

Vários métodos de resolução de problemas de OM podem ser encontrados na literatura, dentre os quais destacam-se os métodos clássicos e os metaheurísticos, descritos nas seções a seguir. Antes da descrição dos métodos de resolução de problemas de OM é necessário definir-se o conceito de ótimo de Pareto, conceito esse de extrema importância para o desenvolvimento dos métodos.

4.1. Ótimo de Pareto

Pareto formulou em 1896 (PARETO, 1896) o conceito de ótimo que continua sendo muito importante para a análise multiobjetivo. Essa definição é baseada no conceito intuitivo de que uma solução x é escolhida como ótima se nenhum objetivo pode ser melhorado sem prejudicar pelo menos um outro objetivo. Entretanto, o ótimo de Pareto quase sempre não se resume a uma única solução, mas a um conjunto de soluções denominadas soluções eficientes ou soluções Pareto-ótimas. Para a

determinação do conjunto de soluções Pareto-ótimas é utilizado o conceito de dominância de Pareto, descrito a seguir.

4.1.1. Dominância de Pareto

Uma solução viável $x \in X^*$ é chamada Pareto-ótima se não existir nenhuma outra solução $x' \in X^*$ tal que $f_k(x') \leq f_k(x)$ para todo $k = 1, 2, ..., r$ e $f_j(x') < f_j(x)$ para algum j. Sendo assim, x é chamada de solução não-dominada e passa a pertencer ao conjunto Pareto-ótimo.

Caso contrário, se existir alguma solução x', tal que $f_k(x') \leq f_k(x)$ para todo $k = 1, 2, ..., r$ e $f_j(x') < f_j(x)$ para algum j, x será uma solução dominada por x' e, portanto, não fará parte do conjunto Pareto-ótimo.

No caso de duas soluções x e x', onde $f_j(x') < f_j(x)$ para algum j e $f_i(x') > f_i(x)$ para algum i, x e x' são ditas não-dominadas entre si ou indiferentes.

Em decorrência dessas definições, todas as soluções pertencentes ao conjunto Pareto-ótimo são indiferentes entre si. A Figura 17 exemplifica o conceito de dominância de Pareto, utilizando como referência a solução representada pelo ponto C, para um problema de minimização de f_1 e f_2.

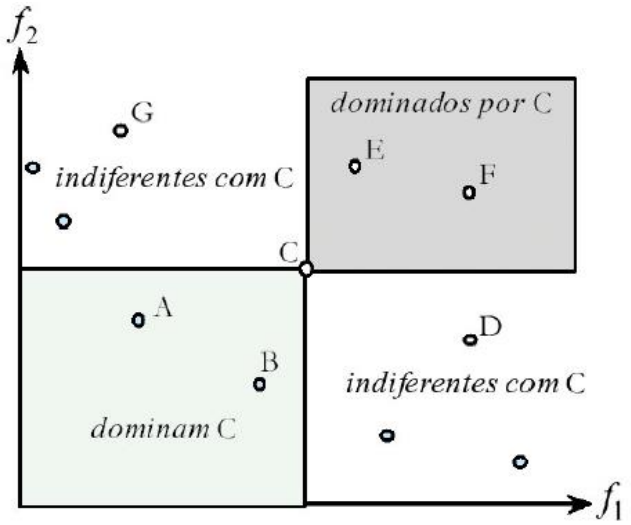

Figura 17: Exemplificação do conceito de dominância de Pareto

Como pode ser visto, quando são considerados objetivos conflitantes em um problema de OM, em geral, não existe uma solução única que seja ótima em relação a todos os objetivos. Por exemplo, na Figura 17, a solução do ponto A possui um valor menor para f_1, mas um valor maior para f_2, se comparada com a solução do ponto B, de modo que, a diminuição de uma função-objetivo implica no aumento da outra. De acordo com o conceito de dominância de Pareto, as soluções A e B são, então, indiferentes.

O conjunto imagem (Z^*) do conjunto Pareto-ótimo é denominado fronteira Pareto-ótima. Esta relação pode ser vista na Figura 18, onde as soluções são divididas por graus de dominância.

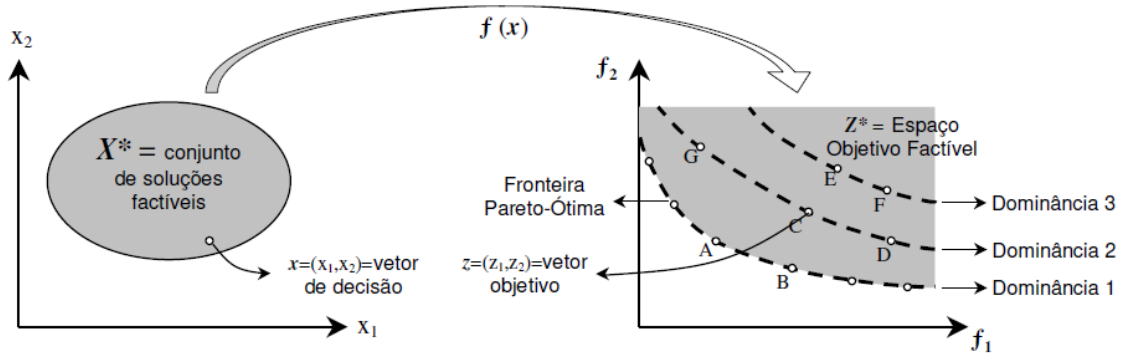

Figura 18: Representação de um Diagrama de Pareto

4.2. Métodos de Resolução de Problemas de OM

Segundo Hwang e Massud (1979), os métodos de resolução de problemas de OM podem ser divididos em três categorias, dependendo de como o tomador de decisão interfere no processo de otimização:

(1) Articulação de preferências antes do início da otimização: informações dadas pelo tomador de decisão levam a um problema de otimização mono-objetivo onde o objetivo, muitas vezes, é uma função linear dos objetivos originais do problema de OM;
(2) Articulação de preferências após o fim da otimização: um conjunto de soluções (preferencialmente o conjunto Pareto-ótimo ou próximo dele) é determinado, só então o tomador de decisão seleciona a solução mais adequada dentro do conjunto;
(3) Articulação de preferências durante a otimização: a otimização é dividida em etapas, onde, após cada uma, certo número de soluções alternativas é apresentado ao tomador de decisão de modo que essas servirão de guia para a próxima etapa.

Basicamente, os principais métodos de resolução de problemas de OM podem sem classificados em dois grupos: os métodos clássicos e os metaheurísticos.

Nos métodos clássicos, a definição dos critérios de geração de soluções ocorre antes da execução da mesma. Isso pode ser feito combinando os objetivos do problema em um único objetivo, segundo uma determinação de pesos de preferência do tomador de decisão. Como consequência, o problema acaba se constituindo na otimização de um único objetivo, cuja resolução requer métodos mono-objetivos.

A classe de métodos metaheurísticos envolve o mapeamento de soluções viáveis, direcionando para soluções Pareto-ótimas. Ao término da busca, é disponibilizado ao tomador de decisão um conjunto de soluções aproximadas ou Pareto-ótimas. Cabe ao tomador de decisão escolher a(s) solução(ões) que lhe convier.

São apresentados alguns dos principais métodos clássicos e metaheurísticos de resolução de problemas de OM nas seções seguintes.

4.2.1. Métodos Clássicos

Os métodos clássicos de resolução de problemas de OM consistem, basicamente, em transformar o vetor função-objetivo em uma função-objetivo escalar. Daí, então, é utilizado algum método de otimização para resolver o problema mono-objetivo resultante. Mas é recomendável a utilização de vários métodos, de forma a permitir a comparação entre as soluções obtidas e a escolha de soluções mais confiáveis.

Dentre os vários métodos clássicos encontrados na literatura, destacam-se aqui os métodos de Ponderação dos Objetivos e do Critério Global. No método de Ponderação dos Objetivos, ao se atribuir, para cada objetivo, diferentes coeficientes de ponderação, permite-se que os tomadores de decisão priorizem um determinado objetivo. O método do Critério Global é indicado nos casos onde se deseja obter uma solução que atenda a todos os objetivos, considerando-os com o mesmo nível de importância.

Para a utilização dos métodos clássicos de resolução de problemas de OM destacados deve-se, primeiramente, definir a solução ideal (f^0) para cada função-objetivo considerada no problema. Para determinar a solução ideal, deve-se encontrar, separadamente, o menor valor possível para cada função-objetivo. Assim, tem-se que:

$$f_i^0 = Min\ f_i(x),\ i = 1, \ldots, r$$

A - Método de Ponderação dos Objetivos

O método de Ponderação dos Objetivos é um dos mais populares para OM. Neste método, os problemas de OM são substituídos por um problema de otimização escalar através da criação de uma função mono-objetivo, que é a soma ponderada dos diversos objetivos. Isto é, dadas *r* funções-objetivo, é criada uma função mono-objetivo *f*, dada por:

$$f(x) = \sum_{i=1}^{r} w_i\ f_i(x)$$

em que $w_i \geq 0$ são os coeficientes de ponderação (ou pesos) que representam a importância relativa de cada função-objetivo f_i. Geralmente assume-se que:

$$\sum_{i=1}^{r} w_i = 1$$

Quando não estão expressas na mesma unidade, as funções-objetivo precisam ser multiplicadas por uma constante d_i, sendo, assim, expressas de forma adimensional (Deb, 2001). Portanto, a função-objetivo pode ser escrita da seguinte forma:

$$f(x) = \sum_{i=1}^{r} w_i\ f_i(x)\, d_i$$

De acordo com Deb (2001), geralmente, melhores resultados são obtidos quando $d_i = 1/f_i^0$, para $i = 1, \ldots, r$.

A principal dificuldade ao utilizar tal método é encontrar pesos adequados para as funções-objetivo. Porém, é possível encontrar algumas soluções do conjunto Pareto-ótimo variando os valores de w_i.

B - Método do Critério Global

Neste método, a solução ótima é um vetor de variáveis de decisão que minimiza algum critério global. Cabe ao tomador de decisão definir a função que melhor descreve o critério global, de forma que se possa obter a solução mais próxima possível da solução ideal. A função que descreve este critério global tem como forma mais comum:

$$f(x) = \sum_{i=1}^{r} \left(\frac{f_i^0 - f_i(x)}{f_i^0} \right)^S$$

Diversos valores podem ser utilizados para *s*. Desse modo, o problema é determinar qual valor será mais satisfatório. De acordo com Osyczaka (1984), os valores mais utilizados são *s* = 1 e *s* = 2. Boychuk e Ovchimikov (1973) propuseram *s* = 1 e Salukvadze (1974) propôs *s* = 2 em seus trabalhos, mas outros valores para *s* podem também ser usados. A solução do problema diferirá de acordo com o valor escolhido para *s*.

4.2.2. Métodos Metaheurísticos

A busca por soluções Pareto-ótimas não é uma tarefa trivial devido, principalmente, às elevadas dimensões dos problemas reais, o que gera um grande número de soluções possíveis, tornando impraticável a enumeração de todas. Visto esta dificuldade, há uma crescente utilização de métodos metaheurísticos, combinados com conceitos de OM, para a resolução de problemas deste tipo. Isto ocorre devido ao fato de as metaheurísticas alcançarem boas soluções sem terem que enumerar todas as possíveis. Outra característica importante dos métodos metaheurísticos, de acordo com Viana e Sousa (2000), é a facilidade de implementação.

Esta classe de métodos de resolução de problemas de OM utiliza metaheurísticas para gerar e analisar diferentes soluções, bem como determinar a fronteira Pareto-ótima. Como principais métodos metaheurísticos temos o Pareto *Simulated Annealing* – PSA (Czyzak e Jaskiewicz (1998)), a Busca Tabu Multiobjetivo (Hansen (1997)), o *Nondominated Sorting Genetic Algorithm II* – NSGA-II (Deb *et al*. (2000)) e o *Strength Pareto Evolutionary Algorithm II* - SPEA-II (Zitzler *et al.* (2001)). Tais métodos são descritos a seguir.

A - Pareto Simulated Annealing

Segundo Viana e Sousa (2000), o Pareto *Simulated Annealing* (PSA), proposto por Czyzak e Jaszkiewicz (1998), consiste em uma adaptação da metaheurística *Simulated Annealing* para problemas de OM, sendo o resultado da otimização um conjunto de soluções Pareto-ótimas baseado em objetivos específicos.

Para problemas de minimização, o algoritmo PSA tem a estrutura descrita na Figura 19, onde $P(s, s', T, \Lambda^s)$ é a probabilidade de aceitação de uma solução, que depende das soluções *s* e *s´*, da temperatura (*T*) e dos pesos associados aos objetivos ($\Lambda^s = [w_1, w_2, ..., w_k]$) na solução *s* e α é uma constante positiva com valor próximo de 1.

Algoritmo PSA

Determine um conjunto inicial de soluções $S \subset D$;
$M \leftarrow$ soluções não-dominadas de $M \cup S$;
$T \leftarrow T_0$;
Enquanto (Condição de Parada = *Falso*) **faça**
  Para (cada solução $s \in S$) **faça**
    Gere uma solução vizinha $s' \in N(s)$;
    $M \leftarrow$ soluções não-dominadas de $M \cup \{s'\}$;
    Selecione a solução $s'' \in S$ mais próxima de s e não-dominada em relação a s;
    Se (não houver tal solução s'' ou esta é a primeira iteração com s) **então**
      Determine aleatoriamente $w_k \geq 0 \ \forall k \mid \sum_k w_k = 1$;
    senão
      Para (cada objetivo f_k) **faça**

$$w_k = \begin{cases} \alpha\, w_k^s \ \ se \ \ f_k(s) \geq f_k(s'') \\ w_k^s / \alpha \ \ se \ \ f_k(s) < f_k(s'') \end{cases};$$

        Normalize os pesos tal que $\sum_k w_k = 1$;
      Fim_para;
    Fim_se;
    $s \leftarrow s'$ com probabilidade $P(s, s', T, \Lambda^s)$;
    Se (as condições de atualização da temperatura forem atendidas) **então**
      Reduzir T;
    Fim_se;
  Fim_para;
Fim_enquanto;
Retorna M;

Figura 19: Estrutura básica do Algoritmo PSA para problemas de minimização

No algoritmo apresentado na Figura 19 tem-se:

D = conjunto de soluções viáveis do problema;
$s, s' \in D$ = soluções viáveis do problema;
S = conjunto de soluções corrente;
M = conjunto de soluções não-dominadas corrente;
T_0 = temperatura inicial;
T = temperatura corrente;
$N(s)$ = vizinhança da solução s.

A distância entre duas soluções s e s', que é utilizada no PSA para determinar a proximidade entre as soluções, é calculada por:

$$\| f(s) - f(s') \|_w = \max_{k=1,\ldots r} \{ w_k^s \times | f_k(s) - f_k(s') | \}$$

Diferentes tipos de funções de probabilidade podem ser encontrados na literatura, porém a mais utilizada é a definida por Hapke *et al.* (1998), onde:

$$P(s, s', T, \Lambda^s) = \min \{1;\ \max_{k=1,\ldots,r} \{ e^{w_k (f_k(s) - f_k(s'))/T} \}$$

No PSA é considerado um conjunto de soluções não-dominadas corrente. Este conjunto é atualizado, iterativamente, através da obtenção de soluções vizinhas que podem ser aceitas ou não de acordo com uma função de probabilidade. Uma solução vizinha que domina alguma solução do conjunto corrente é sempre aceita, passando a fazer parte do conjunto, sendo as soluções dominadas por ela retiradas do mesmo. Uma probabilidade de aceitação (menor que 1) é determinada para soluções dominadas por alguma solução do conjunto corrente. Entretanto, a questão não é tão fácil quando a solução vizinha e alguma solução do conjunto corrente são não-dominadas (indiferentes) entre si. Diferentes regras de aceitação para este caso são encontradas na literatura, como as sugeridas e discutidas por Serafini (1992). A regra mais utilizada neste caso é aceitar sempre a solução vizinha.

De acordo com Viana e Sousa (2000), com o objetivo de gerar boas soluções e uniformemente espalhadas no conjunto de soluções viáveis, um conjunto de pesos é associado a cada um dos objetivos. Estes pesos variam de tal modo que, para objetivos que deseja-se melhorar, eles são aumentados e, diminuídos para os outros. Aumentando os pesos, a probabilidade de aceitação de uma solução onde aquele objetivo é melhor também aumenta. Esta variação nos pesos conduzirá a um conjunto de soluções mais bem distribuídas ao longo da fronteira de soluções Pareto-ótimas.

B - Busca Tabu Multiobjetivo

Segundo Viana e Sousa (2000), o algoritmo Busca Tabu Multiobjetivo (BTMO), proposto por Hansen (1997), funciona de forma parecida com o PSA na questão de utilizar pesos para "forçar" a busca de soluções para uma determinada direção. Entretanto, o BTMO e o PSA possuem diferenças fundamentais de estrutura, semelhantes às que ocorrem nas versões mono-objetivo dos algoritmos.

Para problemas de minimização, o algoritmo BTMO tem a estrutura apresentada na Figura 20.

Algoritmo BTMO

Determine um conjunto inicial de soluções $S \subset D$;
$LT(s) \leftarrow \phi \ \ \forall s \in S$;
$M \leftarrow \phi$;
$Iter \leftarrow 1$;
$\pi_k \leftarrow \frac{1}{k} \ \ \forall k$;
Enquanto (Critério de Parada = *Falso*) **faça**
  Para (cada solução $s \in S$) **faça**
    $w \leftarrow 0$;
    Para (cada solução $s' \in S \mid s'$ não é dominada por s e $f(s') \neq f(s)$) **faça**
      $\lambda \leftarrow g(d(f(s), f(s'), \Pi))$;
      Para (cada objetivo $k \mid f_k(s) < f_k(s')$) **faça**
        $w_k \leftarrow w_k + \pi_k \times \lambda$;
      Fim_para;
    Fim_para;
    Se ($w = 0$) **então**
      Determine aleatoriamente $w_k \geq 0 \ \forall k \mid \sum_k w_k = 1$;
      Encontre a solução $s'' \in SN(s) \mid A(s, s'') \notin LT(s)$ e minimize $w \times f(s'')$;
      Adicione $A(s, s'')$ a $LT(s)$;
      $s \leftarrow s''$;
      $M \leftarrow$ soluções não-dominadas de $M \cup \{s''\}$;
      Atualize Π;
    Fim_se;
    $Iter \leftarrow Iter++$;
  Fim_para;
Fim_enquanto;
Retorna M;

Figura 20: Estrutura básica do Algoritmo BTMO para problemas de minimização

No algoritmo apresentado na Figura 20 tem-se:

D = conjunto de soluções viáveis do problema;
$s, s' \in D$ = soluções viáveis do problema;
S = conjunto de soluções corrente;
M = conjunto de soluções não-dominadas corrente;
$SN(s)$ = sub-vizinhança de s;
$LT(s)$ = lista tabu associada à solução s;
$A(s, s')$ = atributos armazenados em $LT(s)$;
g = função de proximidade;
$\Lambda^s = [w_1, w_2, ..., w_k]$ = conjunto de pesos associados aos objetivos;
$\Pi = \{\pi_1, \pi_2, ..., \pi_k\}$ = fatores de alcance associados aos objetivos.

Como no PSA, um conjunto de soluções não-dominadas corrente é considerado, onde cada solução $s \in M$ possui sua própria Lista Tabu ($LT(s)$). De acordo com Viana e Sousa (2000), estas soluções devem estar dispersas ao longo do espaço de soluções viáveis a fim de permitir uma busca em diferentes áreas da fronteira de soluções Pareto-

ótimas. Para garantir isto, o valor da função de proximidade (*g*) entre duas soluções é calculado, mensurando, assim, a distância entre elas. Esta função é dada por:

$$g(d)=\frac{1}{d(f(s),f(s´),\Pi)}$$

em que:

$$d(f(s),f(s´),\Pi)=\sum_{k=1}^{K}\pi_k\left|f_k(s)-f_k(s´)\right|$$

Um conjunto de pesos, que definirá a direção de busca da otimização, é atribuído a cada objetivo, para cada solução do conjunto corrente. Os valores dos pesos devem ser determinados de forma que cada solução caminhe em direção à fronteira Pareto-ótima.

O seguinte critério de aspiração também é considerado: qualquer solução não-dominada é aceita mesmo se: (a) o movimento é tabu; (b) ela não possuir o menor valor de $w \times f(s´´)$.

Os fatores utilizados para equilibrar o alcance dos objetivos, como sugerido por Steuer (1986), são calculados pela expressão:

$$\pi_k=\frac{1}{Range_k}\left[\sum_{i=1}^{k}\frac{1}{Range_i}\right]^{-1}$$

em que $Range_k$ é a amplitude do alcance do objetivo *k* em *M*.

C - Nondominated Sorting Genetic Algorithm II

Nos últimos anos, vários AGs multiobjetivos têm sido propostos, dentre os quais destaca-se o *Nondominated Sorting Genetic Algorithm II* (NSGA-II), proposto por Deb *et al.* (2000). Segundo Ballestín e Blanco (2011), o NSGA-II é um algoritmo que se utiliza de um *ranking* para dividir uma população em um certo número de fronteiras de dominância, de tal forma que a primeira fronteira contenha os melhores indivíduos. Ou seja, indivíduos da fronteira *n* são melhores que os indivíduos da fronteira *n* + 1. Sendo assim, baseado no critério de dominância de Pareto, o algoritmo agrega o conceito de elitismo que classifica a população total em diferentes categorias de não-dominância ao invés de tratá-las como pertencentes a um único grupo. Isso permite ao algoritmo a priorização dos indivíduos melhor classificados.

O NSGA-II trabalha com uma população pai (*P*) para gerar a população filha (*Q*), como nos AGs convencionais. Seu funcionamento se destaca por possuir dois mecanismos importantes no processo de seleção: o *Fast Non-Dominated Sorting* e o *Crowding Distance* (distância de multidão). A estrutura básica do NSGA-II é apresentada na Figura 21.

Algoritmo NSGA-II

Determine uma população inicial P_0 de tamanho N;
$Q_0 \leftarrow \phi$;
$t \leftarrow 0$;
$R_t \leftarrow P_t \cup Q_t$;
$F \leftarrow$ *fast-non-dominated-sort*(R_t);
$P_{t+1} \leftarrow \phi$;
$i \leftarrow 0$;
Enquanto ($|P_{t+1}| + |F_i| \leq N$)
  Crowding-distance-assignment(F_i);
  $P_{t+1} \leftarrow P_{t+1} \cup F_i$;
  $i \leftarrow i$++;
  Ordenar(F_i);
  $P_{t+1} \leftarrow P_{t+1} \cup F_i[1: (N - | P_{t+1}|)]$;
  $Q_{t+1} \leftarrow$ *cria-nova-população*(P_{t+1});
  $t \leftarrow t$++;
Fim_enquanto;
Retorna P_t;

Figura 21: Estrutura básica do Algoritmo NSGA-II

O NSGA-II inicia-se com a geração de uma população inicial P_0 de tamanho N. Em seguida, cria-se uma população filha Q_0, inicialmente vazia, e, então, as populações P_0 e Q_0 são combinadas em uma única população $R_0 = P_0 \cup Q_0$. O NSGA-II trabalha com populações de tamanho N e as gerações são representadas por t = 0, 1, 2, ..., T, onde T representa o número máximo de gerações.

A cada indivíduo de R_t é atribuído um valor de dominância. Após isso é realizado um ordenamento dos indivíduos de acordo com seus valores de dominância, obtendo, dessa forma, as fronteiras $F = \{F_1, F_2, ..., F_m\}$. Estas fronteiras são, posteriormente, inseridas na nova população P_{t+1}. Para ordenar os indivíduos de R_t, o NSGA-II utiliza o operador denominado *Fast Non-Dominated Sorting*, apresentado na Figura 22.

Algoritmo *Fast Non-Dominated Sorting*

Entrada: P
Saída: F
Para (cada indivíduo $p \in P$) **faça**
  $S_p \leftarrow \phi$;
  $n_p \leftarrow 0$;
  Para (cada indivíduo $q \in P$) **faça**
    Se (p domina q) **então**
      $S_p \leftarrow S_p \cup \{q\}$;
    senão Se (q domina p) **então**
      $n_p \leftarrow n_p$++;
    Fim_se;
  Fim_para;
  Se ($n_p = 0$) **então**
    $p_{rank} \leftarrow 1$;
    $F_1 \leftarrow F_1 \cup \{p\}$;
  Fim_se;
Fim_para;
$i \leftarrow 1$;
Enquanto ($F_i \neq \phi$) **faça**
  $Q \leftarrow \phi$;
  Para (cada indivíduo $p \in F_i$) **faça**
    Para (cada indivíduo $q \in S_p$) **faça**
      $n_q \leftarrow n_q - 1$;
      Se ($n_q = 0$) **então**
        $q_{rank} \leftarrow i + 1$;
        $Q \leftarrow Q \cup \{q\}$;
      Fim_se;
    Fim_para;
  Fim_para;
  $i \leftarrow i$ ++;
  $F_i \leftarrow Q$;
Fim_enquanto;
Retorna F;

Figura 22: Algoritmo *Fast Non-Dominated Sorting*

No algoritmo apresentado na Figura 22, tem-se:

S_p = conjunto de indivíduos que são dominados por p;
n_p = número de indivíduos que dominam p;
Q = conjunto utilizado para armazenar indivíduos para a (i + 1)-ésima fronteira.

Como pode-se ver na Figura 22, o algoritmo *Fast Non-Dominated Sorting* é executado em duas etapas. A primeira etapa analisa todos os indivíduos $p \in P$, comparando-os uns com os outros. O objetivo é classificá-los de acordo com o grau de dominância n_p. Dessa forma, se um indivíduo p for dominado por um número l de indivíduos, o seu valor correspondente de n_p é igual a l. Se ao final da primeira etapa um indivíduo possuir o valor de n_p igual a 0, isso significa dizer que ele não é dominado por nenhum outro indivíduo de P e, portanto, ele fará parte da primeira fronteira (F_1), no qual estarão os melhores indivíduos. A segunda etapa divide os indivíduos de P em

diferentes fronteiras de acordo com os seus valores de dominância, indicados pelos seus respectivos valores de n_p. Cada indivíduo incluído em uma das fronteiras é retirado do sistema, reduzindo em uma unidade o valor de n_p dos indivíduos dominados por ele. Esse procedimento repete-se até que todos os indivíduos sejam alocados em alguma fronteira.

Dado que apenas N indivíduos podem ser inseridos em P_{t+1}, inicia-se a inserção pelos indivíduos que compõem F_1, depois os que compõem F_2 e assim por diante. Cada fronteira F_i deve ser inserida na sua totalidade em P_{t+1} e isto deve acontecer enquanto $|P_{t+1}| + |F_i| \leq N$. Ao inserir em P_{t+1} uma fronteira F_i, tal que $|F_i| > N - |P_{t+1}|$, o algoritmo NSGA-II insere somente os indivíduos de F_i que estejam mais dispersos na fronteira. O algoritmo utiliza um operador denominado *Crowding Distance* com a função de selecionar os indivíduos mais dispersos na fronteira F_i para fazer parte de P_{t+1}. O algoritmo *Crowding Distance* é descrito na Figura 23.

Algoritmo *Crowding Distance Assignment*

Entrada: I
$l \leftarrow |I|$;
Para (cada $i \in I$) **faça**
  $I[i]_{\text{distância}} \leftarrow 0$;
Fim_para;
Para (cada objetivo k) **faça**
  $I \leftarrow ordenar(I, k)$;
  $I[1]_{\text{distância}} \leftarrow I[l]_{\text{distância}} \leftarrow \infty$;
  Para (i = 2 até ($l - 1$)) **faça**

$$I[i]_{\text{distância}} \leftarrow I[i]_{\text{distância}} + \frac{I[i+1].k - I[i-1].k}{f_k^{\max} - f_k^{\min}}$$

  Fim_para;
Fim_para;

Figura 23: Algoritmo *Crowding Distance Assignment*

No algoritmo descrito na Figura 23, tem-se:

I = conjunto de soluções não-dominadas;
$|I|$ = número de indivíduos do conjunto I;
$I[i]_{\text{distância}}$ = distância de multidão para o indivíduo i;
$I[i].k$ = valor do k-ésimo objetivo do i-ésimo indivíduo de I;
f_k^{max} = é o valor máximo do k-ésimo objetivo;
f_k^{min} = é o valor mínimo do k-ésimo objetivo.

O operador *Crowding Distance* ordena os indivíduos de uma mesma fronteira de acordo com sua distância em relação aos seus vizinhos, para cada objetivo. Esse operador permite que haja um melhor espalhamento dos resultados, evitando-se, assim, aglomerações de indivíduos sobre um mesmo ponto. Obtidas as distâncias, as fronteiras F_i são ordenadas decrescentemente em relação às distâncias, e copia-se para P_{t+1} os primeiros $N - |P_{t+1}|$ indivíduos de F_i.

Finalmente, gera-se Q_{t+1} a partir de P_{t+1} utilizando os operadores de seleção, cruzamento e mutação. Na Figura 24 ilustra-se uma iteração do algoritmo NSGA-II.

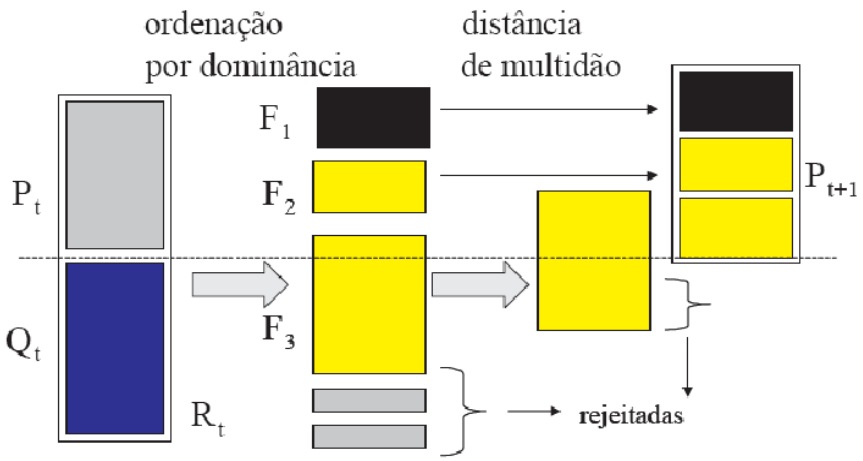

Figura 24: Esquema básico do Algoritmo NSGA-II (Deb et al. (2000))

D - Strength Pareto Evolutionary Algorithm II

O *Strength Pareto Evolutionary Algorithm II* (SPEA-II), desenvolvido por Zitzler *et al.* (2001), é um algoritmo evolutivo de OM que, como o NSGA-II, também utiliza o conceito de elitismo. A estrutura básica do SPEA-II é descrito na Figura 25.

Algoritmo SPEA-II

Determine uma população inicial P_0 de tamanho N;
Gere um arquivo externo $\overline{P}_0$ de tamanho $\overline{N}$;
$\overline{P}_0 \leftarrow \phi$;
$t \leftarrow 0$;
Enquanto (Critério de Parada = *Falso*) **faça**
  Determine o valor da função de aptidão para os indivíduos de P_t e $\overline{P}_t$;
  $\overline{P}_{t+1} \leftarrow$ indivíduos não-dominados de $P_t \cup \overline{P}_t$;
  Se (o tamanho do arquivo $\overline{P}_{t+1}$ exceder $\overline{N}$) **então**
    Reduzir $\overline{P}_{t+1}$ utilizando o operador de truncamento;
  senão Se (o tamanho do arquivo $\overline{P}_{t+1}$ for menor que $\overline{N}$) **então**
    Completar $\overline{P}_{t+1}$ com indivíduos dominados de $P_t \cup \overline{P}_t$;
  Fim_se;
  Se (algum Critério de Parada = *Verdadeiro*) **então**
    $\boldsymbol{A} \leftarrow$ indivíduos não-dominados de $\overline{P_{t+1}}$;
    Pare o Algoritmo;
    Retorna ***A***;
  senão
    Execute o processo de seleção binária com reposição em $\overline{P}_{t+1}$ para preencher o conjunto de pares;
    Aplique operadores de recombinação e mutação ao conjunto de pares e atribua a $\overline{P}_{t+1}$ a população resultante;
    $t \leftarrow t + 1$;
  Fim_se;
Fim_enquanto;
Retorna P_t;

Figura 25: Estrutura básica do Algoritmo SPEA-II

No SPEA-II utiliza-se uma população P_t e um arquivo externo denotado por $\overline{P}_t$. Em $\overline{P}_t$ são armazenadas as soluções não-dominadas encontradas até a iteração corrente *t*. Para o algoritmo, são necessários como parâmetros iniciais: o tamanho de P_t (*N*), o tamanho de $\overline{P}_t$ ($\overline{N}$) e o número máximo de gerações (*T*).

O SPEA-II inicia-se com a geração de uma população inicial P_0 e a criação de um arquivo externo $\overline{P}_0$, inicialmente vazio. A cada iteração, um valor para a função de aptidão (*F*(*i*)) é calculada para cada indivíduo $i \in P_t \cup \overline{P}_t$. No cálculo do valor desta função são utilizados os conceitos de dominância e de densidade. O objetivo é minimizar o valor desta função, ou seja, quanto menor o valor da função de aptidão de um indivíduo, melhor é a sua adaptação. A força de um indivíduo *i* (*S*(*i*)) é dada pelo número de soluções que ele domina, ou seja:

$$S(i) = \left|\{ j \,/\, j \in P_t \cup \overline{P}_t \wedge i \succ j \}\right|$$

em que |.| é a cardinalidade do conjunto, e o símbolo $\succ$ corresponde à relação de dominância de Pareto, onde $i \succ j$ significa que a solução *i* domina a solução *j*.

A aptidão bruta de um indivíduo *i* (*R*(*i*)) é, então, a soma das forças de todos os indivíduos que o dominam, isto é:

$$R(i) = \sum_{j \in P_t \cup \overline{P}_t,\ j \succ i} S(j)$$

A inclusão de um fator de densidade se deve ao fato de que, nos casos em que existam muitas soluções não-dominadas entre si, *S*(*i*) se aproxima de zero para todas as soluções. Nesse caso é necessário haver um mecanismo para privilegiar soluções dentre as não-dominadas. A densidade de um indivíduo *i* (*D*(*i*)) é dada por:

$$D(i) = \frac{1}{dist_{ij}(k) + 2}$$

Para cada indivíduo *i*, as distâncias (no espaço dos objetivos) entre *i* e todos os outros indivíduos *j* de P_t e de $\overline{P}_t$ são calculadas e armazenadas em uma lista. Depois de ordenada em ordem crescente, o *k*-ésimo elemento desta lista representa o termo $dist_{ij}(k)$. É sugerido, na literatura, para *k* o valor $k = \sqrt{N + \overline{N}}$. Sendo assim, a função de aptidão adotada no SPEA-II é dada por:

$$F(i) = R(i) + D(i)$$

Depois de calculados os valores da função de aptidão para os indivíduos de $P_t \cup \overline{P}_t$, executa-se o processo de seleção. Os indivíduos não-dominados de $P_t \cup \overline{P}_t$ são copiados para o novo arquivo externo $\overline{P}_{t+1}$. Se o conjunto de indivíduos não-dominados possuir o tamanho exato do arquivo externo ($\overline{N}$), a seleção está finalizada. Senão, pode-se ocorrer duas situações:

- O tamanho de $\overline{P}_{t+1}$ é menor que $\overline{N}$: neste caso, ordena-se as soluções dominadas de $P_t \cup \overline{P}_t$, de acordo com o valor da função de aptidão e, completa-se $\overline{P}_{t+1}$ com os indivíduos com maior valor da função.
- O tamanho de $\overline{P}_{t+1}$ excede $\overline{N}$: nesse caso, é utilizado um operador de truncamento.

O objetivo do operador de truncamento utilizado no SPEA-II é restringir o tamanho de $\overline{P}_{t+1}$ a $\overline{N}$ indivíduos. Para isso, são removidos de $\overline{P}_{t+1}$ os indivíduos no qual suas distâncias para seus vizinhos mais próximos sejam as menores dentre as

distâncias existentes até que $\overline{P}_{t+1}$ tenha tamanho $\overline{N}$. No caso de empate, avalia-se a segunda menor distância, e assim sucessivamente.

Por último, realiza-se os processos de seleção com reposição, recombinação e mutação sobre $\overline{P}_{t+1}$ para se obter a nova população P_{t+1}.

4.3. Otimização Multiobjetivo Aplicada ao PSAPRRP

Embora o PSAPRRP seja inerentemente um problema multiobjetivo, ele tem sido, na maioria das vezes, resolvido considerando somente um objetivo. Essa abordagem provavelmente foi ignorada por um bom tempo devido à dificuldade de se definir métodos que tratem eficientemente problemas de otimização combinatória de uma forma flexível, além de levarem a boas aproximações para as soluções ótimas. Odedairo e Oladokum (2011), que apresentaram um trabalho explicitando a relevância e a aplicabilidade do PSAP multiobjetivo, também relataram o pequeno número de artigos encontrados na literatura utilizando este enfoque.

Segundo Ballestín e Blanco (2011), o número de possíveis formulações multiobjetivos para o PSAPRRP é muito grande devido aos inúmeros objetivos encontrados na literatura (Slowinski, 1989). Esses objetivos podem ser combinados de diversas formas, formando, assim, novos problemas. Ballestín e Blanco (2011) apresentam algumas funções-objetivo que interessam aos gestores de projeto, dentre as quais os autores destacam:

1) Minimização da duração (*makespan*) do projeto;
2) Minimização dos atrasos na execução das atividades;
3) Minimização do custo do projeto;
4) Minimização da utilização ou do custo dos recursos;
5) Minimização da soma ponderada dos custos relacionados às datas de início de execução das atividades;
6) Minimização da soma das datas de início de execução das atividades;
7) Maximização do valor presente líquido do projeto.

Na formulação multiobjetivo para o PSAPRRP, o conjunto de soluções viáveis (X^*) é determinado pelas restrições de recursos e de precedência.

Na seção a seguir é apresentada uma revisão com alguns trabalhos encontrados na literatura utilizando o enfoque multiobjetivo para o PSAPRRP.

4.3.1. Algumas Aplicações da OM Encontradas na Literatura para o PSAPRRP

Slowinski (1981) foi o primeiro autor a representar explicitamente o PSAPRRP como um problema de OM (Ballestín e Blanco, 2011). Nas últimas décadas, alguns autores têm abordado o PSAPRRP dessa forma, como é o caso de Viana e Sousa (2000), Abbasi *et al*. (2006), Kazemi e Tavakkoli-Moghaddam (2008), Geyer (2009), Hamm *et al*. (2009) e Ballestín e Blanco (2011), dentre outros.

Slowinski (1981) aplicou a programação linear multiobjetivo ao PSAPRRP permitindo a *preempção* das atividades que possuíam múltiplos modos de execução. O problema possuía restrições de recursos renováveis e não-renováveis. Como objetivos, o autor utilizou a minimização da duração e dos custos do projeto. Em seu trabalho, Slowinski (1981) discute também a possibilidade de aplicação da programação por metas e da lógica fuzzy ao PSAPRRP multiobjetivo.

Viana e Sousa (2000) implementaram o PSA e BTMO para resolver o PSAP multiobjetivo com recursos renováveis e não-renováveis, mas sem considerar restrições de recursos. Os recursos são levados em consideração somente em uma das funções-

objetivo utilizadas. Os autores utilizaram três funções-objetivo em seu trabalho: a minimização do *makespan*, a minimização da soma ponderada dos atrasos na execução das atividades e a minimização do somatório das violações na disponibilidade dos recursos.

Abbasi *et al*. (2006) estudaram o PSAPRRP multiobjetivo com recursos renováveis e dois objetivos, a minimização do *makespan* e a maximização do somatório das folgas livres das atividades. Denomina-se folga livre de uma atividade como a disponibilidade de tempo, além de sua duração, que a atividade pode dispor, supondo que sua execução se inicie na data mais cedo possível. Os autores trabalharam com somente um recurso renovável e agregaram os dois objetivos de maneira linear. Daí, então, aplicaram o *SA* ao problema para gerarem diferentes soluções.

Kazemi e Tavakkoli-Moghaddam (2008) apresentaram um modelo matemático para o PSAPRRP multiobjetivo considerando fluxos financeiros positivos e negativos. Os objetivos considerados pelos autores foram a maximização do valor presente líquido e a minimização do *makespan* do projeto. Para a resolução do problema assim formulado os autores utilizaram o NSGA-II.

Geyer (2009) propôs uma metodologia baseada na metaheurística AG para o PSAPRRP utilizando uma função multiobjetivo. O autor levou em consideração objetivos econômicos, ambientais e preferências do projetista.

Hamm *et al*. (2009) aplicaram um algoritmo PSA para a determinação da fronteira Pareto-ótima para o PSAPRRP formulado como multiobjetivo. Durante as iterações do método, soluções são geradas e analisadas através do critério de dominância de Pareto. Segundo os autores, a principal diferença entre o trabalho deles e os publicados até então é a regra de aceitação de novas soluções, que depende da temperatura corrente e do status de dominância da solução vizinha corrente. Dadas duas soluções, o valor da dominância entre elas é determinado por:

$$dom_{s,s'} = \prod_{i=1,\, f_i(s) \neq f_i(s')}^{r} \left(\left| \frac{f_i(s) - f_i(s')}{R_i} \right| \right)$$

em que s é a solução corrente, s' é a solução vizinha, r é o número de objetivos e R_i é a variação do i-ésimo objetivo. R_i é calculado pela diferença entre o maior e o menor valor encontrado para o objetivo i. A probabilidade de aceitação de uma solução vizinha como solução corrente é, então, calculada por:

$$prob = 1 - e^{(-dom_{s,s'} \times T)}$$

Ballestín e Blanco (2011) apresentaram fundamentos teóricos e práticos para a OM aplicada ao PSAPRRP. Os autores propõem também dois algoritmos para a resolução do problema, um heurístico e outro exato. Para isso, eles consideraram como objetivos a minimização do *makespan* e do custo com a disponibilização dos recursos.

Segundo Ballestín e Blanco (2011), ainda existem poucos trabalhos que propõem métodos metaheurísticos eficientes para a resolução do PSAPRRP multiobjetivo.

4.3.2. Exemplo de Aplicação de Métodos Clássicos de OM ao PSAPRRP

Para exemplificar a aplicação de métodos clássicos ao PSAPRRP, é apresentada nesta seção a utilização dos métodos da Ponderação dos Objetivos e do Critério Global na resolução da adaptação de um problema encontrado na literatura. Os métodos foram implementados computacionalmente no *software* IBM ILOG CPLEX 12.1 e executados em um computador *Turion II Dual-Core* 2.20GHz e 4GB de RAM, sob sistema

operacional *Windows* 7 *Home Premium* 64 Bits.

De acordo Martínez *et al.* (2009), a formulação multiobjetivo de um problema é particularmente importante quando os objetivos são conflitantes, ou seja, quando a melhora de um objetivo resulta na piora dos outros. Por isso, na aplicação dos métodos clássicos de OM ao PSAPRRP, aqui apresentada, foram considerados dois objetivos conflitantes: tempo e custo. Ou seja, na resolução do problema buscou-se uma solução que minimize tanto a duração (*makespan*) do projeto como o custo relacionado à data de início de execução de cada atividade.

Sendo assim, o modelo para o PSAPRRP formulado como um problema de OM ficou da seguinte forma:

$$\begin{aligned} &\textit{Min} && y = f(x) = [f_1(x), f_2(x)] \\ &\textit{Sujeito a} && \\ & && x \in X^* \end{aligned}$$

sendo:

- $y_1 = f_1(x) = S_{n+1}$
- $y_2 = f_2(x) = \sum_{i=1}^{n} \frac{c_i}{S_i}$

O problema utilizado consiste no exemplo proposto por Koné *et al.* (2009), descrito na seção 3.3, no qual foram associados custos (c_i) à execução das atividades, conforme apresentado na Tabela 6.

Tabela 6: Dados do problema adaptado

Atividades	1	2	3	4	5	6	7	8	9	10
p_i	7	3	5	5	6	4	5	4	3	7
c_i	200	300	500	100	600	200	500	300	300	200
b_{i1}	0	2	3	3	2	1	1	1	1	3
b_{i2}	2	1	3	2	1	0	3	1	1	1
Sucessoras	3	6, 7	4, 9	11	1	1	5, 8	10	4	9

Fonte: Adaptado de Koné *et al*. (2009).

Como pode ser visto na Tabela 6, o custo associado à execução da atividade 1 é igual a 200, o da atividade 2 é igual a 300 e assim por diante.

As soluções ideais encontradas para os objetivos propostos foram $f_1^0 = 35$ e $f_2^0 = 84$. No método de Ponderação dos Objetivos assumiu-se $d_i = 1/f_i^0$, resultando na seguinte função objetivo:

$$f(x) = w_1 \frac{f_1(x)}{35} + w_2 \frac{f_2(x)}{84}$$

Para a aplicação do método, os valores de w_1 e w_2 utilizados variam de 0,1 até 0,9. Dessa forma, diferentes soluções foram geradas, determinando o conjunto Pareto-ótimo. Os resultados obtidos para o método podem ser observados na Tabela 7.

Tabela 7: Resultados obtidos pelo Método de Ponderação dos Objetivos

w_1	w_2	$f_1(x)$	$f_2(x)$	$f(x)$
0,9	0,1	40	226	266
0,8	0,2	43	176	219
0,7	0,3	50	121	171
0,6	0,4	55	98	153
0,5	0,5	59	86	145
0,4	0,6	60	84	144
0,3	0,7	60	84	144
0,2	0,8	60	84	144
0,1	0,9	60	84	144

No método do Critério Global utilizado, o valor adotado para *s* foi 1, como proposto por Boychuk e Ovchimikov (1973). Sendo assim, considerou-se a seguinte função objetivo:

$$f(x) = \left(\frac{35 - f_1(x)}{35}\right) + \left(\frac{84 - f_2(x)}{84}\right)$$

Para o método do Critério Global, o resultado obtido foi: $f_1(x) = 60$, $f_2(x) = 84$ e $f(x) = 114$. O diagrama de Pareto resultante do método de Ponderação dos Objetivos com a solução encontrada para o método do Critério Global, representada pelo ponto destacado, pode ser visto na Figura 26.

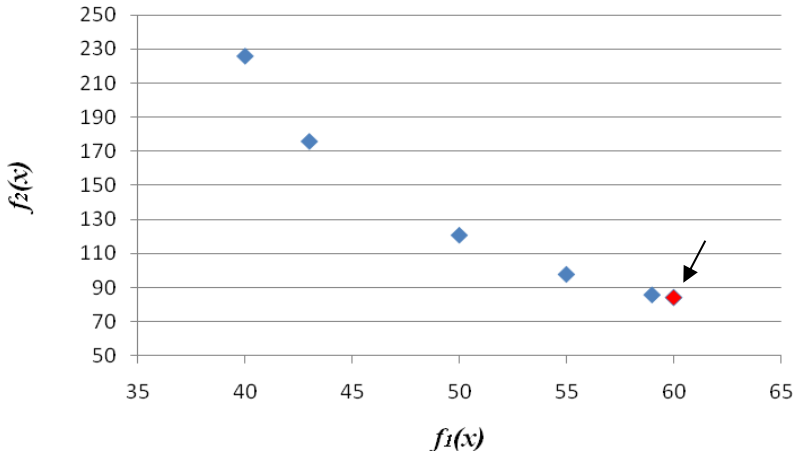

Figura 26: Diagrama de Pareto com as soluções obtidas

Como pode ser observado, a solução obtida para o método do Critério Global é a mesma obtida para algumas configurações do método da Ponderação dos Objetivos.

Os métodos clássicos de Ponderação dos Objetivos e do Critério Global se mostraram eficientes, exigindo baixo esforço computacional, na resolução de problemas com um número pequeno de atividades e recursos como o utilizado. Porém, devido ao fato de o PSAPRRP ser de difícil resolução no caso geral, é importante ressaltar, também, que, para problemas com dimensões elevadas como nos casos reais, os métodos clássicos podem ser menos eficientes computacionalmente. Outra importante dificuldade é a determinação das soluções ideais para os objetivos utilizados. Para objetivos de minimização, a determinação da solução ideal consiste na resolução exata do problema.

Visto isso, e devido à pouca aplicação de métodos metaheurísticos ao PSAPRRP multiobjetivo, são propostos neste trabalho cinco algoritmos para sua resolução: um GRASP multiobjetivo, denominado GMO_PSAP, um VNS multiobjetivo, denominado

MOVNS_PSAP, um GRASP multiobjetivo utilizando o VNS como busca local, denominado GMOVNS_PSAP, um VNS multiobjetivo com intensificação, denominado MOVNS_I_PSAP e um ILS multiobjetivo, denominado PILS_PSAP. Tais métodos são descritos no capítulo a seguir.

Capítulo 5

5. ALGORITMOS DE OTIMIZAÇÃO MULTIOBJETIVO PROPOSTOS PARA O PSAPRRP

Uma revisão geral sobre métodos metaheurísticos multiobjetivos foi publicada por Jones *et al.* (2002). Nela, os autores relatam que em 70% dos artigos pesquisados são utilizados AGs como a metaheurística primária, em 24% o *SA* e em 6% a BT. Jones *et al.* (2002) comentam, também, que até aquela época, não haviam trabalhos relevantes que utilizavam outras técnicas metaheurísticas, tais como o GRASP, o VNS e o ILS, aplicadas a problemas de OM. Em vista dessa lacuna, são propostas no presente trabalho aplicações dos algoritmos *Multi-objective* GRASP (GMO), *Multi-objective* VNS (MOVNS) e Pareto ILS (PILS) para a resolução do PSAPRRP formulado como um problema multiobjetivo.

Nas seções seguintes são apresentadas a formulação multiobjetivo utilizada para o PSAPRRP e a descrição dos métodos metaheurísticos propostos para a resolução do problema.

5.1. Formulação Multiobjetivo para o PSAPRRP

Na formulação multiobjetivo utilizada para o PSAPRRP foram considerados dois objetivos conflitantes, tempo e custo. Ou seja, na utilização dos algoritmos busca-se soluções que minimizem simultaneamente o tempo máximo de conclusão das atividades (*makespan*) e o somatório dos custos associados à data de início de execução das atividades. Os objetivos foram os mesmos utilizados na seção 4.3.2, isto é:

1) A primeira função-objetivo busca a minimização da data de finalização do projeto.

$$f_1(x) = S_{n+1}$$

2) A segunda função-objetivo tem como meta a minimização do somatório dos custos de execução das atividades relacionados às suas datas de início.

$$f_2(x) = \sum_{i=1}^{n} \frac{c_i}{S_i}$$

Em problemas relacionados ao PSAPRRP, a minimização do *makespan* é o objetivo mais encontrado na literatura. Devido à preocupação com o cumprimento de prazos, os responsáveis por projetos buscam finalizá-los o mais cedo possível. O segundo objetivo representa uma questão muito discutida no gerenciamento de projetos: vale a pena um maior investimento para que as atividades sejam executadas o mais cedo possível. As soluções geradas pelos algoritmos propostos apresentarão diferentes relações entre os objetivos, ajudando o gestor de projetos na tomada de decisões.

Nas seções a seguir são apresentados os cinco algoritmos multiobjetivos propostos para a resolução do PSAPRRP.

5.2. GMO_PSAP

Em Festa e Resende (2009) são descritas as principais aplicações da metaheurística GRASP. Segundo os autores, o método GRASP tem sido utilizado com sucesso na resolução de diversas classes de problemas de otimização combinatória. Entretanto, existem poucas referências de aplicações do GRASP a problemas de OM. Esse fato mostra que a utilização da metaheurística GRASP para a resolução de

problemas de OM é recente. Pelo levantamento bibliográfico feito, ainda há poucas aplicações do GMO, como as de Vianna e Arroyo (2004), Arroyo *et al*. (2008), Ishida *et al*. (2008) e Reynolds e Iglesia (2008).

O *Multi-objective* GRASP (GMO) é um algoritmo de otimização multiobjetivo baseado na metaheurística GRASP, proposta por Feo e Resende (1995). O pseudocódigo da versão do GMO proposta neste trabalho, denominada GMO_PSAP e baseada em Reynolds e Iglesia (2008), é apresentado na Figura 27.

Procedimento GMO_PSAP
Entrada: GMO_{max}, α
Saída: *D**
$D^* \leftarrow \phi$;
Para (Iter = 1 até GMO_{max}) **faça**
  $s \leftarrow$ *Construção_GMO_PSAP*(*s*, α, *D**);
  $s \leftarrow$ *BuscaLocal_GMO_PSAP*(*s*, *D**);
Fim_para;
Retorna *D**;

Figura 27: Pseudocódigo do GMO_PSAP

Como no método proposto por Feo e Resende (1995), o GMO_PSAP é composto por duas fases: uma de construção e uma de busca local. Em cada uma das GMO_{max} iterações do algoritmo apresentado na Figura 27, na fase de construção é gerada uma solução *s* através de uma adaptação do MSGS descrito na seção 3.4.1-*B*. Esta adaptação consiste na inserção de uma taxa de aleatoriedade (α) ao método, sendo a função adaptativa gulosa, característica do GRASP mono-objetivo, baseada em critérios de prioridade. O pseudocódigo do procedimento *Construção_GMO_PSAP* é apresentado na Figura 28.

Procedimento *Construção_GMO_PSAP*
Entrada: *s*, α, *D**
Saída: *s*
$s \leftarrow \phi$;
Inicialize a lista *LC* de candidatos;
Determine, aleatoriamente, o valor de $\alpha \in [0, 1]$;
Determine, aleatoriamente, um critério de prioridade;
Enquanto ($LC \neq \phi$) **faça**
  Determine a *LRC*, a partir de *LC*, de acordo com α e com o critério de prioridade selecionado;
  Selecione, aleatoriamente, um elemento $t \in LCR$;
  $s \leftarrow s \cup \{t\}$;
  Atualize *LC*;
Fim_enquanto;
$D^* \leftarrow$ soluções não-dominadas de $D^* \cup \{s\}$;
Retorna *s*;

Figura 28: Pseudocódigo da fase construtiva do GMO_PSAP

No procedimento descrito na Figura 28, a construção de uma solução *s* inicia-se com a geração de uma lista *LC* de atividades candidatas a serem incluídas no sequenciamento. A *LC* é determinada pelas atividades *t* disponíveis para execução no instante de tempo considerado e cujas predecessoras já foram sequenciadas. A partir de *LC*, o valor de $\alpha \in [0, 1]$ definirá a lista de candidatos restrita (*LCR*), onde a função

adaptativa gulosa é determinada pelo critério de prioridade selecionado, isto é, a atividade que possuir maior prioridade será a que trará o maior benefício ao ser incluída no sequenciamento. Definida a *LCR*, uma atividade $t \in LCR$ é selecionada aleatoriamente e inserida em *s*, sendo, então, a *LC* atualizada. Para a determinação da função adaptativa gulosa das atividades foram utilizados três diferentes tipos de regras de prioridade: menor tempo de execução, maior número de atividades sucessoras e menor custo.

Por fim, a solução *s* gerada é avaliada para fazer parte ou não do conjunto de soluções não-dominadas *D**. Objetivando a geração de diferentes soluções ao longo das fronteiras de Pareto, o valor de $\alpha \in [0, 1]$ e o critério de prioridade a ser utilizado são determinados aleatoriamente a cada chamada do procedimento *Construção_GMO_PSAP*.

Na fase de busca local, a solução *s* gerada pelo procedimento *Construção_GMO_PSAP* é modificada por meio do movimento troca de atividades, apresentado por Thomas e Salhi (1998) e descrito na seção 3.4.2-*A*, de forma que novas soluções sejam geradas. O pseudocódigo do procedimento *BuscaLocal_GMO_PSAP* é apresentado na Figura 29.

Procedimento *BuscaLocal_GMO_PSAP*
Entrada: *s*, *D**
Saída: *D**
Determine, aleatoriamente, uma solução vizinha $s' \in N(s)$;
Para (*cada vizinho* $s'' \in N(s')$) **faça**
  $D^* \leftarrow$ soluções não-dominadas de $D^* \cup \{s''\}$;
Fim_para;
Retorna *D**;

Figura 29: Pseudocódigo da fase de busca local do GMO_PSAP

O procedimento descrito na Figura 29 inicia-se com a determinação aleatória de uma solução $s' \in N(s)$. Daí, o conjunto *D**, a ser retornado pelo GMO_PSAP, é atualizado através da avaliação de todas as soluções vizinhas $s'' \in N(s')$. A avaliação das soluções para definir o conjunto *D** é feita utilizando o critério de dominância de Pareto descrito na seção 4.1.1.

5.3. MOVNS_PSAP

A metaheurística VNS tem sido aplicada eficientemente na resolução de diversos tipos de problemas de otimização combinatória mono-objetivo, mas poucas aplicações são encontradas na literatura para problemas de OM. Uma das primeiras aplicações do VNS multiobjetivo foi proposta por Geiger (2008). O autor propôs um algoritmo VNS multiobjetivo para a resolução do problema de *flowshop* biobjetivo. Algumas aplicações mais recentes podem ser encontradas em Liang *et al.* (2009), Liang e Lo (2010), Paiva *et al.* (2010) e Ottoni *et al.* (2011).

O *Multi-objective Variable Neighborhood Search* (MOVNS) é um algoritmo de otimização multiobjetivo proposto por Geiger (2008), cuja estrutura se baseia na metaheurística VNS, proposta por Mladenovic e Hansen (1997). Na Figura 30 é apresentada a versão proposta do MOVNS, denominada MOVNS_PSAP e baseada em Ottoni *et al.* (2011), para o problema abordado neste trabalho.

Procedimento MOVNS_PSAP

Entrada: *r*, critério de parada
Saída: *D**
Construa 3 soluções (sequenciamentos) $\{s_1, s_2, s_3\}$ utilizando diferentes regras de prioridade;
$D^* \leftarrow$ soluções não-dominadas de $\{s_1, s_2, s_3\}$;
Enquanto (Critério de Parada = *Falso*) **faça**
  Escolha, aleatoriamente, uma solução não visitada $s \in D^*$;
  Mark(*s*) $\leftarrow$ *True*;
  Determine, aleatoriamente, uma estrutura de vizinhança $N_i \in \{N_1, ..., N_r\}$;
  Determine, aleatoriamente, uma solução $s' \in N_i(s)$;
  Para (*cada vizinho* $s'' \in N_i(s')$) **faça**
    $D^* \leftarrow$ soluções não-dominadas de $D^* \cup \{s''\}$;
  Fim_para;
  Se (*todas as soluções de D* estão marcadas como visitadas*) e**ntão**
    Remova a marcação de todas as soluções de *D**;
  Fim_se;
Fim_enquanto;
Retorna *D**;

Figura 30: Pseudocódigo do MOVNS_PSAP

O procedimento descrito na Figura 30 inicia com a geração de três soluções (s_1, s_2 e s_3) através do MSGS descrito na seção 3.4.1-*B*, sendo cada uma delas obtida utilizando um critério de prioridade diferente. Os critérios de prioridade utilizados foram os mesmos do GMO_PSAP. Estas soluções são, então, avaliadas entre si e, as não-dominadas são armazenadas no conjunto *D**. Da mesma forma proposta por Geiger (2008), a cada iteração da busca local seleciona-se aleatoriamente uma solução não visitada $s \in D^*$, sendo esta marcada como visitada (*Mark*(*s*) $\leftarrow$ *True*), e uma estrutura de vizinhança $N_i \in \{N_1, ..., N_r\}$. No MOVNS_PSAP foram utilizadas duas estruturas de vizinhança ($r = 2$): troca de atividades e inserção de atividade do tipo 1, descritas na seção 3.4.2-*A*. Em seguida, é determinada aleatoriamente uma solução $s' \in N_i(s)$ e, o conjunto *D** é atualizado através da avaliação de todas as soluções vizinhas $s'' \in N_i(s')$. Por fim, é verificado se todas as soluções pertencentes a *D** estão marcadas como visitadas e, em caso afirmativo, a marcação é removida de todas. Este procedimento é repetido até que o critério de parada seja satisfeito, quando é retornado o conjunto *D**.

Da mesma forma que no GMO_PSAP, para avaliar as soluções geradas e, determinar as não-dominadas que farão parte de *D**, é utilizado o critério de dominância de Pareto descrito na seção 4.1.1.

5.4. GMOVNS_PSAP

O GMOVNS_PSAP proposto neste trabalho é um algoritmo híbrido que combina características do GMO_PSAP com características do MOVNS_PSAP, descritos nas seções 5.2 e 5.3, respectivamente. O algoritmo GMOVNS_PSAP segue a estrutura descrita na Figura 27, mas apresenta modificações nas fases de construção e de busca local. O pseudocódigo do GMOVNS_PSAP é apresentado na Figura 31.

Procedimento GMOVNS_PSAP
Entrada: $GMOVNS_{max}$, α, β
Saída: D^*
$D^* \leftarrow \phi$;
Para (Iter = 1 até $GMOVNS_{max}$) **faça**
  $D_1 \leftarrow$ *Construção_GMOVNS_PSAP*(α, β, D_1);
  $D_1 \leftarrow$ *BuscaLocal_GMOVNS_PSAP*(D_1, D^*, r);
Fim_para;
Retorna D^*;

Figura 31: Pseudocódigo do GMOVNS_PSAP

Como pode ser visto na Figura 31, da mesma forma que o GMO_PAP, o GOMVNS_PSAP é composto por duas fases: construção e busca local. A Figura 32 descreve o procedimento *Construção_GMOVNS_PSAP*, no qual um conjunto de soluções não-dominadas D_1 é gerado a cada iteração do algoritmo.

Procedimento *Construção_GMOVNS_PSAP*
Entrada: α, β
Saída: D_1
$D_1 \leftarrow \phi$;
Para (Iter = 1 até β) **faça**
  $s \leftarrow \phi$;
  Inicialize a lista *LC* de candidatos;
  Determine, aleatoriamente, o valor de $\alpha \in [0, 1]$;
  Determine, aleatoriamente, um critério de prioridade;
  Enquanto ($LC \neq \phi$) **faça**
    Determine a *LRC*, a partir de *LC*, de acordo com α e com o critério de prioridade selecionado;
    Selecione, aleatoriamente, um elemento $t \in LCR$;
    $s \leftarrow s \cup \{t\}$;
    Atualize *LC*;
  Fim_enquanto;
  $D_1 \leftarrow$ soluções não-dominadas de $D_1 \cup \{s\}$;
Fim_para;
Retorna D_1;

Figura 32: Pseudocódigo da fase construtiva do GMOVNS_PSAP

Em cada uma das $GMOVNS_{max}$ iterações do método, na fase de construção descrita na Figura 32 são geradas β soluções, que são avaliadas e, as não-dominadas armazenadas no conjunto D_1. Todas as soluções desta fase são geradas através da mesma adaptação do MSGS utilizado no GMO_PSAP e, para que diferentes soluções possam ser geradas, na construção de cada uma são determinados aleatoriamente um valor para $\alpha \in [0, 1]$ e um critério de prioridade. Os critérios de prioridade utilizados também são os mesmos do GMO_PSAP.

Na fase de busca local do algoritmo foi proposta a utilização da metaheurística VNS, com as mesmas estruturas de vizinhança utilizadas no MOVNS_PSAP (r = 2). O VNS é capaz de explorar melhor o espaço de soluções factíveis do problema devido à sua troca sistemática de estruturas de vizinhança. Com isso, buscou-se melhorar a qualidade do conjunto D^*. O pseudocódigo do procedimento *BuscaLocal_GMOVNS_PSAP* é apresentado na Figura 33.

Procedimento *BuscaLocal_GMOVNS_PSAP*

Entrada: D_1, D^*, r, critério de parada
Saída: D^*
Enquanto (Critério de Parada = *Falso*) **faça**
  Escolha, aleatoriamente, uma solução não visitada $s \in D_1$;
  Mark(s) $\leftarrow$ *True*;
  Determine, aleatoriamente, uma estrutura de vizinhança $N_i \in \{N_1, ..., N_r\}$;
  Determine, aleatoriamente, uma solução $s' \in N_i(s)$;
  Para (*cada vizinho* $s'' \in N_i(s')$) **faça**
    $D_1 \leftarrow$ soluções não-dominadas de $D_1 \cup \{s''\}$;
  Fim_para;
  Se (*todas as soluções de* D_1 *estão marcadas como visitadas*) e**ntão**
    Remova a marcação de todas as soluções de D_1;
  Fim_se;
Fim_enquanto;
$D^* \leftarrow$ soluções não-dominadas de $D^* \cup D_1$;
Retorna D^*;

Figura 33: Pseudocódigo da fase de busca local do GMOVNS_PSAP

A cada iteração do GMOVNS_PSAP, a solução a ser explorada é determinada aleatoriamente dentre as não visitadas pertencentes ao conjunto D_1 gerado pela fase de construção. Em seguida, uma estrutura de vizinhança $N_i \in \{N_1, ..., N_r\}$ e uma solução $s' \in N_i(s)$ são escolhidas aleatoriamente. O conjunto D_1 é, então, atualizado através da avaliação de todas as soluções vizinhas $s'' \in N_i(s')$. Finalmente, é verificado se todas as soluções pertencentes a D_1 estão marcadas como visitadas e, em caso afirmativo, a marcação é removida de todas. Este procedimento é repetido até que o critério de parada seja satisfeito. A partir de D_1, a cada iteração o conjunto D^* é atualizado com a avaliação de todas as soluções de $D^* \cup D_1$. Para avaliar as soluções e determinar as não-dominadas que farão parte de D^*, a ser retornado pelo GMOVNS_PSAP, é utilizado o critério de dominância de Pareto descrito na seção 4.1.1.

5.5. MOVNS_I_PSAP

São encontradas na literatura duas variantes do algoritmo MOVNS, uma proposta por Ottoni *et al*. (2011) e outra proposta por Arroyo *et al*. (2011). Estas variantes consistem em adicionar um procedimento de intensificação ao método. A intensificação da busca ao redor da melhor solução encontrada é obtida, por exemplo, pela aplicação de pequenas perturbações sobre ela.

O MOVNS com intensificação, denominado MOVNS_I_PSAP, porposto neste trabalho é baseado na variante proposta por Ottoni *et al*. (2011) e descrito na Figura 34.

Procedimento MOVNS_I_PSAP

Entrada: *r*, critério de parada
Saída: *D**
Construir 3 soluções (sequenciamentos) $\{s_1, s_2, s_3\}$ utilizando diferentes regras de prioridade;
$D^* \leftarrow$ soluções não-dominadas de $\{s_1, s_2, s_3\}$;
Enquanto (Critério de Parada = *Falso*) **faça**
  Escolha, aleatoriamente, uma solução não visitada $s \in D^*$;
  Mark(*s*) $\leftarrow$ *True*;
  Determine, aleatoriamente, uma estrutura de vizinhança $N_i \in \{N_{1, \ldots,} N_r\}$;
  Determine, aleatoriamente, uma solução $s' \in N_i(s)$;
  Para (*cada vizinho* $s'' \in N_i(s')$) **faça**
    $D^* \leftarrow$ soluções não-dominadas de $D^* \cup \{s''\}$;
  Fim_para;
  Se (*todas as soluções de D* estão marcadas como visitadas*) **então**
    Remover as marcações das soluções;
  Fim_se;
  Selecione aleatoriamente uma solução $s \in D^*$;
  $D_1 \leftarrow$ ***INTENSIFICAÇÃO***(*s*, *d*);
  $D^* \leftarrow$ soluções não-dominadas de $D^* \cup D_1$;
Fim_enquanto;
Retorna *D**;

Figura 34: Pseudocódigo do MOVNS_I_PSAP

Conforme visto na Figura 34, o MOVNS_I_PSAP consiste no acréscimo de um procedimento de intensificação ao MOVNS_PSAP descrito na seção 5.3. De acordo como proposto por Ottoni *et al.* (2011), o procedimento de intensificação utilizado é composto por duas etapas: uma de *destruição* e uma de *reconstrução*, conforme apresentado na Figura 35.

Procedimento *INTENSIFICAÇÃO*

Entrada: s, d
Saída: D_1
$s_r \leftarrow \phi$;
$s_p \leftarrow s$;
Selecione aleatoriamente pesos w_1 e $w_2 \in [0, 1]$ tal que $w_1 + w_2 = 1$;
Para (i = 1 até d) **faça**
  Seja $s_p(j)$ a j-ésima atividade de s_p escolhida aleatoriamente;
  Remova $s_p(j)$ de s_p;
  Adicione $s_p(j)$ à s_r;
Fim_para;
Para (i = 1 até (d – 1)) **faça**
  $f_p^* \leftarrow \infty$;
  Para (j = 1 até (n - d + i)) **faça**
    $s' \leftarrow$ resultado da inserção da i-ésima atividade de s_r na j-ésima posição de s_p;
    Se ($f(s') < f_p^*$) **então**
      $s_p^* \leftarrow s'$;
      $f_p^* \leftarrow f(s')$;
    Fim_se;
  Fim_para;
  $s_p \leftarrow s_p^*$;
Fim_para;
Para (j = 1 até n) **faça**
  $s' \leftarrow$ resultado da inserção da última atividade de s_r na j-ésima posição de s_p;
  $D_1 \leftarrow$ soluções não-dominadas de $D_1 \cup \{s'\}$;
Fim_Para;
Retorna D_1;

Figura 35: Pseudocódigo do procedimento de Intensificação do MOVNS_I_PSAP

O procedimento de intensificação inicia-se com a etapa de *destruição*, na qual d atividades são removidas de uma solução $s \in D^*$ escolhida aleatoriamente. Esta estratégia resulta na geração de uma solução parcial s_p, composta por ($n – d$) atividades, e, de um conjunto s_r com as d atividades removidas de s. Em seguida, a solução s é reconstruída inserindo-se (d - 1) atividades em s_p. Para isso, uma atividade pertencente a s_r é inserida em todas as posições possíveis de s_p, sendo escolhida a posição que fornecer a melhor solução parcial. A avaliação das soluções parciais é feita através de uma função ponderada dada pela equação $f = w_1 f_1 + w_2 f_2$, onde w_1 e w_2 são pesos associados às funções-objetivo e $w_1 + w_2 = 1$. Este procedimento é realizado até que (d – 1) atividades de s_r sejam inseridas em s_p. Por fim, a solução parcial s_p sofre a inserção da última atividade de s_r em todas as suas possíveis posições. Todas as soluções geradas por esse último processo de inserção são avaliadas e, as não-dominadas armazenadas em D_1. Após o procedimento de intensificação, o conjunto D^* é atualizado através da avaliação de todas as soluções de $D^* \cup D_1$.

Para avaliar as soluções e determinar as não-dominadas que farão parte de D^*, a ser retornado pelo MOVNS_I_PSAP, é utilizado o critério de dominância de Pareto descrito na seção 4.1.1.

5.6. PILS_PSAP

O *Pareto Iterated Local Search* (PILS) é um algoritmo de otimização multiobjetivo proposto por Geiger (2006), sendo baseado na metaheurística ILS (Lourenço *et al*., 2002). Apresenta-se na Figura 36 o pseudocódigo do PILS proposto para o PSAPRRP, baseado em Geiger (2006) e denominado PILS_PSAP.

Procedimento PILS_PSAP
Entrada: *r*, critério de parada
Saída: *D**
Determine um conjunto de soluções não-dominadas inicial *D**;
Selecione aleatoriamente uma solução $s \in D^*$;
Enquanto (Critério de Parada = *Falso*) **faça**
  $i \leftarrow 1$;
  Enquanto ($i < r \wedge$ Critério de Parada = *Falso*) **faça**
    Para (*cada vizinho* $s' \in N_i(s)$) **faça**
      $D^* \leftarrow$ soluções não-dominadas de $D^* \cup \{s'\}$;
    Fim_para;
    Se ($\exists\ s' \in N_i(s) |\ s'$ domina s) **então**
      $s \leftarrow s'$**;**
      Reordene as estruturas de vizinhança N_1, ..., N_r, em uma ordem aleatória;
      $i \leftarrow 1$;
    senão i ++;
    Fim_se;
  Fim_enquanto;
  Mark(*s*) ← *True*;
  Se ($\exists\ s' \in D^* |\ s'$ ainda não foi visitada) **então**
    $s \leftarrow s'$**;**
  senão
    Selecione aleatoriamente uma solução $s' \in D^*$;
    $s'' \leftarrow$ ***PERTURBACAO***(s');
    $s \leftarrow s''$;
  Fim_se;
Fim_enquanto;
Retorna *D**;

Figura 36: Pseudocódigo do PILS_PSAP

O PILS_PSAP inicia-se com a geração de um conjunto de soluções não-dominadas inicial *D**, utilizando o MSGS, descrito na seção 3.4.1-*B*, e os mesmos critérios de prioridade usados no GMO_PSAP. Em seguida, uma solução $s \in D^*$ é selecionada aleatoriamente, passando a ser a solução corrente e, toda a sua vizinhança é explorada. As estruturas de vizinhança utilizadas são as mesmas do MOVNS_PSAP (r = 2), descritas na seção 3.4.2-*A*. Caso alguma solução vizinha $s' \in N_i(s)$ domine a solução corrente *s*, então *s*´ passa a ser a nova solução corrente, as estruturas de vizinhança são reordenadas aleatoriamente e o procedimento retorna à primeira estrutura de vizinhança da nova ordem gerada. Este procedimento é repetido até que não haja mais soluções não visitadas em *D**, ou seja, quando o algoritmo chegar a um ótimo local em relação à vizinhança explorada. Ocorrido isso, seleciona-se aleatoriamente uma solução $s' \in D^*$, na qual aplica-se uma perturbação. O objetivo ao perturbar uma solução é explorar outros ótimos locais. A perturbação aqui utilizada é a proposta originalmente

por Geiger (2006) e funciona da seguinte forma: após a seleção de uma solução $s' \in D^*$, determina-se aleatoriamente uma posição $j \leq n-4$ e quatro atividades consecutivas de s' nas posições j, $j + 1$, $j + 2$ e $j + 3$. Uma solução s'' é, então, gerada aplicando-se o movimento de troca das atividades das posições j e $j + 3$, assim como das atividades das posições $j + 1$ e $j + 2$. Dessa forma, as atividades anteriores à atividade da posição j e as após a atividade da posição $j + 3$ continuam nas mesmas posições após a aplicação da perturbação. Em seguida, a solução s'' passa a ser a solução corrente e explora-se toda a sua vizinhança. Caso todas as soluções vizinhas da solução gerada a partir da perturbação sejam dominadas por alguma solução pertencente a D^*, repete-se o procedimento de perturbação. Este procedimento é repetido até que o critério de parada seja satisfeito.

Para avaliar as soluções e determinar as não-dominadas que farão parte de D^*, a ser retornado PILS_PSAP, é utilizado o critério de dominância de Pareto descrito na seção 4.1.1.

Para avaliar a eficiência dos algoritmos implementados, os resultados obtidos através da utilização de adaptações de instâncias encontradas na literatura foram comparados utilizando quatro métricas de avaliação de desempenho: medidas de distância, diferença de hipervolume, *epsilon* e taxa de erro. Foram realizados, também, experimentos estatísticos com o intuito de veficar se há diferença significativa entre os algoritmos com relação às métricas.

As instâncias e as métricas de avaliação de desempenho são descritas no capítulo a seguir. São apresentados, também, os valores obtidos pelos algoritmos propostos para as métricas, bem como os resultados dos experimentos estatísticos.

Capítulo 6

6. RESULTADOS

Os algoritmos propostos neste trabalho foram implementados computacionalmente utilizando a linguagem C++ e executados em um computador AMD *Turion II Dual-Core* 2.20GHz e 4GB de RAM, sob sistema operacional *Windows* 7 *Home Premium* 64 Bits.

Visando avaliar os algoritmos sob as mesmas condições, o critério de parada adotado para todos foi o mesmo, baseado no número de soluções geradas. Na literatura, esse critério é muito utilizado em algoritmos mono e multiobjetivos para o PSAP, como pode ser visto em Deiranlou e Jolai (2009), Ballestín e Blanco (2011) e Agarwal *et al.* (2011), dentre outros. Deiranlou e Jolai (2009) adotaram os valores de 1000, 5000 e 50000 como critério de parada e apresentaram uma comparação entre os resultados obtidos com cada um dos três valores. Da mesma forma, Ballestín e Blanco (2011) compararam os resultados obtidos com os valores de 5000, 10000, 25000 e 50000, e Agarwal *et al.* (2011) com os valores de 1000 e 5000. Diversos valores podem ser encontrados na literatura, mas adotou-se aqui o número máximo de soluções igual a 5000 como critério de parada para os algoritmos.

Na execução do GMO_PSAP foi definido empiricamente o valor 100 para o parâmetro GMO_{max} e, na execução do GMOVNS_PSAP foram, também, definidos empiricamente os valores 10 para β e 100 para $GMOVNS_{max}$. Como em Ottoni *et al.* (2011), o valor adotado para *d*, na execução do MOVNS_I_PSAP, foi 4 (quatro).

Para a realização dos testes computacionais foram utilizadas adaptações de instâncias encontradas na literatura. Tais instâncias são descritas na seção seguinte.

6.1. Instâncias Utilizadas

O estudo do PSAPRRP multiobjetivo envolve algumas dificuldades, principalmente relacionadas à disponibilidade de instâncias na literatura. Vários problemas mono-objetivos podem ser encontrados, como é o caso do *Project Scheduling Problem Library - PSPLib*, desenvolvido por Kolisch e Sprecher (1996), porém nada foi encontrado relacionado a problemas multiobjetivos.

Em vista disso, e devido à dificuldade de obtenção de dados reais, pois as empresas de construção metálica não possuem ou não disponibilizam os dados necessários, para testar os algoritmos foram utilizadas 160 instâncias diferentes retiradas do *PSPLib*, disponíveis no endereço http://129.187.106.231/psplib, contendo o número de atividades $n \in \{30, 60, 90, 120\}$. Para cada valor de *n* foram utilizadas 40 instâncias, nas quais a execução das atividades demandam 4 tipos diferentes de recursos recuperáveis. Como as instâncias utilizadas são para o PSAPRRP mono-objetivo e não apresentam custos (c_i) associados às atividades, esses valores foram gerados de forma aleatória e com distribuição uniforme no intervalo [1, 500].

Devido ao fato de os algoritmos propostos utilizarem escolhas aleatórias, para cada instância foram feitas trinta execuções, com trinta sementes diferentes geradas randomicamente. A partir das soluções obtidas nas trinta execuções de cada algoritmo, determinou-se os conjuntos de soluções não-dominadas para cada instância. O valor trinta foi escolhido por ser estatisticamente significativo.

Para comparar a eficiência dos algoritmos propostos foram utilizadas quatro métricas de avaliação de desempenho, descritas na seção a seguir.

6.2. Métricas de Avaliação de Desempenho

A comparação entre conjuntos de soluções não-dominadas obtidos por diferentes algoritmos de otimização multiobjetivo não é uma tarefa trivial. Não existe uma medida simples e natural que seja capaz de capturar informações sobre a qualidade de um conjunto aproximado em relação ao conjunto Pareto-ótimo (Arroyo, 2002). Entretanto, diversas métricas de avaliação de desempenho de algoritmos multiobjetivos podem ser encontradas na literatura, como em Hansen e Jaszkiewiez (1998), Czyzak e Jaszkiewiez (1998), Veldhuizen (1999), Zitzler *et al.* (2000), Deb e Jain (2002) e Fonseca *et al.* (2005).

Neste trabalho, para avaliar a qualidade das soluções não-dominadas obtidas pelos algoritmos propostos, foram utilizadas quatro métricas de avaliação de desempenho: medidas de distância, diferença de hipervolume, *epsilon* e taxa de erro.

Para cada instância, seja D_i o conjunto de soluções não-dominadas encontrado pelo algoritmo i, para $i = 1, 2, ..., h$, sendo h o número de algoritmos avaliados, neste caso $h = 5$. A partir destes conjuntos, determina-se o conjunto de soluções não-dominadas de referência denotado por *Ref*, onde $Ref = \{s \in D_1 \cup D_2 \cup ... \cup D_h | \, s$ é uma solução não-dominada$\}$. O conjunto *Ref* é o conjunto de soluções mais próximo da fronteira Pareto-ótima conhecido. Sendo assim, a qualidade dos conjuntos de soluções não-dominadas obtidos pelos algoritmos propostos é mensurada com base nas soluções do conjunto *Ref*, utilizando as métricas apresentadas a seguir.

6.2.1. Medidas de Distância

A métrica medidas de distância, proposta por Czyzak e Jaszkiewicz (1998) e Ulungu *et al*. (1998), mede a proximidade das soluções do conjunto D_i em relação às soluções do conjunto *Ref*. Tal métrica mede, também, a distribuição das soluções dentro do conjunto D_i. Segundo os autores, D_i é uma boa aproximação de *Ref* se D_i fornece informações importantes de todas as regiões do conjunto *Ref*, em outras palavras, se para cada solução $x \in Ref$ existe uma solução $x' \in D_i$ tal que a distância entre x' e x é pequena. Ou seja, quanto mais próximas de zero forem as medidas de distância, melhor será a qualidade das soluções encontradas pelo algoritmo. As fórmulas utilizadas para calcular as distâncias média (D_{med}) e máxima (D_{max}) das soluções de D_i em relação ao conjunto *Ref* são:

$$D_{med}(D_i) = \frac{1}{|Ref|} \sum_{s \in Ref} \min_{s' \in D_i} d(s, s')$$

$$D_{\max}(D_i) = \max_{s \in Ref} \{ \min_{s' \in D_i} d(s, s') \}$$

em que $|Ref|$ é a cardinalidade do conjunto *Ref* e:

$$d(s, s') = \max \left\{ \frac{(f_1(s) - f_1(s'))}{\Delta_1}, \frac{(f_2(s) - f_2(s'))}{\Delta_2} \right\}$$

Δ_j é a diferença entre o maior e o menor valor da função-objetivo f_j, considerando as soluções do conjunto *Ref*.

Note que D_{med} é a média das distâncias de uma solução $x \in Ref$ em relação a solução mais próxima de D_i, enquanto D_{max} fornece o máximo das distancias mínimas de uma solução $x \in Ref$ em relação a alguma solução de D_i. As distâncias D_{med} e D_{max} são amplamente utilizadas como medidas de avaliação de desempenho de algoritmos multiobjetivos como, por exemplo, em Czyzak e Jaszkiewicz (1998), Viana e Sousa (2000) e Ottoni *et al*. (2011).

6.2.2. Diferença de Hipervolume

O hipervolume, denotado por $H(D_i)$ e proposto por Zitzler e Thiele (1998), mede a área coberta ou dominada pelo conjunto de soluções não-dominadas D_i obtido por um algoritmo multiobjetivo. Para o caso da minimização de dois objetivos, um ponto de referência (x, y) é utilizado para limitar $H(D_i)$, sendo x e y limitantes superiores para f_1 e f_2, respectivamente. Uma maior área de dominância indica que as soluções obtidas pelo algoritmo geram uma boa cobertura da fronteira Pareto-ótima. Sendo assim, o valor da métrica diferença de hipervolume, denotado por $H^-(D_i)$, é calculado da seguinte maneira:

$$H^-(D_i) = H(Ref) - H(D_i)$$

Sendo $H(Ref) > H(D_i)$, quanto menor o valor de $H^-(D_i)$, melhor será a qualidade do conjunto D_i. Na Figura 37, ilustram-se as áreas cobertas por dois conjuntos de soluções, D_1 e D_2.

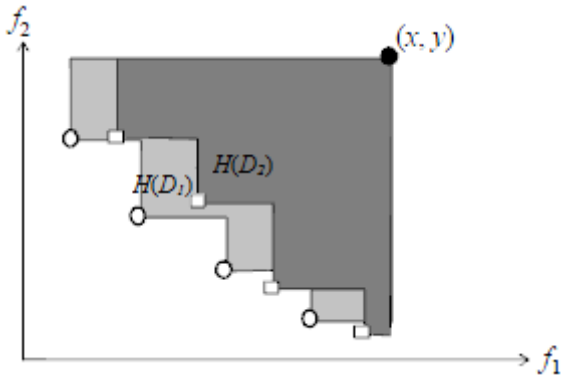

Figura 37: Exemplo de áreas cobertas por dois conjuntos de soluções

Conforme observa-se na Figura 37, $H(D_1) > H(D_2)$, logo $H^-(D_1) < H^-(D_2)$, indicando que as soluções do conjunto D_1 são "melhores" com relação às soluções do conjunto D_2.

6.2.3. *Epsilon*

Dados dois conjuntos de soluções não-dominadas D_1 e D_2, gerados respectivamente por dois algoritmos diferentes, e $z^a = (z_1^a, ..., z_r^a)$ e $z^b = (z_1^b, ..., z_r^b)$ duas soluções pertencentes aos conjuntos D_1 e D_2, respectivamente, a métrica *epsilon*, denotada por $I_\varepsilon(D_1, D_2)$ e proposta por Fonseca *et al.* (2005), mede a distância normalizada máxima do conjunto D_1 com relação ao conjunto D_2. Tal métrica é calculada pela formula a seguir:

$$I_\varepsilon(D_1, D_2) = \max_{z^b \in D_2} \{ \min_{z^a \in D_1} \{ \max_{1 \le i \le r} \frac{z_i^a}{z_i^b} \} \}$$

Sendo assim, a qualidade de um conjunto de soluções não-dominadas obtido por um algoritmo para uma determinada instância é avaliada com relação ao conjunto *Ref*, ou seja, $I_\varepsilon^1(D_i) = I_\varepsilon(D_i, Ref)$. Como $I_\varepsilon(D_i, Ref)$ mede a distância normalizada máxima do conjunto D_i com relação ao conjunto *Ref*, então um valor próximo a um de $I_\varepsilon^1(D_i)$ indica uma boa qualidade do conjunto D_i.

Para utilizar a métrica *epsilon* para avaliar um conjunto de soluções D_i, os valores das funções-objetivo devem ser normalizados de acordo com a seguinte equação:

$$f_i^*(s) = \left(\frac{f_i(s) - f_i^{\min}}{f_i^{\max} - f_i^{\min}} \right)$$

onde $f_i^{\min}$ e $f_i^{\max}$ são, respectivamente, o menor e o maior valor encontrado para o *i*-ésimo objetivo considerando as soluções pertencentes ao conjunto *Ref*. Desta forma, os valores da função objetivo $f_i^*(s)$ calculados pela equação estão no intervalo [0, 1].

6.2.4. Taxa de Erro

A métrica taxa de erro indica a porcentagem de soluções pertencentes ao conjunto D_i que não faz parte do conjunto *Ref*. A métrica, baseada em Veldhuizen (1999) e denotada por TE_i, é calculada pela equação a seguir:

$$TE_i = \frac{|D_i| - |Ref \cap D_i|}{|D_i|} \times 100$$

onde $|D_i|$ corresponde à cardinalidade do conjunto D_i e $|Ref \cap D_i|$ ao número de soluções de referência provenientes do conjunto D_i.

Segundo Coello e Lamont (2004), $TE_i = 0$ indica que todas as soluções pertencentes a D_i fazem parte de *Ref* e, por outro lado, $TE_i = 100$ indica que nenhuma solução de D_i faz parte de *Ref*. Portanto, quanto mais próximo de zero for o valor de TE_i, melhor o desempenho do algoritmo.

A seguir são apresentados os resultados obtidos pelos algoritmos propostos para as métricas de avaliação de desempenho utilizadas.

6.3. Resultados Obtidos

Na Tabela 8 são descritos os tempos computacionais médios, em segundos, gastos por cada algoritmo para a obtenção dos conjuntos de soluções não-dominadas.

Tabela 8: Tempos Computacionais Médios Gastos por Cada Algoritmo (em segundos)

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	**0,18**	0,42	0,39	1,05	1,12
60	**1,01**	3,93	3,01	4,12	10,98
90	**2,78**	13,21	11,04	15,79	53,44
120	**7,69**	54,13	36,40	57,70	143,34

Na primeira coluna da Tabela 8 indica-se o conjunto de instâncias agrupadas de acordo com o número *n* de atividades. Nas demais colunas, para cada algoritmo, estão os tempos computacionais médios gastos na obtenção dos conjuntos de soluções não-dominadas para cada conjunto de instâncias.

Como pode ser observado na Tabela 8, todos os algoritmos apresentaram baixo esforço computacional, ou seja, obtiveram os conjuntos de soluções não-dominadas em um tempo computacional aceitável. Para todos os conjuntos de instâncias, o GMO_PSAP foi o que apresentou o menor tempo computacional médio para a obtenção dos conjuntos de soluções não-dominadas.

Nas Tabelas 9 e 10 são apresentados os resultados obtidos pelos algoritmos em relação às medidas de distância. Na Tabela 9 são apresentados os resultados em relação à distância média e na Tabela 10 em relação à distância máxima. Na primeira coluna das

tabelas indica-se o conjunto de instâncias agrupadas de acordo com o número *n* de atividades e, nas demais estão os valores médios das medidas de distância obtidos, por cada algoritmo, para cada conjunto de instâncias.

Tabela 9: Resultados da Métrica Medidas de Distância – Distância Média

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	0,16	0,18	0,14	**0,04**	0,06
60	0,65	0,19	0,19	0,14	**0,06**
90	0,14	0,14	0,15	**0,06**	0,10
120	0,33	0,17	0,26	**0,03**	0,13
Média	0,32	0,17	0,19	**0,07**	0,09

Tabela 10: Resultados da Métrica Medidas de Distância – Distância Máxima

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	0,65	0,49	0,56	**0,12**	0,20
60	1,02	0,63	0,78	0,35	**0,21**
90	0,47	0,41	0,38	**0,15**	0,30
120	0,88	0,44	0,96	**0,14**	0,37
Média	0,76	0,49	0,67	**0,19**	0,27

Pelas Tabelas 9 e 10, verifica-se que o algoritmo MOVNS_I_PSAP é o que produz menores valores médios, isto é, mais próximos de zero, das distâncias média e máxima para a maioria dos conjuntos de instâncias. O MOVNS_I_PSAP só não obteve valores médios menores para o conjunto de instância com $n = 60$, onde o PILS_PSAP apresentou melhores resultados. No geral, o MOVNS_I_PSAP obteve o menor valor médio para as distâncias considerando todos os conjuntos de instâncias. Portanto, o MOVNS_I_PSAP apresentou, na maioria dos casos, um melhor desempenho em relação à métrica medidas de distância, ou seja, soluções melhor distribuídas ao longo da fronteira de Pareto de referência.

Na Tabela 11 são apresentados os valores obtidos pelos algoritmos propostos com relação à diferença de hipervolume.

Tabela 11: Resultados da Métrica Diferença de Hipervolume

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	925,78	616,44	**352,12**	373,22	367,21
60	1769,16	807,31	**535,98**	1433,66	888,32
90	2700,65	2351,28	**1979,39**	2143,45	2132,65
120	5165,89	5913,00	**3715,94**	3876,98	3800,99
Média	2640,37	2422,01	**1645,86**	1956,83	1797,29

Na primeira coluna da Tabela 11 indica-se o conjunto de instâncias agrupadas de acordo com o número *n* de atividades. Nas demais colunas, para cada algoritmo, estão os valores médios da diferença de hipervolume obtidos para cada conjunto de instâncias.

Pela Tabela 11, verifica-se que o algoritmo GMOVNS_PSAP apresentou os menores valores médios, comparados com os dos demais algoritmos, da diferença de hipervolume para todos os conjuntos de instâncias. Isto significa que o algoritmo GMOVNS_PSAP produz uma melhor cobertura para a fronteira Pareto-ótima.

Na Tabela 12 são apresentados os resultados obtidos pelos algoritmos em relação à métrica *epsilon*.

Tabela 12: Resultados da Métrica *Epsilon*

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	1,47	1,93	**1,21**	1,82	1,25
60	1,44	1,75	**1,28**	1,50	1,44
90	1,88	1,85	**1,58**	1,91	1,80
120	1,36	1,70	**1,21**	1,90	1,50
Média	1,54	1,81	**1,30**	1,78	1,50

Na primeira coluna da Tabela 12 indica-se o conjunto de instâncias agrupadas de acordo com o número *n* de atividades e, nas demais estão os valores médios da métrica *epsilon* obtidos, por cada algoritmo, para cada conjunto de instâncias.

Pela Tabela 12, verifica-se que o algoritmo GMOVNS_PSAP é o que produz os menores valores médios para a métrica *epsilon* para todos os conjuntos de instâncias, indicando que as soluções não-dominadas geradas por este algoritmo estão mais próximas do conjunto *Ref*.

Na Tabela 13 são apresentados os valores obtidos pelos algoritmos propostos com relação à taxa de erro.

Tabela 13: Resultados da Métrica Taxa de Erro (%)

n	**Algoritmo**				
	GMO_PSAP	MOVNS_PSAP	GMOVNS_PSAP	MOVNS_I_PSAP	PILS_PSAP
30	77,20	40,72	59,62	43,30	**40,20**
60	86,19	74,65	66,94	72,90	**44,48**
90	89,65	62,11	75,15	63,33	**61,37**
120	91,47	90,52	67,01	57,88	**55,78**
Média	86,13	67,00	67,18	59,35	**50,46**

Na primeira coluna da Tabela 13 indica-se o conjunto de instâncias agrupadas de acordo com o número *n* de atividades. Nas demais colunas, para cada algoritmo, estão os valores médios da taxa de erro obtidos para cada conjunto de instâncias.

Como pode ser observado na Tabela 13, o algoritmo PILS_PSAP apresentou, para todos os conjuntos de instâncias, um menor valor médio para a taxa de erro. Isso significa que, baseado na taxa de erro, o algoritmo PILS_PSAP se mostrou superior aos demais, ou seja, apresentou um número maior de soluções não-dominadas pertencentes ao conjunto de referência.

Nas seções a seguir é apresentada uma análise estatística dos resultados obtidos pelos algoritmos para as métricas de avaliação de desempenho utilizadas. O objetivo de tal análise é verificar se existe diferença significativa entre os algoritmos com relação às métricas.

6.4. Análise Estatística dos Resultados

Os experimentos que seguem têm por objetivo verificar se existe diferença estatisticamente significativa entre os algoritmos propostos no que tange às métricas de avaliação de desempenho utilizadas. Esses experimentos foram conduzidos com o auxílio do pacote computacional Minitab®, em sua versão 16. Vale ressaltar que essa experimentação permite fazer inferências para toda a população de instâncias.

Para a realização dos experimentos, optou-se pelo uso da técnica estatística de Análise de Variância (ANOVA), conforme descrita por Montgomery (2009). A ANOVA testa a hipótese de que as médias (μ) de duas ou mais populações são iguais. Essa técnica avalia a importância de um ou mais fatores, comparando as médias da variável resposta nos diferentes níveis dos fatores. No presente trabalho, os algoritmos

são os fatores avaliados, sendo cada um dos cinco um nível de fator, e os valores obtidos para as métricas são as variáveis resposta.

Na ANOVA, a hipótese nula H_0 (sem variação) declara que todas as médias populacionais (médias dos valores das métricas obtidas pelos algoritmos) são iguais, enquanto a hipótese alternativa H_1 declara que pelo menos uma é diferente. A ANOVA fornece uma estatística de teste que permite aceitar ou rejeitar H_0. A partir do valor desta estatística de teste e de um critério de aceitação/rejeição, é possível concluir, com uma margem tolerada de erro definida *a priori*, qual das hipóteses aceitar. Na aplicação da ANOVA é usual representar essa estatística de teste por *p-value*, assumindo um valor tolerável para erros, denominado nível α de significância do teste. Compara-se, então, o *p-value* com essa tolerância, de tal modo que se: $\alpha \geq p\text{-}value \rightarrow$ Rejeita-se H_0. De acordo com Montgomery (2009), para executar uma ANOVA, deve-se ter:

i. Uma variável resposta contínua e pelo menos um fator com dois ou mais níveis;
ii. Dados populacionais normalmente distribuídos;
iii. Variâncias (σ^2) aproximadamente iguais entre os níveis de fator.

Sem o atendimento dessas premissas, o uso da técnica pode gerar interpretações equivocadas. Logo, antes de iniciar a análise, é crucial a verificação do atendimento dos pontos acima.

Em relação à premissa *i*, esta é atendida em todos os casos devido às métricas utilizadas (variáveis resposta) assumirem valores contínuos e o fator (algoritmos) apresentar cinco níveis (número de algoritmos avaliados). Embora o teste baseie-se na suposição de que os dados devam ser normalmente distribuídos (premissa *ii*), segundo Kulinskaya *et al*. (2003) esta hipótese não é crítica quando os tamanhos das amostras são pelo menos 15 ou 20. Como todas as amostras neste trabalho possuem tamanho igual a 160 (número de instâncias utilizadas) para cada algoritmo, portanto, a normalidade não é crítica. Sendo assim, a premissa de normalidade é verificada para todos os algoritmos com relação a todas as métricas. Portanto, para a utilização da ANOVA é necessário verificar somente a proximidade das variâncias entre os dados dos algoritmos com relação a cada métrica (premissa *iii*).

Entretanto, a ANOVA não nos diz quais pares de algoritmos apresentam diferenças significativas, ou seja, resultam em diferentes médias para a métrica avaliada. Para responder a esta questão, foi utilizado o método da Mínima Diferença Significativa (MDS), também conhecido como método de Fisher (Montgomery, 2009).

Quando a hipótese nula $H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$ é rejeitada na ANOVA, sabe-se que algumas médias dos algoritmos são diferentes, mas não se sabe qual ou quais delas são diferentes. O método MDS compara todos os pares de médias com as hipóteses nulas $H_0: \mu_i = \mu_j$ (para todo par (i, j), com $i \neq j$) e constrói um intervalo de confiança para a diferença dos pares de médias $(\mu_i - \mu_j)$. Se este intervalo contiver o valor zero, pode-se inferir que não existe evidência de que as médias μ_i e μ_j diferem.

Nas seções seguintes são apresentados os resultados obtidos pela ANOVA e pelo método MDS para as quatro métricas de avaliação de desempenho utilizadas.

6.4.1. Medidas de Distância

Para a métrica medidas de distância foram feitos dois testes, um para a distância média e outro para a distância máxima. Os testes dessa métrica são descritos a seguir.

A - Distância Média

Antes de aplicar a ANOVA aos dados da medida de distância média, foi verificada a adequação da premissa *iii*. Para isso, utilizou-se o teste de hipótese que afirma a igualdade das variâncias contra a diferença entre elas, da seguinte maneira:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \quad (1)$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (2)$$

A hipótese (1) representa a hipótese da igualdade das variâncias populacionais da medida de distância média para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre os algoritmos com relação à variância desta métrica. Já a hipótese (2), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,075. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha < p\text{-value}$, aceita-se a hipótese da igualdade das variâncias entre os dados populacionais para os cinco algoritmos, a um nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, a premissa *iii* também é verificada.

Sendo assim, verificadas as três premissas, aplica-se a ANOVA aos dados da métrica em questão, onde testou-se as seguintes hipóteses:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \quad (3)$$

$$H_1 : \mu_i \neq \mu_j \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (4)$$

A hipótese (3) representa a hipótese da igualdade das médias populacionais da medida de distância média para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação à média desta métrica. Já a hipótese (4), conjectura o contrário.

Primeiramente procede-se com o teste (3)-(4), verificando qual das hipóteses deve ser rejeitada através do cálculo do *p-value* para o teste. Com a ajuda do Minitab®, encontrou-se *p-value* = 0,027. Logo, pode-se afirmar, a um nível de 5% de significância ($\alpha = 0{,}05$), que a hipótese (3) deve ser rejeitada, isto é, como $\alpha \geq p\text{-value}$, há evidências estatísticas suficientes para concluir que os valores médios referentes à medida de distância média são diferentes entre os algoritmos. Existe, portanto, uma forte evidência para concluir que os algoritmos têm um efeito na distância média. Como a ANOVA não nos diz quais algoritmos resultam em diferentes médias dessa métrica, essa questão é respondida pelo método MDS. Com a ajuda do Minitab®, pode-se construir todos os intervalos conforme a Figura 38, a seguir.

```
Grouping Information Using Fisher Method

Fisher 95% Individual Confidence Intervals
All Pairwise Comparisons among Levels of Algorithm

Simultaneous confidence level = 71,65%


Algorithm = 1 subtracted from:

Algorithm    Lower  Center  Upper  --+---------+---------+---------+------
2           -44,08  -21,80   0,49      (--------*--------)
3           -42,65  -20,37   1,92       (--------*--------)
4           -54,46  -32,17  -9,88  (--------*--------)
5           -52,68  -30,40  -8,11   (--------*--------)
                                   --+---------+---------+---------+------
                                   -50       -25         0        25


Algorithm = 2 subtracted from:
```

```
21
22  Algorithm    Lower  Center  Upper  --+---------+---------+---------+------
23  3           -20,85    1,43  23,72               (--------*-------)
24  4           -32,66  -10,37  11,91           (--------*--------)
25  5           -30,89   -8,60  13,68            (--------*-------)
26                                  --+---------+---------+---------+------
27                                  -50       -25         0        25
28
29
30  Algorithm = 3 subtracted from:
31
32  Algorithm    Lower  Center  Upper  --+---------+---------+---------+------
33  4           -34,09  -11,80  10,48          (--------*--------)
34  5           -32,32  -10,03  12,25           (--------*--------)
35                                  --+---------+---------+---------+------
36                                  -50       -25         0        25
37
38  Algorithm = 4 subtracted from:
39
40  Algorithm    Lower  Center  Upper  --+---------+---------+---------+------
41  5           -20,51    1,77  24,06               (--------*--------)
42                                  --+---------+---------+---------+------
43                                  -50       -25         0        25
44
45  Algorithm    N   Mean  Grouping
46  1          160  0,32   A
47  3          160  0,17   A B
48  2          160  0,19   A B
49  5          160  0,07     B
50  4          160  0,09     B
51
52  Means that do not share a letter are significantly different.
```

Figura 38: Resultados do teste MDS para Distância Média Fonte: Minitab®

A Figura 38 apresenta os intervalos para as diferenças das médias entre os pares de algoritmos. A partir deles, pode-se visualizar que o algoritmo 1 (GMO_PSAP) apresenta média que difere das médias dos algoritmos 4 (MOVNS_I_PSAP) e 5 (PILS_PSAP), pois os intervalos para $\mu_1 - \mu_4$ (linha 14) e $\mu_1 - \mu_5$ (linha 15) não contém o zero, com 95% de confiança. Logo, pode-se afirmar que há evidências estatísticas de que os valores médios relativos à medida de distância média são diferentes entre os pares de algoritmos: GMO_PSAP × MOVNS_I_PSAP e GMO_PSAP × PILS_PSAP.

O mesmo procedimento foi adotado para a medida de distância máxima, conforme descrito a seguir.

B - Distância Máxima

Da mesma forma como feito para a medida de distância média, antes de aplicar a ANOVA aos dados da medida de distância máxima, é necessário verificar a adequação da premissa *iii*. Para isso, utilizou-se novamente o teste de hipótese que afirma a igualdade das variâncias contra a diferença entre elas, da seguinte maneira:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \quad (5)$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (6)$$

A hipótese (5) representa a hipótese da igualdade das variâncias populacionais da medida de distância máxima para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre os algoritmos com relação à variância desta métrica. Já a hipótese (6), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,055. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha <$ *p-value*, aceita-se a hipótese da igualdade das variâncias entre os dados populacionais para os cinco algoritmos, a um

nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, a premissa *iii* também é verificada.

Sendo assim, verificadas as três premissas, aplica-se a ANOVA aos dados da métrica em questão, onde testou-se as seguintes hipóteses:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \tag{7}$$

$$H_1 : \mu_i \neq \mu_j \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \tag{8}$$

A hipótese (7) representa a hipótese da igualdade das médias populacionais da medida de distância máxima para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação a esta métrica. Já a hipótese (8), conjectura o contrário.

Primeiramente procede-se com o teste (7)-(8), verificando qual das hipóteses deve ser rejeitada através do cálculo do *p-value* para o teste. Com a ajuda do Minitab®, encontrou-se *p-value* = 0,004. Logo, pode-se afirmar, a um nível de 5% de significância ($\alpha = 0{,}05$), que a hipótese (7) deve ser rejeitada, isto é, como $\alpha \geq$ *p-value*, há evidências estatísticas suficientes para concluir que os valores médios referentes à medida de distância máxima são diferentes entre os algoritmos. Existe, portanto, uma forte evidência para concluir que os algoritmos têm um efeito na distância máxima. Como a ANOVA não nos diz quais algoritmos resultam em diferentes médias dessa métrica, novamente essa questão é respondida pelo método MDS. Com a ajuda do Minitab®, pode-se construir todos os intervalos conforme a Figura 39, a seguir.

```
Grouping Information Using Fisher Method

Fisher 95% Individual Confidence Intervals
All Pairwise Comparisons among Levels of Algorithm

Simultaneous confidence level = 71,65%


Algorithm = 1 subtracted from:

Algorithm   Lower  Center  Upper    --+---------+---------+---------+-----
2           -143,7   -75,3   -6,8       (--------*-------)
3           -122,9   -54,4   14,0          (-------*--------)
4           -173,7  -105,2  -36,8    (--------*-------)
5           -165,9   -97,4  -29,0     (--------*-------)
                                    --+---------+---------+---------+-----
                                   -160       -80         0        80



Algorithm = 2 subtracted from:

Algorithm  Lower  Center  Upper    --+---------+---------+---------+------
3           -47,6    20,8   89,3                  (--------*-------)
4           -98,4   -30,0   38,5             (--------*--------)
5           -90,6   -22,2   46,3              (--------*--------)
                                   --+---------+---------+---------+------
                                  -160       -80         0        80



Algorithm = 3 subtracted from:

Algorithm   Lower  Center  Upper    --+---------+---------+---------+-----
4           -119,3   -50,8   17,7          (--------*-------)
5           -111,4   -43,0   25,5           (--------*-------)
                                    --+---------+---------+---------+-----
                                   -160       -80         0        80



Algorithm = 4 subtracted from:

Algorithm   Lower  Center  Upper   --+---------+---------+---------+------
5           -60,6     7,8   76,3                (--------*--------)
                                   --+---------+---------+---------+------
```

42	
43	-160 -80 0 80
44	Algorithm N Mean Grouping
45	1 160 0,76 A
46	3 160 0,49 A B
47	2 160 0,67 B
48	5 160 0,19 B
49	4 160 0,27 B
	Means that do not share a letter are significantly different.

Figura 39: Resultados do teste MDS para Distância Máxima Fonte: Minitab®

A partir dos intervalos para diferenças das médias entre os pares de algoritmos apresentados na Figura 39, pode-se visualizar que o algoritmo 1 (GMO_PSAP) apresenta média que difere das médias dos algoritmos 2 (MOVNS_PSAP), 4 (MOVNS_I_PSAP) e 5 (PILS_PSAP), pois os intervalos para $\mu_1 - \mu_2$ (linha 11), $\mu_1 - \mu_4$ (linha 13) e $\mu_1 - \mu_5$ (linha 14) não contém o zero, com 95% de confiança. Logo, pode-se afirmar que há evidências estatísticas de que os valores médios relativos à medida de distância máxima são diferentes entre os pares de algoritmos: GMO_PSAP × MOVNS_PSAP, GMO_PSAP × MOVNS_I_PSAP e GMO × PILS_PSAP.

6.4.2. Diferença de Hipervolume

Assim como foi feito para as medidas de distância, antes de aplicar a ANOVA aos dados da diferença de hipervolume, é necessário verificar a adequação da premissa *iii*. Para isso, utilizou-se novamente o teste de hipótese que afirma a igualdade das variâncias contra a diferença entre elas, da seguinte maneira:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \tag{9}$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \tag{10}$$

A hipótese (9) representa a hipótese da igualdade das variâncias populacionais da diferença de hipervolume para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação à variância desta métrica. Já a hipótese (10), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,56. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha <$ *p-value*, aceita-se a hipótese da igualdade das variâncias entre os dados populacionais para os cinco algoritmos, a um nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, a premissa *iii* também é verificada.

Verificadas as premissas, aplica-se a ANOVA aos dados da métrica em questão, onde testou-se as seguintes hipóteses:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \tag{11}$$

$$H_1 : \mu_i \neq \mu_j \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \tag{12}$$

A hipótese (11) representa a hipótese da igualdade das médias populacionais da diferença de hipervolume para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação a esta métrica. Já a hipótese (12), conjectura o contrário.

Primeiramente procede-se com o teste (11)-(12), verificando qual das hipóteses deve ser rejeitada através do cálculo do *p-value* para o teste. Com a ajuda do Minitab®, encontrou-se *p-value* = 0,443. Logo, pode-se afirmar, a um nível de 5% de significância ($\alpha = 0{,}05$), que a hipótese (11) deve ser aceita, isto é, como $\alpha <$ *p-value*, há evidências estatísticas suficientes para concluir que os valores médios referentes à diferença de hipervolume são iguais entre os algoritmos. Existe, portanto, uma forte evidência para

concluir que os algoritmos não têm um efeito na diferença de hipervolume. Para verificar a não existência de diferença significativa entre os algoritmos propostos com relação às médias para esta métrica, foi utilizado o método MDS. Com a ajuda do Minitab®, pode-se construir todos os intervalos conforme a Figura 40, a seguir.

```
Grouping Information Using Fisher Method

Fisher 95% Individual Confidence Intervals
All Pairwise Comparisons among Levels of Algorithm

Simultaneous confidence level = 71,65%

Algoritmoc = 1 subtracted from:

Algoritmoc  Lower  Center  Upper  ---------+---------+---------+---------+
2           -1586    -327    933         (---------*----------)
3           -2325   -1066    193  (---------*----------)
4           -2050    -791    469    (---------*----------)
5           -2113    -853    406   (----------*---------)
                                  ---------+---------+---------+---------+
                                       -1200         0      1200      2400


Algoritmoc = 2 subtracted from:

Algoritmoc  Lower  Center  Upper  ---------+---------+---------+---------+
3           -1999    -739    520    (----------*---------)
4           -1723    -464    796       (---------*----------)
5           -1786    -526    733      (----------*---------)
                                  ---------+---------+---------+---------+
                                       -1200         0      1200      2400


Algoritmoc = 3 subtracted from:

Algoritmoc  Lower  Center  Upper  ---------+---------+---------+---------+
4            -984     276   1535             (---------*----------)
5           -1046     213   1472           (----------*---------)
                                  ---------+---------+---------+---------+
                                       -1200         0      1200      2400


Algoritmoc = 4 subtracted from:

Algoritmoc  Lower  Center  Upper  ---------+---------+---------+---------+
5           -1322     -63   1197         (---------*----------)
                                  ---------+---------+---------+---------+
                                       -1200         0      1200      2400
Algoritmoc   N  Mean  Grouping
1          160 2640   A
2          160 2422   A
4          160 1645   A
5          160 1956   A
3          160 1797   A

Means that do not share a letter are significantly different.
```

Figura 40: Resultados do teste MDS para a Diferença de Hipervolume Fonte: Minitab®

A partir dos intervalos para diferenças das médias entre os pares de algoritmos apresentados na Figura 40, pode-se visualizar que os algoritmos não apresentam diferenças significativas com relação à média da diferença de hipervolume, pois todos os intervalos gerados contém o zero, com 95% de confiança. Logo, pode-se afirmar que não há evidências estatísticas de que os valores médios relativos à diferença de hipervolume sejam diferentes entre os pares de algoritmos.

6.4.3. *Epsilon*

Antes de aplicar a ANOVA aos dados da métrica *epsilon*, é necessário verificar a adequação da premissa *iii*. Para isso, utilizou-se novamente o teste de hipótese que afirma a igualdade das variâncias contra a diferença entre elas, da seguinte maneira:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \quad (13)$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (14)$$

A hipótese (13) representa a hipótese da igualdade das variâncias populacionais da métrica *epsilon* para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação à variância desta métrica. Já a hipótese (14), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,077. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha <$ *p-value*, aceita-se a hipótese da igualdade das variâncias entre os dados populacionais para os cinco algoritmos, a um nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, a premissa *iii* também é verificada.

Verificadas as premissas, aplica-se a ANOVA aos dados da métrica em questão, onde testou-se as seguintes hipóteses:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \quad (15)$$

$$H_1 : \mu_i \neq \mu_j \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (16)$$

A hipótese (15) representa a hipótese da igualdade das médias populacionais da métrica *epsilon* para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação a esta métrica. Já a hipótese (16), conjectura o contrário.

Primeiramente procede-se com o teste (15)-(16), verificando qual das hipóteses deve ser rejeitada através do cálculo do *p-value* para o teste. Com a ajuda do Minitab®, encontrou-se *p-value* = 0,026. Logo, pode-se afirmar, a um nível de 5% de significância ($\alpha = 0{,}05$), que a hipótese (15) deve ser rejeitada, isto é, como $\alpha \geq$ *p-value*, há evidências estatísticas suficientes para concluir que os valores médios referentes à métrica *epsilon* são diferentes entre os algoritmos. Existe, portanto, uma forte evidência para concluir que os algoritmos tem um efeito na métrica *epsilon*. Como a ANOVA não nos diz quais algoritmos resultam em diferentes médias dessa métrica, novamente essa questão é respondida pelo método MDS. Com a ajuda do Minitab®, pode-se construir todos os intervalos conforme a Figura 41, a seguir.

```
Grouping Information Using Fisher Method

Fisher 95% Individual Confidence Intervals
All Pairwise Comparisons among Levels of Algorithm

Simultaneous confidence level = 71,65%

Algoritmod = 1 subtracted from:

Algoritmod   Lower   Center   Upper  ------+---------+---------+---------+--
2          -0,0657   0,2659  0,5974                (-----*------)
3          -0,5224  -0,1908  0,1407        (-----*------)
4          -0,0538   0,2778  0,6093                (------*-----)
5          -0,3534  -0,0218  0,3098           (------*-----)
                                  ------+---------+---------+---------+--
                                      -0,50      0,00      0,50      1,00


Algoritmod = 2 subtracted from:

Algoritmod   Lower   Center    Upper  ------+---------+---------+---------+--
3          -0,7883  -0,4567  -0,1251  (------*-----)
4          -0,3196   0,0119   0,3435           (-----*------)
5          -0,6192  -0,2877   0,0439      (-----*------)
                                  ------+---------+---------+---------+--
                                      -0,50      0,00      0,50      1,00
```

```
26
27
28  Algoritmod = 3 subtracted from:
29
30  Algoritmod   Lower  Center   Upper  ------+---------+---------+---------+--
31  4           0,1371  0,4686  0,8002                    (-----*------)
32  5          -0,1625  0,1690  0,5006               (-----*------)
33                                     ------+---------+---------+---------+--
34                                         -0,50      0,00      0,50      1,00
35
36
37  Algoritmod = 4 subtracted from:
38
39  Algoritmod   Lower   Center   Upper  ------+---------+---------+---------+--
40  5          -0,6311  -0,2996  0,0320     (------*------)
41                                      ------+---------+---------+---------+--
42                                          -0,50      0,00      0,50      1,00
43
44  Algoritmod   N    Mean  Grouping
45  4           160   1,54   A
46  2           160   1,81   A
47  1           160   1,30   A B
48  5           160   1,70   A B
49  3           160   1,50     B
50
    Means that do not share a letter are significantly different.
```

Figura 41: Resultados do teste MDS para Métrica Epsilon Fonte: Minitab®

A partir dos intervalos para diferenças das médias entre os pares de algoritmos apresentados na Figura 41, pode-se visualizar que o algoritmo 3 (GMOVNS_PSAP) apresenta média que difere das médias dos algoritmos 2 (MOVNS_PSAP) e 4 (MOVNS_I_PSAP), pois os intervalos para $\mu_2 - \mu_3$ (linha 22), $\mu_3 - \mu_4$ (linha 31) não contém o zero, com 95% de confiança. Logo, pode-se afirmar que há evidências estatísticas de que os valores médios relativos à métrica *epsilon* são diferentes entre os pares de algoritmos: GMOVNS_PSAP × MOVNS_PSAP e GMOVNS_PSAP × MOVNS_I_PSAP.

6.4.4. Taxa de Erro

Para a métrica taxa de erro, antes de aplicar a ANOVA, é necessário verificar a adequação da premissa *iii*. Para isso, utilizou-se o teste de hipótese que afirma a igualdade das variâncias contra a diferença entre elas, da seguinte maneira:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \tag{17}$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \tag{18}$$

A hipótese (17) representa a hipótese da igualdade das variâncias populacionais da taxa de erro para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação à variância desta métrica. Já a hipótese (18), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,008. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha >$ *p-value*, rejeita-se a hipótese da igualdade das variâncias entre os dados populacionais para os cinco algoritmos, a um nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, a premissa *iii* não é verificada.

Sendo assim, não atendidas todas as premissas, não pode-se aplicar a ANOVA aos dados da métrica em questão. Uma alternativa nesse caso é fazer uso do teste não-paramétrico de *Kruskal-Wallis* (Montgomery e Runger, 2009). O viés deste teste, assim como todas as estatísticas não-paramétricas, é o poder reduzido em detectar pequenas diferenças entre os fatores (algoritmos). A diferença da ANOVA para o teste não-paramétrico de *Kruskal-Wallis* é que, este último, ao invés de trabalhar com médias, faz

uso de medianas. O teste para a mediana pode ser usado para testar a igualdade de medianas (η) de duas ou mais populações e, aplicado a este trabalho, testa as seguintes hipóteses:

$$H_0 : \eta_1 = \eta_2 = \eta_3 = \eta_4 = \eta_5 \quad (19)$$

$$H_1 : \eta_i \neq \eta_j \text{ para pelo menos um par } (i, j), \text{ com } i, j = 1, 2, 3, 4, 5 \text{ e } i \neq j \quad (20)$$

A hipótese (19) representa a hipótese da igualdade das medianas populacionais da taxa de erro para os cinco algoritmos, ou seja, conjectura que não há diferença significativa entre estes algoritmos com relação a esta métrica. Já a hipótese (20), conjectura o contrário. Para este teste de hipóteses, o cálculo da estatística de teste *p-value* foi feito no Minitab® e retornou o valor 0,001. Adotando-se um nível de significância $\alpha = 0{,}05$ para o teste, e uma vez que $\alpha >$ *p-value*, rejeita-se a hipótese da igualdade das medianas entre os dados populacionais para os cinco algoritmos, a um nível de 5% de significância, isto é, com probabilidade de 5% de se estar cometendo erro nessa inferência. Portanto, os algoritmos diferem com relação à taxa de erro.

A Tabela 14, a seguir, apresenta os resultados das comparações pareadas. Nesta tabela, a primeira coluna indica o par de algoritmos que é comparado e a segunda coluna é *booleana* que assume valor *TRUE* se existe diferença estatística entre os algoritmos comparados e valor *FALSE*, caso contrário.

Tabela 14: Teste de Kruskal-Wallis para a Métrica Taxa de Erro

Par de Algoritmos	**Diferença**
GMO_PSAP × MOVNS_PSAP	*TRUE*
GMO_PSAP × GMOVNS_PSAP	*TRUE*
GMO_PSAP × MOVNS_I_PSAP	*TRUE*
GMO_PSAP × PILS_PSAP	*TRUE*
MOVNS_PSAP × GMOVNS_PSAP	*FALSE*
MOVNS_PSAP × MOVNS_I_PSAP	*FALSE*
MOVNS_PSAP × PILS_PSAP	*TRUE*
GMOVNS_PSAP × MOVNS_I_PSAP	*FALSE*
GMOVNS_PSAP × PILS_PSAP	*TRUE*
MOVNS_I_PSAP × PILS_PSAP	*FALSE*

Pelos resultados da Tabela 14, pode-se observar que os valores medianos da métrica taxa de erro são diferentes entre os pares de algoritmos: GMO_PSAP × MOVNS_PSAP, GMO_PSAP × GMOVNS_PSAP, GMO_PSAP × MOVNS_I_PSAP, GMO_PSAP × PILS_PSAP, MOVNS_PSAP × PILS_PSAP e GMOVNS_PSAP × PILS_PSAP.

No capítulo a seguir é apresentada a aplicação dos cinco algoritmos propostos em um exemplo fictício de um projeto de construção civil utilizando estruturas metálicas. É apresentada, também, uma análise da relação entre a disponibilidade de recursos e os objetivos utilizados.

Capítulo 7

7. APLICAÇÃO DOS ALGORITMOS E ANÁLISE DE RESULTADOS

Com o intuito de exemplificar a aplicação dos cinco algoritmos descritos no Capítulo 5, é proposto neste capítulo um exemplo fictício e simplificado de um projeto de construção civil utilizando estruturas metálicas. Em seguida é apresentada uma análise acerca da influência da disponibilidade de recursos, com relação aos objetivos adotados, utilizando um novo cenário para o exemplo proposto.

7.1. Descrição do Exemplo

O projeto descrito pelo exemplo proposto trata da construção de um galpão de 500 m^2 com pé direito de 9,0 m e fundação rasa (tipo sapata corrida) e, supõe-se que as ligações entre as peças da estrutura sejam parafusadas.

Os conjuntos de atividades e de recursos, bem como as demandas das atividades pelos recursos e as relações de precedência entre as atividades, presentes no exemplo foram criados com base na ABNT NBR 8800, em Bellei *et al*. (2008) e em um exemplo proposto em Jaskowski e Sobotka (2006). Vale ressaltar que diversas outras atividades e recursos poderiam ser incluídos no exemplo, ou até mesmo alguma atividade pertencente ao exemplo poderia ser desmembrada em outras atividades, mas da forma que está, o exemplo atende aos propósitos deste capítulo.

Devido à dificuldade de obtenção de dados reais e ao fato de que a duração e o custo de execução das atividades variam de um projeto para outro, no exemplo utilizado estes dados foram definidos aleatoriamente. Portanto, os valores utilizados para as durações e custos de execução das atividades podem estar bem fora da realidade. Esta questão, no entanto, não afetou o entendimento das analises e conclusões feitas acerca da resolução do exemplo. O intuito aqui é exemplificar a aplicação dos algoritmos propostos.

O projeto proposto no exemplo contém 30 atividades e 20 recursos recuperáveis, conforme descrito a seguir.

ATIVIDADES

- Etapa Inicial

1. Projeto Inicial (arquitetônico e estrutural)
2. Escolha do aço e dos perfis a serem utilizados
3. Preparação do terreno (escavação, nivelamento, etc.)
4. Preparação da fundação
5. Projeto detalhado das peças (pilares e vigas) e dos tipos de ligação

- Etapa de Fabricação das Peças

6. Traçagem
7. Corte
8. Acabamento
9. Pré-deformação (para as peças que forem soldadas)
10. Solda
11. Desempeno (para as peças que precisarem de desempeno)
12. Dobra (fabricação dos perfis)
13. Furação (para as ligações parafusadas)

14. Ponteamento
15. Preparação da superfície
16. Pintura e/ou proteção contra corrosão das peças
17. Transporte das peças

- Etapa de Montagem das Peças

18. Preparação das bases das colunas (pilares)
19. Nivelamento das colunas
20. Posicionamento dos componentes de desenvolvimento horizontal (colocação das vigas e treliças)
21. Estabilização do conjunto/alinhamento
22. Ajustes
23. Execução das ligações definitivas

- Etapa Final

24. Definição e montagem dos tipos de fechamentos verticais a serem utilizados (paredes)
25. Definição e montagem do tipo de cobertura a ser utilizado (telhado)
26. Montagem das instalações elétrica, hidráulica, de esgoto, etc. (pode ser realizada em paralelo com a montagem dos fechamentos verticais)
27. Preparação e colocação dos pisos
28. Instalação das janelas, portas e portões
29. Montagem e instalação dos elementos de proteção contra incêndio
30. Acabamentos Finais (acertos, retoques, pintura, etc.)

RECURSOS

- Mão-de-obra

1. Engenheiro
2. Projetista/Desenhista Industrial/Riscador de estruturas de aço
3. Mestre (obras, pintura, etc.)
4. Pintores/Jatistas
5. Operadores de máquina de corte (serralheiro)
6. Maçariqueiros/Soldadores/Soldadores de Ponteamento
7. Operadores de guindaste/Motoristas (em geral)
8. Ajudantes (fabricação, pintura, montagem, parafusamento/soldagem, montagem, preparação, lixamento, etc.)
9. Montadores
10. Bombeiros

- Máquinas, Ferramentas e Equipamentos

11. Máquina de corte de aço
12. Máquina de furar aço (furadeira)
13. Guindastes/elevadores
14. Caminhões/carreta (transporte das peças, etc.)
15. Máquina de solda
16. Máquina de pintura
17. Máquina para processos auxiliares (torno, calandra, etc.)
18. Lixadeira
19. Tratores/escavadeiras
20. Parafusadeiras

São apresentadas na Tabela 15 a duração (em u. t.), o custo (em u. m.) e as atividades predecessoras das 30 atividades do exemplo.

Tabela 15: Duração, custo e predecessoras das atividades do exemplo

Atividade (i)	**Duração** (p_i)	**Custo** (c_i)	**Atividades Predecessoras**
1	15	20000	---
2	3	5000	1
3	10	10000	---
4	11	9000	1, 3
5	7	11000	1
6	7	3000	2, 5
7	10	10500	6
8	3	2000	7
9	4	3000	8
10	6	7000	9
11	5	2500	10
12	11	5000	8
13	7	3000	11, 12
14	6	3500	13
15	6	2000	14
16	8	7000	15
17	12	12000	4, 16
18	9	9500	17
19	7	6500	18
20	8	7500	19
21	5	5000	20
22	3	2000	21
23	9	7500	22
24	14	16500	1, 23
25	18	17500	1, 23
26	11	10000	23
27	10	9000	26
28	8	7000	24
29	7	6000	25, 26, 28
30	6	3000	25, 26, 27, 28

Como pode ser visto na Tabela 15, a atividade 1 possui duração igual a 15 u. t., custo igual a 20000 u. m. e não possui atividades predecessoras. Já a atividade 2 possui duração igual a 3 u. t., custo igual a 5000 u. m. e a atividade 1 como predecessora. E assim por diante. De acordo com a Tabela 15, as atividades 1, 24 e 25 são as que apresentam a maior duração (p_i), isto é, 15, 14 e 18 u. t., respectivamente. Portando, estas atividades são as que mais contribuem para elevar a duração total do projeto.

Na Tabela 16 são descritas as demandas das atividades pelos 20 recursos, bem como a disponibilidade dos mesmos.

Tabela 16: Demanda das atividades pelos recursos e disponibilidade de recursos

Recurso/ Atividade	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	1	5	0	0	0	0	0	2	0	0	0	0	1	0
4	0	0	1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	2	0	0	2	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	3	0	0	0
9	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	2	0	2	0	0	0	0	0	0	2	0	0	0	0	0
11	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0
12	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	2	0	0	0
13	0	0	0	0	0	0	0	2	0	0	0	2	0	0	0	0	0	0	0	0
14	0	1	0	0	0	1	0	3	2	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	1	0	0
16	0	0	1	2	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
17	0	0	0	0	0	0	2	2	0	0	0	0	1	1	0	0	0	0	0	0
18	0	0	1	0	0	0	2	3	0	0	0	0	1	0	0	0	0	0	1	0
19	0	0	1	0	0	0	1	2	0	0	0	0	1	0	0	0	0	0	0	0
20	0	0	1	0	0	0	1	2	2	0	0	0	1	0	0	0	0	0	0	0
21	0	0	1	0	0	0	1	2	2	0	0	0	1	0	0	0	0	0	0	0
22	0	0	1	0	0	0	1	2	2	0	0	0	1	0	0	0	0	0	0	0
23	0	0	1	0	0	1	0	2	2	0	0	0	0	0	1	0	0	0	0	2
24	1	0	1	0	0	0	1	2	2	0	0	0	1	0	0	0	0	0	0	0
25	1	0	1	0	0	0	1	4	2	0	0	0	1	0	0	0	0	0	0	0
26	0	0	1	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0
27	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	1	0	0	0	0	2	0	0	0	2	0	0	0	0	0	0	0	2
29	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
30	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0
Disponibilidade	1	1	1	2	1	2	2	5	2	1	2	2	1	2	2	1	3	1	1	2

Conforme visualizado na Tabela 16, a atividade 1 demanda apenas uma unidade do recurso 1; a atividade 2 demanda, também, apenas uma unidade do recurso 1; e assim sucessivamente. O exemplo proposto apresenta como disponibilidade de cada recurso a quantidade mínima necessária para a execução das atividades, ou seja, duas atividades que possuam demandas pelo mesmo recurso não podem ser executadas paralelamente, mesmo sem apresentarem relações de precedência. Da forma que está, qualquer redução na disponibilidade de algum dos recursos torna o problema inviável.

Na seção a seguir são apresentados os resultados obtidos pelos cinco algoritmos, em termos do conjunto de soluções não-dominadas de referência (*Ref*), para o exemplo proposto.

7.2. Apresentação e Análise dos Resultados

Para a resolução do exemplo proposto, os cinco algoritmos foram executados no mesmo computador e com as mesmas características descritas no Capítulo 6. As funções objetivo utilizadas também foram as mesmas usadas no Capítulo 6, ou seja:

- Duração: $Min\ f_1(x) = S_{n+1}$
- Custo: $Min\ f_2(x) = \sum_{i=1}^{n} \frac{c_i}{S_i}$

As soluções descritas na Tabela 17 representam o conjunto de soluções não-dominadas de referência (*Ref*) obtido através da comparação entre os conjuntos de soluções obtidos pelos cinco algoritmos. A forma como é determinado o conjunto *Ref* é apresentada na seção 6.2.

Tabela 17: Soluções do conjunto *Ref* obtidas para o exemplo

Solução	$f_1(x)$	$f_2(x)$
1	212	33406
2	214	33305
3	220	33090
4	223	32977
5	217	33196

Como pode ser visto na Tabela 17, o conjunto *Ref* obtido para o exemplo é composto por cinco soluções, onde a solução 1 apresenta duração de 212 u. t. e custo de 33406 u. m., a solução 2 214 u. t. e 33305 u. m., e assim por diante. O diagrama de Pareto resultante do conjunto *Ref* descrito na Tabela 17 é apresentado na Figura 42.

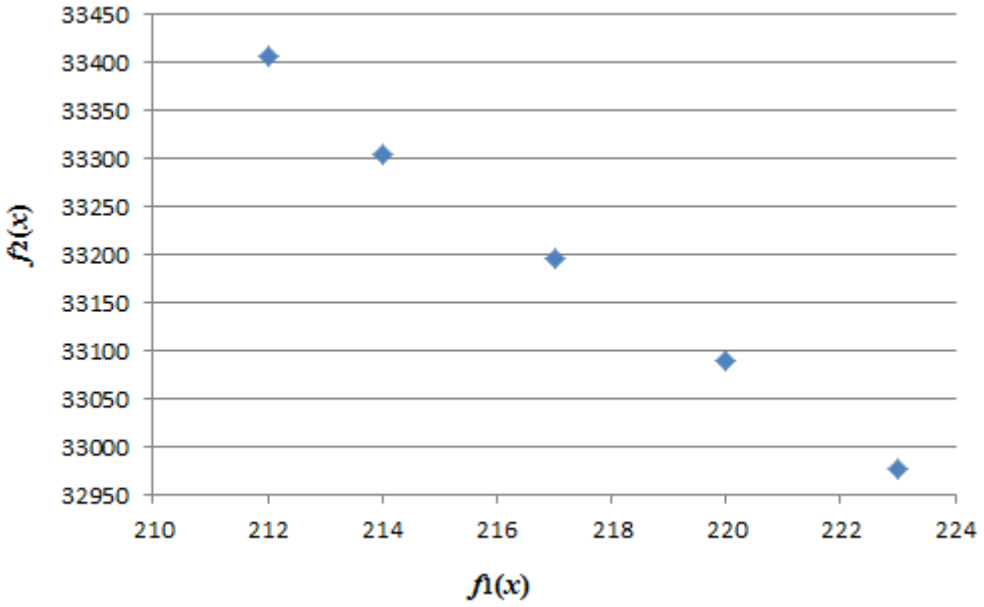

Figura 42: Diagrama de Pareto resultante do conjunto *Ref* obtido para o exemplo

Com base no conjunto *Ref* obtido para o exemplo, se o tomador de decisões (projetista) priorizar o objetivo duração ($f_1(x)$), ele optará pela solução 1, que apresenta o menor valor para tal objetivo, ou seja, 212 u. t.. Porém, a solução 1 apresenta o maior valor para o custo ($f_2(x)$), isto é, 33406 u. m.. O gráfico de Gantt para esta solução pode ser visto na Figura 43.

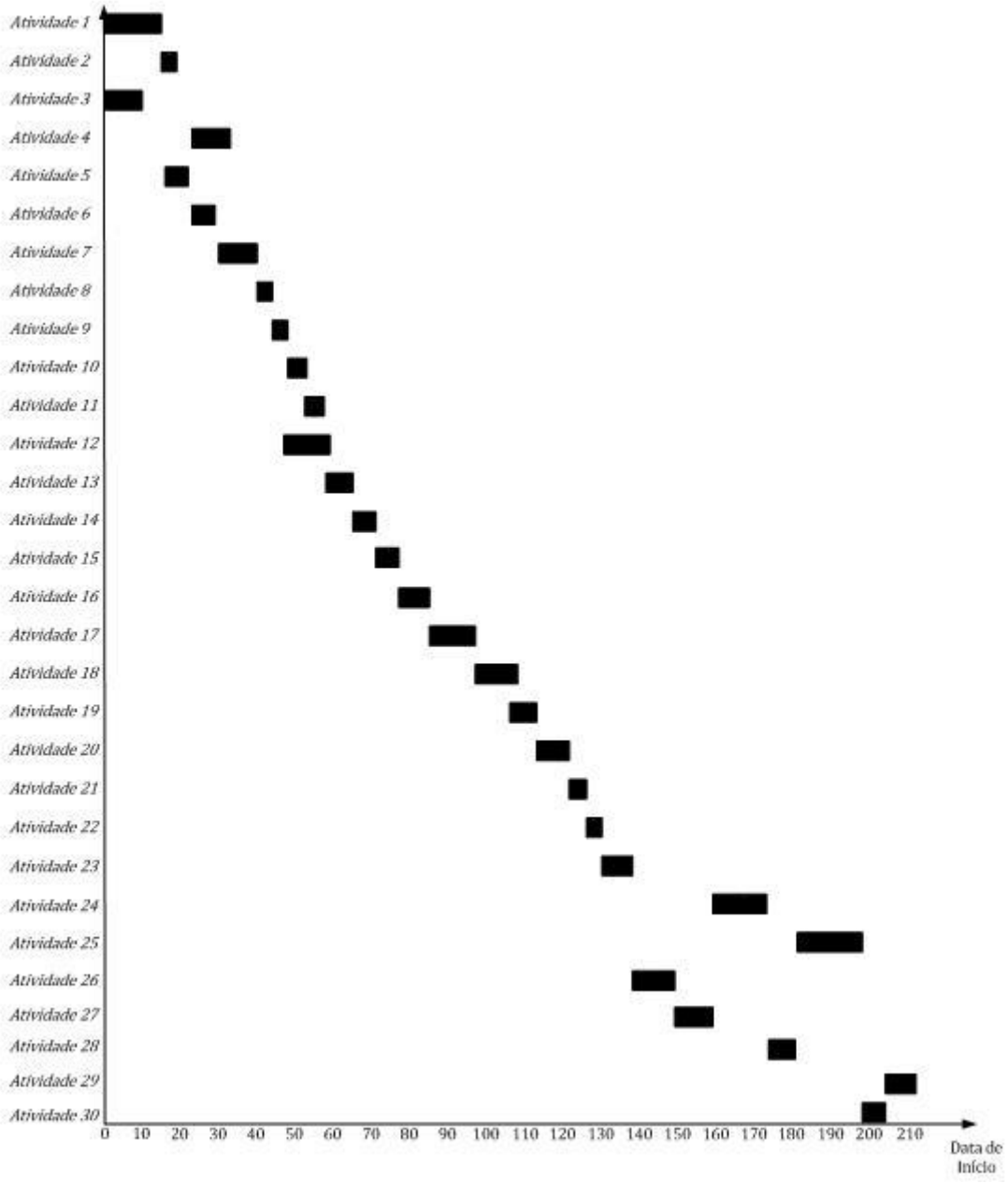

Figura 43: Gráfico de Gantt da solução 1

Como pode ser visto na Figura 43, as datas iniciais de execução das atividades na solução 1 são S = {0, 16, 0, 23, 16, 23, 30, 40, 43, 47, 53, 47, 58, 65, 71, 77, 85, 97, 106, 113, 121, 126, 129, 159, 181, 138, 149, 173, 205, 199}. Portanto, estas devem ser as datas adotadas para o início da execução das atividades, obtendo, dessa forma, a menor duração para o projeto. Caso o tomador de decisões necessite reduzir ainda mais a duração do projeto, ele deve tentar otimizar a execução das atividades que mais contribuem para elevar a duração, como as atividades 1, 24 e 25.

Os custos de execução das atividades para a solução 1, calculados de acordo com a função objetivo $f_2(x)$, são C = {20000, 312, 10000, 391, 687, 130, 350, 50, 69, 148, 47, 106, 51, 53, 28, 90, 141, 97, 61, 66, 41, 15, 58, 103, 96, 72, 60, 40, 29, 15}. Analisando os custos de execução, verifica-se que as atividades 1 e 3 são responsáveis por grande parte do custo total do projeto, ou seja, por, aproximadamente, 90% deste custo. Sendo assim, caso o tomador de decisões deseje reduzir o custo total do projeto,

mantendo as datas de início de execução das atividades obtidas para a solução 1, ele deve tentar reduzir os custos destas atividades.

O gráfico apresentado na Figura 44 descreve a participação, em porcentagem, de cada atividade na duração e custo totais obtidos para a solução 1.

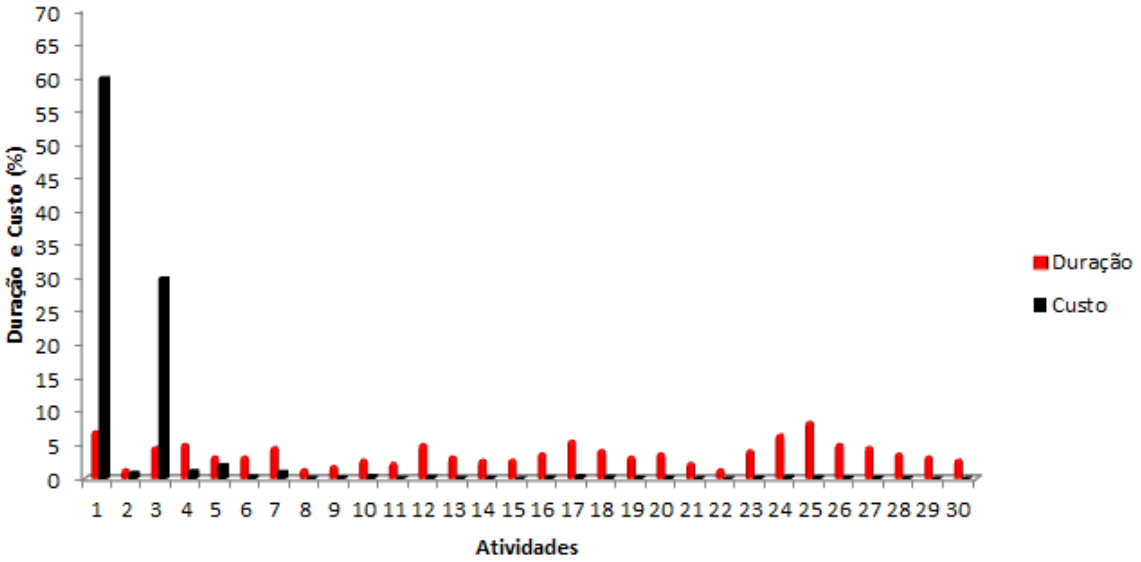

Figura 44: Participação das atividades na duração e custo totais da solução 1

Como pode ser observado na Figura 44, a atividade 1 é responsável por, aproximadamente, 7,08% da duração total e 59,87% do custo total obtidos na solução 1. A atividade 2 é responsável por 1,42% da duração total e 0,93% do custo total. E assim por diante.

Por outro lado, se o tomador de decisões optar por priorizar o objetivo custo ($f_2(x)$), ele escolherá a solução 4, com valor para tal objetivo igual a 32977 u. m.. Entretanto, a solução 4 apresenta o maior valor para o objetivo duração ($f_1(x)$), ou seja, 223 u. t.. O gráfico de Gantt para tal solução é apresentado na Figura 45.

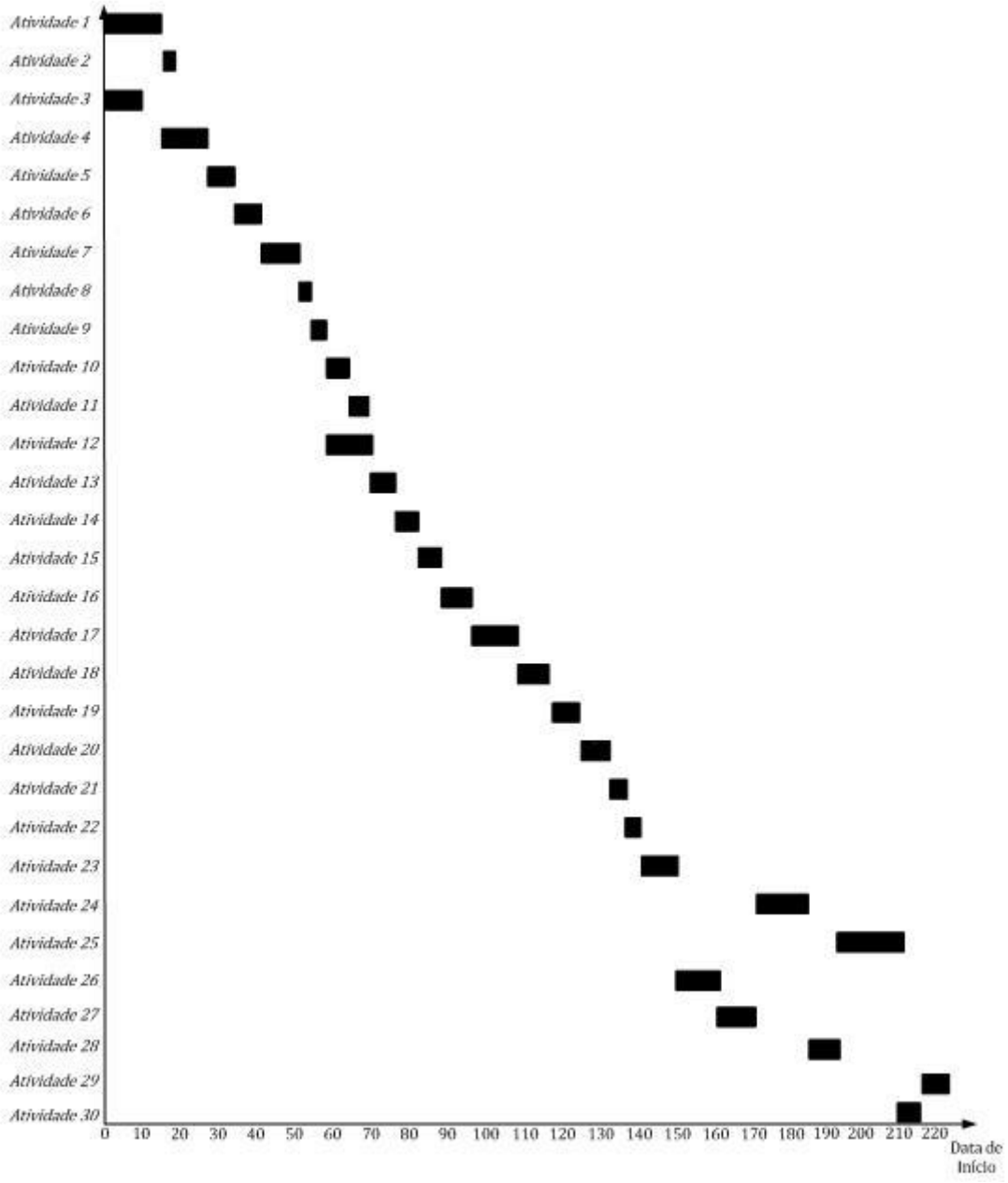

Figura 45: Gráfico de Gantt da solução 4

Como pode ser visto na Figura 45, as datas de início das atividades na solução 4 são S = {0, 16, 0, 16, 27, 34, 41, 51, 54, 58, 64, 58, 69, 76, 82, 88, 96, 108, 117, 124, 132, 137, 140, 170, 192, 149, 160, 184, 216, 210}. Sendo assim, o tomador de decisões adotará estas datas para iniciar as atividades, obtendo o menor custo para o projeto. Caso ele necessite reduzir a duração do projeto, deverá tentar otimizar a execução das atividades que mais contribuem para elevar a duração, como as atividades 1, 24 e 25.

Os custos de execução das atividades para a solução 4, calculados de acordo com a função objetivo $f_2(x)$, são C = {20000, 312, 10000, 562, 407, 88, 256, 39, 55, 120, 39, 86, 43, 46, 24, 79, 125, 87, 55, 60, 37, 14, 53, 97, 91, 67, 56, 38, 27, 14}. Analisando os custos de execução, novamente as atividades responsáveis pela maior parcela do custo total do projeto são a 1 e 3, ou seja, estas atividades são responsáveis por, aproximadamente, 91% deste custo. Dessa forma, caso o tomador de decisões deseje reduzir ainda mais o custo total do projeto, mantendo as datas de início de execução obtidos para a solução 4, ele deve tentar reduzir os custos destas atividades.

O gráfico apresentado na Figura 46 descreve a participação, em porcentagem, de cada atividade na duração e custo totais obtidos para a solução 4.

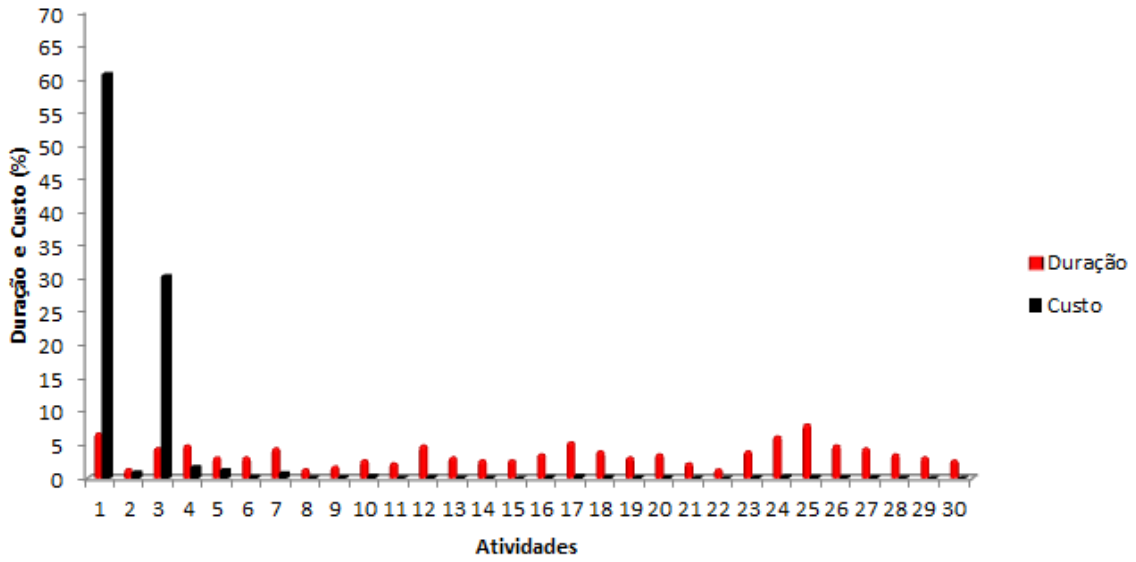

Figura 46: Participação das atividades na duração e custo totais da solução 4

Como pode ser observado na Figura 46, a atividade 1 é responsável por, aproximadamente, 6,73% da duração total e 60,65% do custo total obtidos para a solução 4. A atividade 2 é responsável por 1,35% da duração total e 0,95% do custo total. E assim por diante.

Na seção a seguir é apresentada uma análise acerca dos resultados obtidos para o exemplo com relação à disponibilidade de recursos. Para isso, é utilizado um novo cenário, também descrito na seção a seguir.

7.2.1. Novo Cenário e Comparação dos Resultados

Como no exemplo proposto a disponibilidade de cada recurso é a quantidade mínima necessária para a execução das atividades (Tabela 16), duas atividades que possuam demandas pelo mesmo recurso não podem ser executadas simultaneamente (cenário inicial). Para a comparação de resultados, supõe-se agora um novo cenário em que há disponibilidade de recursos suficiente para que duas ou mais atividades que necessitem do mesmo recurso possam ser realizadas paralelamente. As disponibilidades dos 20 recursos para o novo cenário são apresentadas na Tabela 18.

Tabela 18: Disponibilidades dos 20 recursos para o novo cenário

Recurso	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Disponibilidade	2	1	3	2	1	2	2	8	4	1	2	2	2	2	2	1	3	1	1	2

Como pode ser observado na Tabela 18, para o novo cenário foram alteradas somente as disponibilidades dos recursos 1, 3, 8, 9 e 13. Foram realizados testes com disponibilidades de recursos maiores do que as utilizadas neste novo cenário, e os resultados obtidos foram os mesmos. Entretanto, os recursos disponibilizados a mais ficaram ociosos.

Os resultados obtidos através da resolução deste novo cenário são comparados com as soluções obtidas para o cenário inicial, analisando-se a influência da disponibilidade de recursos na duração e no custo do projeto representado pelo exemplo. As soluções obtidas (conjunto *Ref*) para o novo cenário são apresentas na Tabela 19.

Tabela 19: Soluções do Conjunto *Ref* obtidas para o novo cenário

Solução	$f_1(x)$	$f_2(x)$
1	174	33457
2	178	33350
3	185	33190
4	200	32898
5	189	33098

De acordo com a Tabela 19, o conjunto *Ref* obtido para o novo cenário também é composto por cinco soluções, onde a solução 1 apresenta duração de 174 u. t. e custo de 33457 u. m., a solução 2 178 u. t. e 33350 u. m., e assim por diante. O diagrama de Pareto resultante do conjunto *Ref* descrito na Tabela 19 é apresentado na Figura 47.

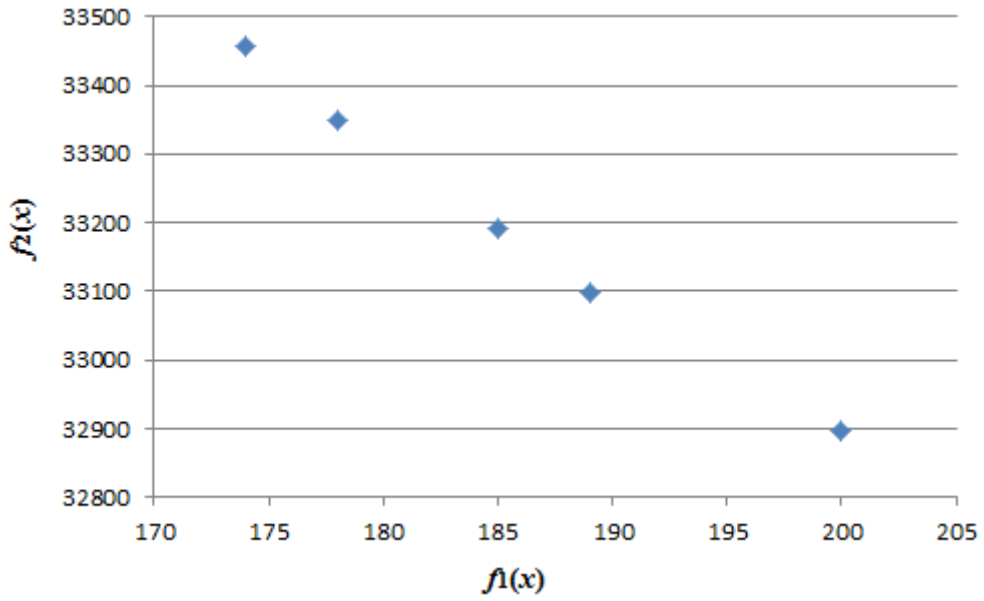

Figura 47: Diagrama de Pareto resultante do conjunto *Ref* obtido para o novo cenário

A Figura 48 apresenta um diagrama de Pareto com as duas fronteiras geradas pelos conjuntos *Ref* obtidos para os dois cenários.

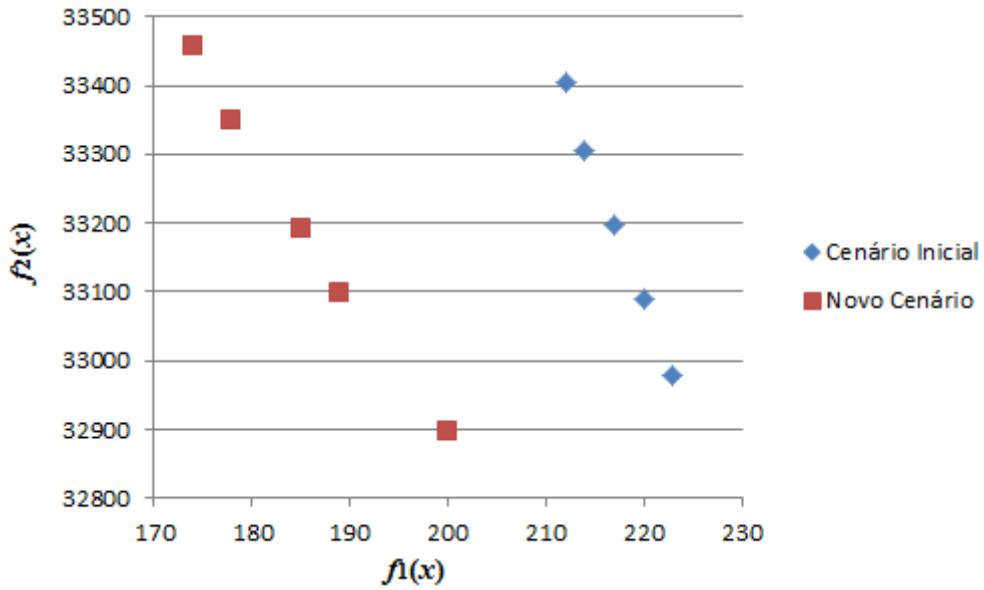

Figura 48: Comparação entre os diagramas de Pareto obtidos para os dois cenários

Como pode ser visto na Figura 48, a fronteira de Pareto obtida para o novo cenário apresenta soluções melhores, comparando-as utilizando o critério de dominância de Pareto, com as soluções da fronteira de Pareto obtida para o cenário inicial.

Baseando-se no conjunto *Ref* obtido para o novo cenário, se o tomador de decisões priorizar o objetivo duração ($f_1(x)$), ele optará pela solução 1, que apresenta o menor valor para tal objetivo, ou seja, 174 u. t.. Porém, a solução 1 apresenta o maior valor para o custo ($f_2(x)$), isto é, 33457 u. m. O gráfico de Gantt para a solução 1 do novo cenário pode ser visto na Figura 49.

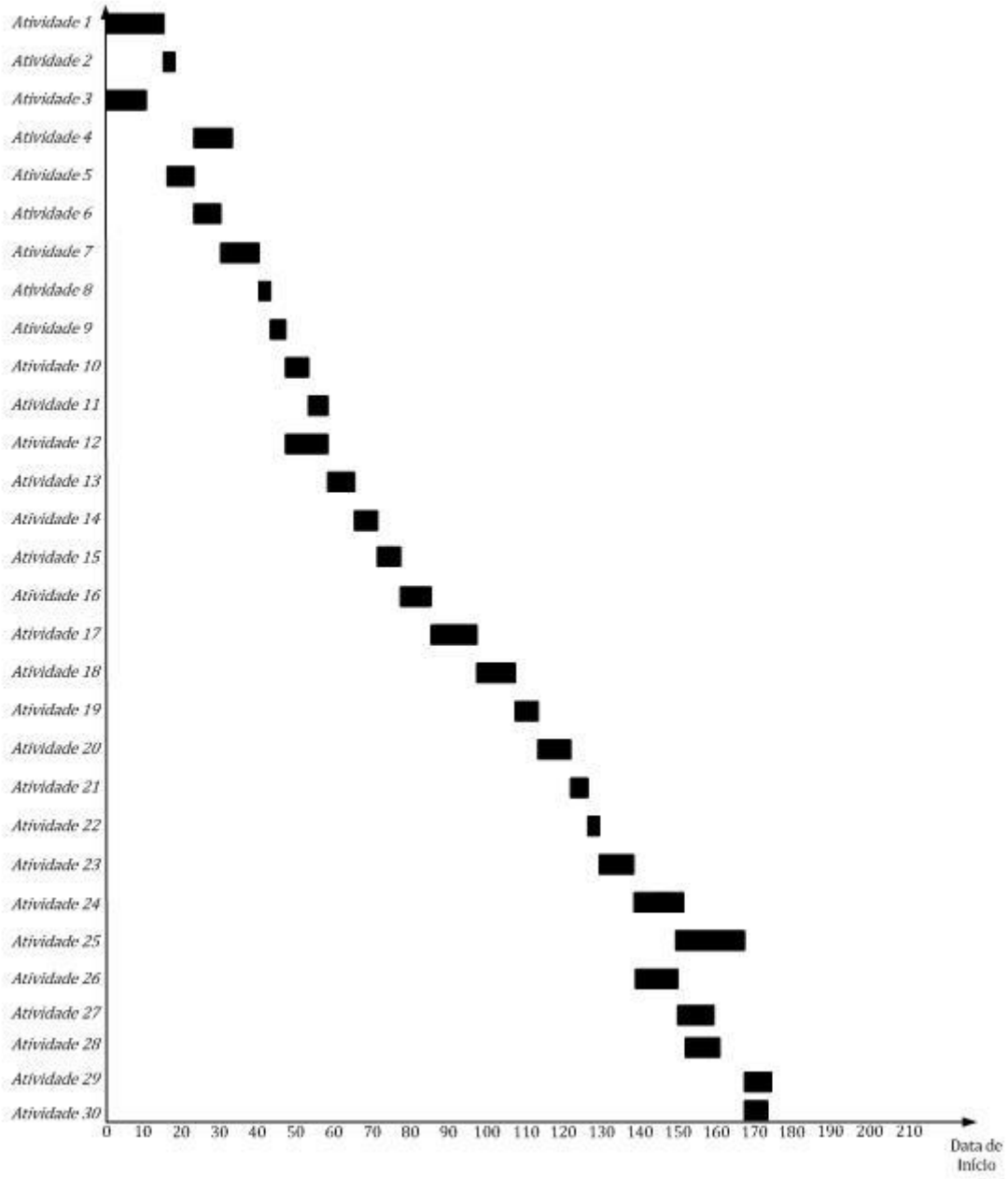

Figura 49: Gráfico de Gantt da solução 1 do novo cenário

Como pode ser observado na Figura 49, as datas de início das atividades na solução 1 do novo cenário são S = {0, 16, 0, 23, 16, 23, 30, 40, 43, 47, 53, 47, 58, 65, 71, 77, 85, 97, 106, 113, 121, 126, 129, 138, 149, 138, 149, 152, 167, 167}. Portanto, o tomador de decisões adotará estas datas para iniciar a execução das atividades, obtendo a menor duração para o projeto. Caso ele necessite reduzir ainda mais a duração do

projeto, deverá tentar otimizar a execução das atividades que mais contribuem para elevar a duração, ou seja, as atividades 1, 24 e 25.

Os custos de execução das atividades para a solução 1 do novo cenário, calculados de acordo com a função objetivo $f_2(x)$, são C = {20000, 312, 10000, 391, 687, 130, 350, 50, 69, 148, 47, 106, 51, 53, 28, 90, 141, 97, 61, 66, 41, 15, 58, 119, 117, 72, 60, 46, 35, 17}. Analisando os custos de execução, verifica-se que as atividades 1 e 3 são responsáveis por grande parte do custo total do projeto, ou seja, por, aproximadamente, 90% deste custo. Portanto, caso o tomador de decisões deseje reduzir o custo total do projeto, mantendo as datas de início de execução das atividades obtidas para a solução 1, ele deve tentar reduzir os custos destas atividades.

O gráfico apresentado na Figura 50 descreve a participação, em porcentagem, de cada atividade na duração e custo totais obtidos para a solução 1 do novo cenário.

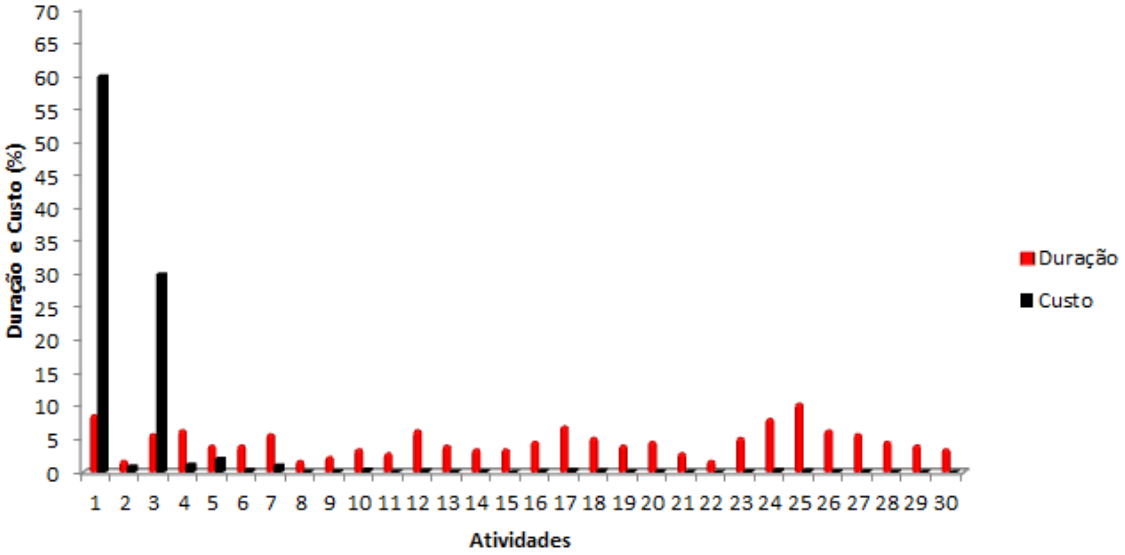

Figura 50: Participação das atividades na duração e custo totais da solução 1 do novo cenário

Como pode ser observado na Figura 50, a atividade 1 é responsável por, aproximadamente, 8,62% da duração total e 59,78% do custo total obtidos na a solução 1 do novo cenário. A atividade 2 é responsável por 1,72% da duração total e 0,93% do custo total. E assim por diante.

Comparando tal solução com a escolhida para o cenário inicial, o projeto teve uma redução em sua duração de, aproximadamente, 17,92%, ou seja, 38 u. t.. Em relação ao custo, o projeto teve um aumento de, aproximadamente, 0,15%, isto é, 51 u. m..

Por outro lado, se o tomador de decisões optar por priorizar o objetivo custo ($f_2(x)$), ele escolherá a solução 4, com valor para tal objetivo igual a 32898 u. m.. Entretanto, a solução 4 apresenta o maior valor para o objetivo duração ($f_1(x)$), ou seja, 200 u. t.. O gráfico de Gantt para a solução 4 do novo cenário pode ser visto na Figura 51.

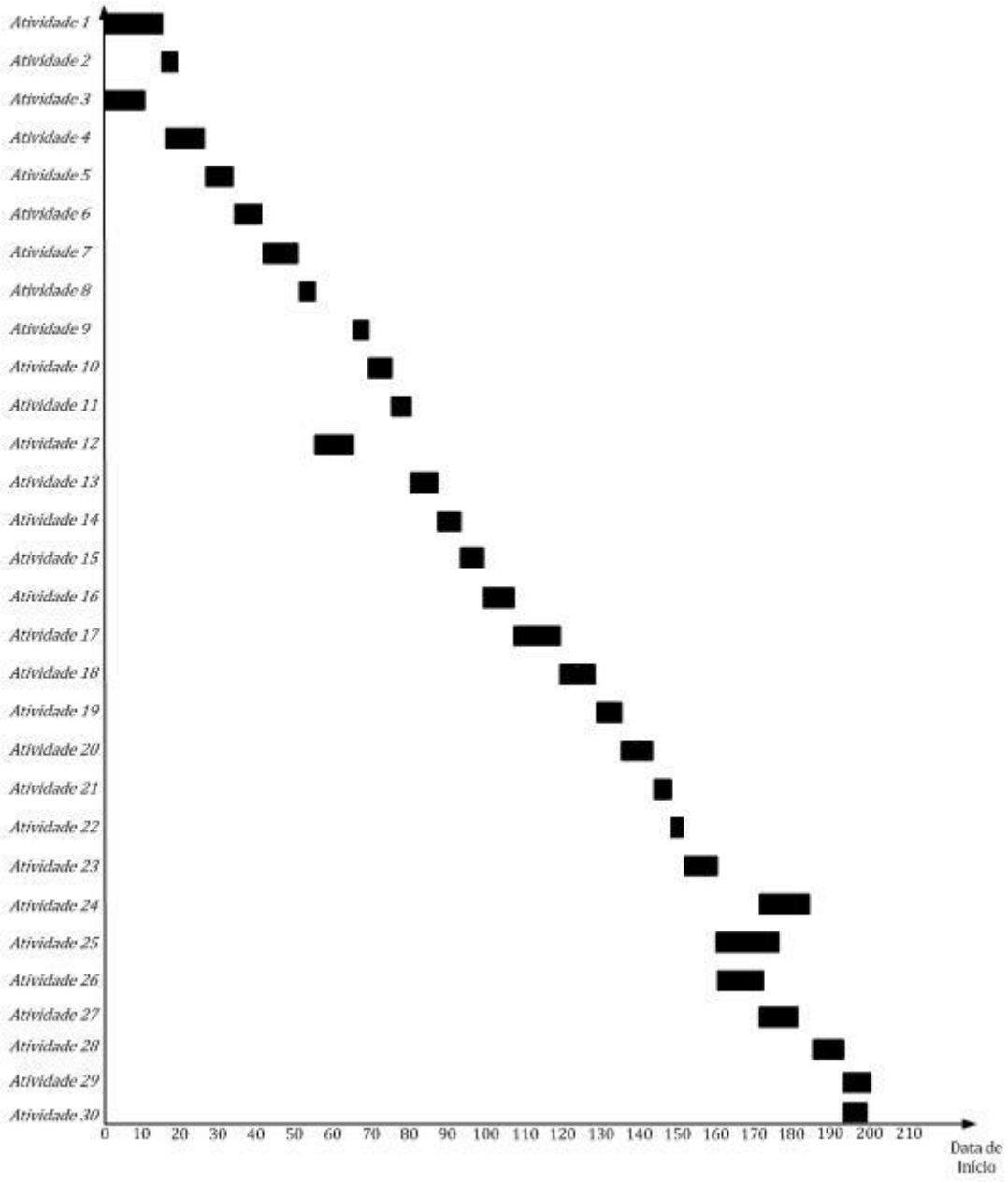

Figura 51: Grafico de Gantt da solução 4 do novo cenário

Como pode ser visto na Figura 51, as datas de início das atividades na solução 4 do novo cenário são S = {0, 16, 0, 16, 27, 34, 41, 51, 65, 69, 75, 54, 80, 87, 93, 99, 107, 119, 128, 135, 143, 148, 151, 171, 160, 160, 171, 185, 193, 193}. Portanto, serão adotadas estas datas para iniciar as atividades, obtendo, dessa forma, o menor custo para o projeto. Caso o tomador de decisões necessite reduzir a duração do projeto, ele deve tentar otimizar a execução das atividades que mais contribuem para elevar a duração do projeto, isto é, as atividades 1, 24 e 25.

Os custos de execução das atividades para a solução 4 do novo cenário, calculados de acordo com a função objetivo $f_2(x)$, são C = {20000, 312, 10000, 562, 407, 88, 256, 39, 46, 101, 33, 92, 37, 40, 21, 70, 112, 79, 50, 55, 34, 13, 49, 96, 109, 62, 52, 37, 31, 15}. Analisando os custos de execução, novamente as atividades responsáveis pela maior parte do custo total do projeto são a 1 e 3, ou seja, estas atividades são responsáveis por, aproximadamente, 91,2% deste custo. Dessa forma, caso o tomador de decisões deseje reduzir ainda mais o custo total do projeto, mantendo

as datas de inicio de execução das atividades obtidas para a solução 4, ele deve tentar minimizar os custos destas atividades.

O gráfico apresentado na Figura 52 descreve a participação, em porcentagem, de cada atividade na duração e custo totais obtidos para a solução 4 do novo cenário.

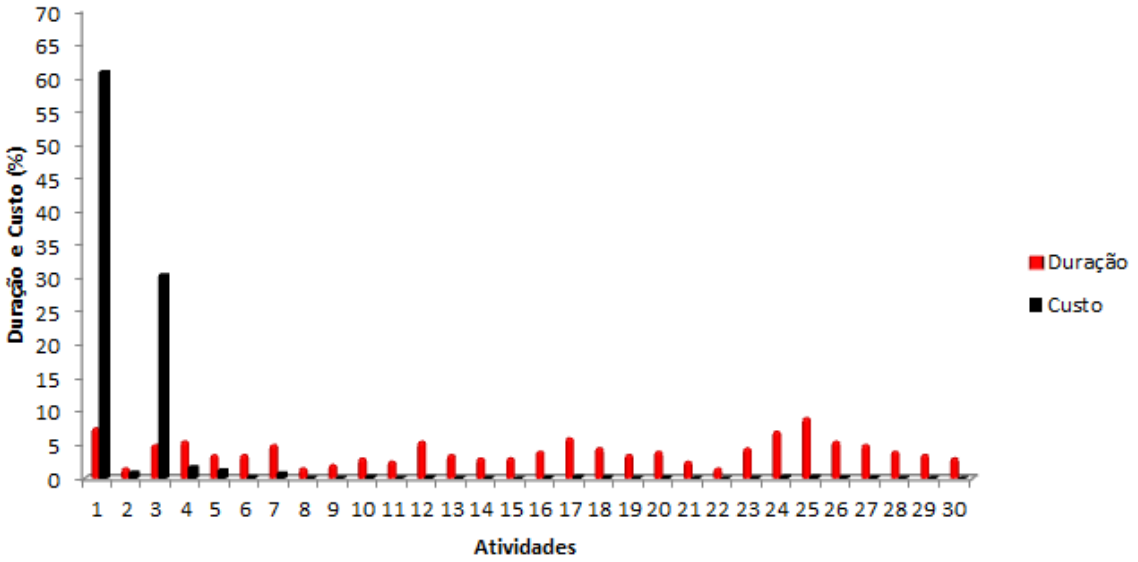

Figura 52: Participação das atividades na duração e custo totais da solução 4 do novo cenário

Como pode ser observado na Figura 52, a atividade 1 é responsável por, aproximadamente, 7,50% da duração total e 60,79% do custo total obtidos para a solução 4 do novo cenário. A atividade 2 é responsável por 1,50% da duração total e 0,95% do custo total. E assim por diante.

Comparando tal solução com a escolhida para o primeiro cenário, o custo do projeto teve uma redução de, aproximadamente, 0,24%, isto é, 79 u. m.. Em relação à duração, o projeto também teve uma redução, só que agora de, aproximadamente, 10,31%, ou seja, 23 u. t..

Suponha agora que o contratante do projeto esteja disposto a investir apenas 33200 u. m. e deseje uma solução que minimize a duração do projeto. Com base no valor que o contratante esta disposto a investir, os novos conjuntos de soluções obtidos para os dois cenários são os apresentados nas Tabelas 20 e 21.

Tabela 20: Novo conjunto *Ref* para o cenário inicial

Solução	$f_1(x)$	$f_2(x)$
1	220	33090
2	223	32977
3	217	33196

Tabela 21: Novo conjunto *Ref* para o novo cenário

Solução	$f_1(x)$	$f_2(x)$
1	185	33190
2	200	32898
3	189	33098

De acordo com as Tabelas 20 e 21, os novos conjuntos *Ref* obtidos para os dois cenários são compostos por três soluções. Como o contratante do projeto deseja a solução que minimize a duração do projeto, a solução a ser escolhida para o cenário inicial é a 3, com duração de 217 u. t. e custo de 33196 u. m., e para o novo cenário é a 1, com duração de 185 u. t. e custo de 33190 u. m.. Comparando tais soluções, a solução 1 do novo cenário apresenta uma duração cerca de, aproximadamente, 14,75% menor que a solução 3 do cenário inicial, ou seja, 32 u. t.. Em relação ao custo, a solução 1 do

novo cenário apresenta um valor cerca de, aproximadamente, 0,02% menor que a solução 3 do cenário inicial, isto é, 6 u. m..

Por fim, com base nas datas de início de execução das atividades, obtidas nas soluções escolhidas em cada situação, e de acordo com as demandas das atividades pelos recursos, o tomador de decisões pode determinar quando e quanto adquirir/contratar de cada tipo de recurso, evitando, assim, ociosidade e/ou desperdício, bem como a falta dos mesmos. Para exemplificar, considere a solução 1 obtida para o novo cenário. Na Tabela 22 são apresentadas as datas inicial e final no qual cada recurso será utilizado. Nesta tabela são apresentados, também, os períodos e as quantidades em que os recursos serão usados. A quantidade de cada recurso a ser utilizada em cada período de tempo é descrita, na Tabela 22, entre parênteses ao lado de cada período.

Tabela 22: Datas iniciais e finais e períodos de utilização dos recursos da solução 1 do novo cenário

Recurso	Data inicial de utilização	Data final de utilização	Períodos de utilização (Quantidade utilizada)
1	0	158	0-18(1); 30-39(1); 138-148(1); 149-151(2); 152-158(1)
2	30	84	30-42(1); 77-84(1)
3	16	173	16-29(1); 85-96(1); 106-137(1); 138-148(2); 149-158(3); 159-159(2); 160-166(1); 167-172(2); 173-173(1)
4	85	173	85-96(2); 167-173(1)
5	43	46	43-46(1)
6	47	148	47-57(2); 77-84(1); 138-148(1)
7	16	158	16-22(1); 97-112(2); 113-148(1); 149-151(2); 152-158(1)
8	16	173	16-29(5); 30-33(2); 43-46(2); 47-57(4); 58-76(2); 77-84(3); 85-96(1); 97-105(2); 106-112(3); 113-137(2); 138-148(6); 149-151(8); 152-158(6); 159-159(4); 160-166(2); 167-172(3); 173-173(1)
9	77	158	77-84(2); 121-137(2); 138-151(4); 152-158(2)
10	152	173	152-159(1); 167-173(1)
11	43	46	43-46(2)
12	71	172	71-76(2); 167-172(2)
13	97	158	97-148(1); 149-151(2); 152-158(1)
14	16	105	16-22(2); 97-105(1)
15	47	148	47-57(2); 138-148(1)
16	85	173	85-96(1); 167-173(1)
17	23	173	23-33(1); 47-52(3); 53-64(1); 65-70(2); 167-173(1)
18	23	173	23-33(1); 167-173(1)
19	16	112	16-29(1); 106-112(1)
20	138	172	138-148(2); 167-172(2)

De acordo com a Tabela 22, o recurso 1 terá sua primeira utilização na data 0, última utilização na data 158 e será utilizado nos períodos 0-18, 30-39, 138-148, 149-151 e 152-158, sendo necessário 1, 1, 1, 2 e 1 unidades em cada um dos períodos, respectivamente. O recurso 2 terá sua primeira utilização na data 30, última utilização na data 84 e será utilizado nos períodos 30-42 e 77-84, sendo necessário 1 unidade em cada um dos períodos. E assim sucessivamente.

Adotando-se como exemplo o recurso 2, o tomador de decisões deverá programar para que tal recurso esteja disponível em 1 unidade na data 30, sendo utilizado até a data 42, e em 1 unidade na data 77, sendo utilizado até a data 84. Não é necessário, portanto, que o recurso 2 esteja disponível antes da data 30, após a data 84 e, entre as datas 43 e 76. Com essa possibilidade de programação, o tomador de decisões evitará a ociosidade e/ou a falta do recurso 2.

7.3. Conclusões

Analisando e comparando todas as soluções dos conjuntos *Ref* obtidos para os dois cenários, conclui-se que a maior disponibilidade de recursos (novo cenário) gera, em média, aproximadamente, 14,73% de redução na duração ($f_1(x)$) e 0,01% de aumento no custo ($f_2(x)$) do projeto representado pelo exemplo. Sendo assim, se o tomador de decisões priorizar o objetivo duração do projeto, torna-se interessante esta maior disponibilização de recursos, visto que esta gera uma redução na duração tendo em contrapartida um pequeno aumento no custo. Caso contrário, se o tomador de decisões priorizar o objetivo custo, esta maior disponibilização de recursos pode deixar de ser interessante, mesmo que o aumento no custo seja pequeno. Daí cabe ao tomador de decisões analisar se vale a pena ou não pagar por este aumento.

Conforme visualizado nos dados das atividades (Tabela 15) e nos custos das soluções obtidas para os dois cenários, pode-se concluir, também, que as atividades 1, 24 e 25 são as que mais contribuem para elevar a duração total e que as atividades 1 e 3 são responsáveis por grande parte do custo total do projeto. Portanto, o tomador de decisões deve dar uma atenção especial a estas atividades, buscando otimizar suas execuções e reduzir os seus custos.

Com base nas datas de início de execução das atividades, obtidas nas soluções escolhidas em cada situação, e de acordo com as demandas das atividades pelos recursos, o tomador de decisões pode programar quando e quanto adquirir/contratar de cada tipo de recurso, evitando, assim, ociosidade e/ou desperdício, bem como a falta dos mesmos.

Vale ressaltar, novamente, que devido à dificuldade de obtenção de dados reais, no exemplo proposto foram utilizados dados definidos aleatoriamente e, portanto, podem estar bem fora da realidade. Esta questão, no entanto, não afetou o entendimento das analises e conclusões feitas acerca da resolução do exemplo. O intuito foi exemplificar a aplicação dos algoritmos propostos.

Capítulo 8

8. CONCLUSÃO E TRABALHOS FUTUROS

O presente trabalho teve como proposta desenvolver métodos metaheurísticos multiobjetivos para a obtenção de conjuntos de soluções não-dominadas para o problema de sequenciamento de atividades em projetos com restrições de recursos e de precedência (PSAPRRP). O PSAPRRP pertence à classe de problemas de otimização combinatória *NP*-difíceis e, é um problema comum a um grande número de situações reais de tomada de decisão, tais como os problemas que surgem no gerenciamento de projetos na construção civil.

O objetivo no desenvolvimento desses métodos foi auxiliar os projetistas da construção metálica no gerenciamento de projetos de obras, tornando-as mais planejadas e controladas. Um correto gerenciamento de projetos pode propiciar a redução de prazos e custos, minimização de riscos e redução de erros no processo produtivo, bem como a melhor utilização dos recursos produtivos.

Para a aplicação dos métodos desenvolvidos, primeiramente, o PSAPRRP foi formulado como um problema de otimização multiobjetivo. Na formulação utilizada foram considerados dois objetivos conflitantes: a minimização da duração (*makespan*) do projeto e a minimização do somatório dos custos de execução das atividades associados às suas datas de início. Buscando a determinação de soluções não-dominadas para o problema assim formulado, foram implementados cinco algoritmos: um GMO (GMO_PSAP), um MOVNS (MOVMNS_PSAP), um GMO utilizando o VNS como busca local (GMOVNS_PSAP), um MOVNS com intensificação, baseado em Ottoni et al. (2011) (MOVNS_I_PSAP), e um PILS (PILS_PSAP).

Devido à dificuldade de obtenção de instâncias multiobjetivos na literatura e/ou dados reais, os algoritmos foram testados utilizando 160 instâncias mono-objetivos encontradas na literatura e adaptadas para o problema multiobjetivo. Os algoritmos propostos se mostraram eficientes na resolução de todas as instâncias utilizadas, apresentando baixo esforço computacional, ou seja, obtiveram os conjuntos de soluções não-dominadas em um tempo computacional aceitável.

A comparação entre os resultados obtidos pelos algoritmos foi feita utilizando-se de quatro métricas de avaliação de desempenho: medidas de distância, diferença de hipervolume, *epsilon* e taxa de erro. Com base nos resultados obtidos através dos experimentos computacionais, conclui-se que o MOVNS_I_PSAP foi superior aos outros, na maioria das instâncias, em relação às medidas de distância, apresentando soluções melhor distribuídas ao longo da fronteira de Pareto de referência. O GMOVNS_PSAP se mostrou superior em relação às métricas diferença de hipervolume, produzindo uma melhor cobertura para a fronteira Pareto-ótima, e *epsilon*, gerando soluções não-dominadas mais próximas das soluções do conjunto de referência. O PILS_PSAP foi superior aos demais em relação à taxa de erro, apresentando um número maior de soluções não-dominadas pertencentes ao conjunto de referência.

Experimentos estatísticos foram realizados utilizando a técnica ANOVA, o método MDS e o teste não-paramétrico de *Kruskal-Wallis* e, com base nos resultados obtidos, pode-se concluir, estatisticamente, que existe diferença significativa entre alguns algoritmos propostos em relação às métricas medidas de distância, *epsilon* e taxa de erro. Os resultados mostraram não haver evidências estatísticas de que os algoritmos difiram em relação à diferença de hipervolume.

Com o intuito de exemplificar a aplicação dos cinco algoritmos, foi proposto um

exemplo fictício e simplificado de um projeto de construção metálica. Para analisar a relação entre a disponibilidade de recursos e os objetivos utilizados, foram usados dois cenários para o exemplo. Com base nos conjuntos de soluções não-dominadas de referência obtidos para os dois cenários analisados, concluiu-se que a maior disponibilidade de recursos gerou, em média, aproximadamente, 14,73% de redução na duração e 0,01% de aumento no custo do projeto. Sendo assim, se o tomador de decisões priorizar o objetivo duração do projeto, torna-se interessante a maior disponibilização de recursos, visto que esta gerou uma redução na duração tendo em contrapartida um pequeno aumento no custo. Caso contrário, se o tomador de decisões priorizar o objetivo custo, a maior disponibilização de recursos pode deixar de ser interessante, mesmo que o aumento no custo seja pequeno. Daí cabe ao tomador de decisões analisar se vale a pena ou não pagar por este aumento.

Com base nos dados das atividades e nos conjuntos de soluções obtidos para os dois cenários do exemplo, foi possível, também, identificar quais atividades necessitam de atenção especial. Identificando-se tais atividades, pode-se buscar a otimização de suas execuções e a redução de seus custos. Além disso, baseando-se nas datas de início de execução das atividades, obtidas na resolução dos dois cenários, e de acordo com as demandas das atividades pelos recursos, o tomador de decisões pode determinar quando e quanto adquirir/contratar cada tipo de recurso, evitando, assim, ociosidade e/ou desperdício, bem como a falta dos mesmos.

Como propostas para trabalhos futuros, outros métodos de geração de soluções iniciais podem ser utilizados, já que boas soluções de partida são importantes para o processo de busca local. Poderia ser utilizado, por exemplo, o MPGS, descrito na seção 3.4.1-*C*, com outras regras de prioridade. Podem ser utilizados, também, outros e/ou mais objetivos, como os descritos na seção 4.3. Outros métodos metaheurísticos e outras métricas de avaliação de desempenho também podem ser utilizados.

REFERÊNCIAS BIBLIOGRÁFICAS

ABBASI, B.; SHADROKH, S.; ARKAT, J. *Bi-objective Resource-constrained Project Scheduling with Robustness and Makespan Criteria.* Applied Mathematics and Computation, v. 180, p. 146-152, 2006.

ADELI, H.; KARIN, A. *Construction Scheduling, Cost Optimization and Management*. Spon Press, New York, 2001.

AGARWAL, A.; COLAK, S.; ERENGUC, S. *A Neurogenetic Approach for the Resource-constrained Project Scheduling Problem.* Computers & Operations Research, v. 38, p. 44-50, 2011.

ALVAREZ-VALDES, R., TAMARIT, J. M. *Heuristic Algorithms for Resource-Constrained Project Scheduling: a Review and an Empirical Analysis*. Advances in Project Scheduling, Amsterdam, p. 113-134, 1989.

ARROYO, J. E. C.; VIEIRA, P. S.; VIANNA, D. S. *A GRASP Algorithm for the Multi-criteria Minimum Spanning Tree Problem.* Annals of Operations Research, v. 159, n.1, p.125-133, 2008.

ARROYO, J. E. C.; OTTINI, R. S.; OLIVEIRA, A. P. *Multi-objective Variable Neighborhood Search Algorithms for a Single Machine Scheduling Problem with Distinct Due Windows*. Electronic Notes in Theoretical Computer Science, v. 281, p. 5-19, 2011.

ARROYO, J. E. C. *Heurísticas e Metaheurísticas para Otimização Combinatória Multiobjetivo*. Tese de Doutorado, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 2002.

ARTIGUES, C.; MICHELON, P.; REUSSER, S. *Insertion Techniques for Static and Dynamic Resource-Constrained Project Scheduling.* European Journal of Operational Research, v. 149, p. 249-267, 2003.

BACKER, K. R. *Introduction to Sequencing and Scheduling*. John Wiley, New York - NY, 1974.

BALAS, E. *Project Scheduling with Resource Constraints*. Operational Research, v. 15, p. 915-957, 1967.

BALLESTÍN, F.; BLANCO, R. *Theoretical and Practical Fundamentals for Multi-objective Optimisation in Resource-constrained Project Scheduling Problems.* Computers & Operations Research, v. 38, p. 51-62, 2011.

BELLEI, I. H.; PINHO, F. O.; PINHO, M. O. *Edifícios de Múltiplos Andares em Aço*. Pini, São Paulo – Brasil, 2008.

BLAZEWICZ, J.; ECKER, K. H.; PESCH, E.; SCHMIDT, G.; WEGLARZ, J. Scheduling Computer and Manufacturing Processes. Springer-Verlag, Berlin, 1996.

BOCTOR, F. F. *Heuristics for Scheduling Projects with Resources Restrictions and Several Resource-Duration Modes.* International Journal of Production Research, v. 31, p. 2547-2558, 1993.

BOZEJKO, W.; WODECKI, M. *Evolutionary Heuristics for Hard Permutational Optimization Problems*. International Journal of Computation, v. 2, p. 151-158, 2005.

BOYCHUK, L. M. e OVCHINNIKOV, V. O. *Principal Methods of Solution Multicriterial Optimization Problems (Survey)*, Soviet Automatic Control, 6:1– 4, 1973.

BRUCKER, P.; KNUST, S.; SCHOO, A.; THIELE, O. *A Branch and Bound Algorithm for the Resource-Constrained Project Scheduling Problem*. European Journal of Operational Research, v. 107, p. 272-288, 1998.

BRUCKER, P.; DREXL, A.; MÖHRING, R.; NEUMANN, K.; PESCH, E. *Resource-constrained Project Scheduling: Notation, Classification, Models, and Methods*.

European Journal of Operational Research, v. 112, p. 3-41, 1999.
BRUCKER, P.; KNUST, S. *A Linear Programming and Constraint Propagation-Based Lower Bound for the RCPSP*. European Journal of Operational Research, v. 127, p. 355-362, 2000.
CAMBIAGHI, H. *Projeto e Obra no Difícil Caminho da Qualidade*. Obra, Planejamento e Construção, v. 37, p. 10-12, 1992.
CARLIER, J.; NÉRON, E. *On Linear Lower Bound for the Resource Constrained Project Scheduling Problem*. European Journal of Operational Research, v. 149, p. 314-324, 2003.
CHISTODOULOU, S. *Construction Imitating Ants: Resource-unconstrained Scheduling with Artificial Ants.* Automation in Construction, v. 18, p. 285-293, 2009.
CHRISTOFIDES N.; ALVAREZ-VALDES R.; TAMARIT J. M. *Project Scheduling with Resource Constraints: a Branch-and-Bound Approach*. European Journal of Operational Research, v. 29, p. 262-273, 1987.
COELLO, C. A.; LAMONT, G. B. *Applications of Multi-objective Evolutionary Algorithms*. World Scientific Printers (S), Pte Ltd, Singapura, 2004.
CZYZAK, P.; JASZKIEWICZ, A. *Pareto Simulated Annealing – a Metaheuristic Technique for Multipleobjective Combinatorial Optimization*. Journal of Multi-Criteria Decision Analysis, v. 7, p. 34–47, 1998.
DAVIS, L. *Job Shop Scheduling with Genetic Algorithms.* 1ª International Conference on Genetic Algorithms, Carne-Mellon University, Pittsburg, PA, USA, 1985.
DAVIS, W. E. *Project Scheduling Under Resources Constraints – Historical Review and Categorization of Procedures*. AIIE Transactions, v. 5, p. 147-163, 1973.
DEB, K.; AGRAWAL, S.; PRATAP, A.; MEYARIVAN, T. *A Fast Elitist Nondominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II.* Springer, Berlin, 2000.
DEB, K. *Multiobjective Optimization using Evolutionary Algorithms*. New York: John Wiley, 2001.
DEB, K.; JAIN, S. *Running Performance Metrics for Evolutionary Multi-objective Algorithms*. 2002.
DEIRANLOU, M.; JOLAI, F. *A New Efficient Genetic Algorithm for Project Scheduling under Resource Constraint*. World Applied Science Journal, v. 7, p. 987-997, 2009.
DEMEULEMEESTER, E. J.; HERROELEN, W. S. *Project Scheduling: a Research Handbook*, Kluwer Academic Publisher, Boston, 2002.
DORIGO, M. Optimization, *Learning and Natural Algorithm*. Tese de Doutorado, Universidade Politécnica de Milão, Itália, 1992.
DORIGO, M.; MANIEZZO, V.; COLORNI, A. *The Ant System: Optimization by a Colony of Cooperating Agents*. IEEE Transaction on Systems, Man and Cybernetics, v. 26, p. 29-41, 1996.
FEO, T. A.; RESENDE, M. G. C. *Greedy Randomized Adaptive Search Procedures*. Journal of Global Optimization, v. 40, p. 109-133, 1995.
FESTA, P.; RESENDE, M. G. C. *An Annotated Bibliography of GRASP, Part II: Applications*. International Transactions in Operational Research, v. 16, p. 131-172, 2009.
FONSECA, C.; KNOWLES, J.; THIELE, L.; ZITZLER, E. *A Tutorial on the Performance Assessment of Stochastic Multi-objective Optimizers*. In Third International Conference on Evolutionary Multi-criterion Optimization (EMO), v. 216, 2005.

GAREY, M. R.; D. S. JONHSON. *Computers and intractability: A guide to the theory of NP-Completeness*. W. H. Freeman and Company, 1979.

GEIGER, M. J. *Foundations of the Pareto Iterated Local Search Metaheuristic*. In: Proceedings of the 18th MCDM, Chania, Greece, June, 2006.

GEIGER, M. J. *Randomized Variable Neighborhood Search for Multiobjective Optimization*. 4th *EU/ME: Design and Evaluation of Advanced Hybrid Meta-Heuristics*, p. 34-42, 2008.

GEYER, P. *Component-oriented Decomposition for Multidisciplinary Design Optimization in Building Design*. Advanced Engineering Informatics, v. 23, p. 12-31, 2009.

GLOVER, F. *Future Paths for Integer Programming and Links to Artificial Intelligence*. Computers and Operations Research, v. 5, p. 533-549, 1986

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Berkeley, 1989.

GRAHAM, R. L.; LAWLER, E. L.; LENSTRA, J. K.; RINNOOY KAN, A. H. G. *Optimization and approximation in deterministic sequencing and scheduling theory: a survey.* Annals of Discrete Mathematics, v. 5, p. 287- 326, 1979.

HAMM, M.; BEIBERT, U.; KÖNIG, M. *Simulation-Based Optimization of Construction Schedules by Using Pareto Simulated Annealing*. 18th International Conference on the Application of Computer. Science and Mathematics in Architecture and Civil Engineering. Weimar, Alemanha, 2009.

HANSEN, M. P. *Tabu Search for Multiobjective Optimization: MOTS*. In: 13th INTERNATIONAL CONFERENCE ON MULTIPLE CRITERIA DECISION MAKING, University of Cape Town, p. 6-10, January, 1997.

HANSEN, M.; JASZKIEWIEZ, A. *Evaluating the Quality of Approximations to the Non-dominated Set*. IMM, Department of Mathematical Modeling, Technical University of Denmark, 1998.

HAPKE, M.; JASZKIEWICZ, A.; SLOWINSKI, R. *Interactive Analysis of Multiple-criteria Project Scheduling Problems.* European Journal of Operational Research, v. 107, p. 315-324, 1998.

HENDRICKSON, C. *Project Management for Construction: Fundamental Concepts for Owners, Engineers, Architects and Builders*. Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA l52l3. First Edition originally printed by Prentice Hall, ISBN 0-13-731266-0, 1989 with co-author Tung Au. Second Edition prepared for world wide web publication in 2000. Version 2.2 prepared Summer, 2008. Disponível em < http://pmbook.ce.cmu.edu/>. Data de acesso: 16/07/2011.

HWANG, C. L; MASUD, A. S. *Multiobjective Decision Maker, Methods and Applications*. A State of the Art Survey. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, Berlin, 1979.

INABA, R. *Construções Metálicas: O uso do Aço na Construção Civil*. (2009) Disponível em <http://www.metalica.com.br/construcoes-metalicas-o-uso-do-aco-na-construcao-civil>. Data de acesso: 20/07/2011.

ISHIDA, C. Y., CARVALHO, A. B., POZO, A. T. R., GOLDBARG, E. F. G. AND GOLDBARG, M. C. *Exploring Multiobjective PSO and GRASP-PR for Rule Induction*, Eighth European Conference on Evolutionary Computation in Combinatorial Optimisation, Springer, Berlin, v. 1, p. 73-84, 2008.

JASKOWSKI, P.; SOBOTKA, A. *Multicriteria Construction Project Scheduling Method Using Evolutionary Algorithm*. Operational Research, v. 6, p. 283-297, 2006.

JONES, D. F.; MIRRAZAVI, S.K.; TAMIZ, M. *Multiobjective Metaheuristics: an Overview of the Current State-of-art*. European Journal of Operational Research, v. 137, p. 1-19, 2002.

KAZEMI, F. S.; TAVAKKOLI-MOGHADDAN, R. *Solving a Multi-objective Multi-mode Resource-constrained Project Scheduling with Discounted Cash flows*. 6th International Management Conference, Tehran – Iran, 2008.

KEELLING, R. *Gestão de Projetos – Uma Abordagem Global*. Saraiva, São Paulo, 2002.

KELLEY, J. E. *The Critical Method: Resources Planning and Scheduling.* Industrial Scheduling. Prentice-Hall, p. 347-365. New Jersey, 1963.

KIRKPATRICK, S.; GELLAT, D. C.; VECCHI, M. P. (1983) *Optimization by Simulated Annealing*. Science, v. 220, p. 671-680.

KLEIN, R. *Project Scheduling with Time-varying Resources Constraint.* International Journal of Production Research, v. 38, p. 3937-3952, 2000.

KOLISCH, R. *Serial and Parallel Resource-Constrained Project Scheduling Methods Revisited: Theory and Computation.* European Journal of Operational Research, v. 90, p. 320-333, 1996.

KOLISCH, R.; SPRECHER, A. *PSPLIB – A Project Scheduling Problem Library*. European Journal of Operational Research, v. 96, p. 205-216, 1996.

KONÉ, O.; ARTIGUES, C.; LOPEZ, P.; MONGEAU, M. *Event-Based MILP Models for Resource-Constrained Project Scheduling Problems*. Computers & Operations Research, v. 38, p. 3-13, 2009.

KÖNIG, M.; BEIBET, U. *Construction Scheduling Optimization by Simulated Annealing*. 26th International Symposium on Automation and Robotics in Construction. Austin, Texas, Estados Unidos, p. 183-190, 24 a 27 de junho de 2009.

KULINSKAYA, E.; STAUDTE, R. G.; GAO, A. *Power Approximations in Testing for Unequal Means in a One-way ANOVA Weighted for Unequal Variances*. Communication in Statistics, v. 32 (12), p. 2353-2371, 2003.

LIANG, Y. C.; CHEN, H. L.; TIEN, C. Y. *Variable Neighborhood Search for Multiple-objective Parallel Machine Scheduling Problems*. Proceeding of the 8th International conference on Information and Management Science - China, p. 519-522, 2009.

LIANG, Y. C.; LO, M. H. *Multiple-objective Redundancy Allocation Optimization using a Variable Neighborhood Search Algorithm*. Journal of Heuristics, v. 16, p. 511-535, 2010.

LIU, S.; WANG, C. *Optimization Model for Resource Assignment Problems of Linear Construction Scheduling Projects*. Automation in Construction, v. 16, p. 460-473, 2007.

LOURENÇO, H.; MARTIN, O.; STÜZLE, T. *Iterated Local Search*. Handbook of Metaheuristics, p. 320-353, 2002.

MACIEL, L. L.; MELHADO, S. B. *Qualidade na Construção Civil: Fundamentos*. EPUSP, Texto Técnico, São Paulo, 1996.

MARTÍNEZ-IRANO, M.; HERRERO, J. M.; SANCHIS, J.; BLASCO, X. e GARCIA-NIETO, S. *Applied Pareto Multiobjective Optimization by Stochastic Solvers*. Engeneering Applications of Artificial Intelligence, v. 22, p. 455-465, 2009.

MERKLE, D.; MIDDENDORF, M,; SCHEMECK, H. *Ant Colony Optimization for Resource-constrained Project Scheduling*. IEEE Transactions on Evolutionary Computation, v. 6, p. 333-346, 2002.

MLADENOVIC, N.; HANSEN, P. *Variable Neighborhood Search*, Computers and Operations Research, v. 24, p. 1097-1100, 1997.

MONTGOMERY, D. C. *Design and Analysis of Experiments.* John Wiley & Sons, 7th Edition, New York, 2009.

MONTGOMERY, D. C.; RUNGER, C. R. *Estatística Aplicada e Probabilidade para Engenheiros*, 4ª ed., LTC, Rio de Janeiro, 2009.

NBR 8800 – *Projeto de Estruturas de Aço e de Estruturas Mistas de Aço e Concreto de Edifícios*. ABNT, 2008.

ODEDAIRO, B. O., OLADOKUN, V. *Relevance and Applicability of Multi-objective Resource Constrained Project Scheduling Problem*. ETASR – Engineering, Technology & Applied Science Research, v. 1, nº 6, p. 144-150, 2011.

OGUZ, O.; BALA, H. *A Comparative Study of Computational Procedures for the Resource Constrained Project Scheduling Problem*. European Journal of Operational Research, v. 72, p. 406–416, 1994.

OTTONI, R. S.; ARROYO, J. E. C.; SANTOS, A. G. *Algoritmo VNS Multiobjetivo para um Problema de Programação de Tarefas em uma Máquina com Janelas de Entrega.* XLIII Simpósio Brasileiro de Pesquisa Operacional, Ubatuba - São Paulo, v. 1, p.1-12, 2011.

OSYCZAKA, A. *Multicriterion Optimization in Engineering with FORTRAM Programs*. England: Ellis Horwood Ltd., 1984.

PAIVA, A. E. C.; ARROYO, J. E. C., OTTONI, R. S. *Uma Heurística VNS Multiobjetivo para o Problema de Sequenciamento de Tarefas em uma Máquina com Penalidades por Antecipação e Atraso e Fluxo Total.* XXX Encontro Nacional de Engenharia de Produção, São Carlos – São Paulo, v. 1, p. 1-12, 2010.

PARETO, V. *Cours D'Economie Politique*, volume 1, F. Rouge, 1896.

PAULA, M. R. de. *Heurísticas para a Minimização dos Atrasos em Sequenciamento de Máquinas Paralelas com Tempos de Preparação Dependentes da Sequência*. Dissertação de Mestrado, Programa de Pós-graduação em Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, 2008.

PICCHI, F. *Sistemas da Qualidade: Uso em Empresas de Construção de Edifícios*. Tese de Doutorado – Escola Politécnica, Universidade de São Paulo, São Paulo, 1993.

PRITSKER A.; WATTERS L.; WOLFE P. *Multi-project Scheduling with Limited Resources: a zero-one Programming Approach*. Management Science, v. 16, p. 93-108, 1969.

REEVES, C. R. *Genetic Algorithms*. In Reeves, C. R., editor, *Modern Heuristics Techniques for Combinatorial Problems*. Advanced Topics in Computer Science Series, chapter 4, pages 151-196. Blackwell Scientific Publications, 1983.

REYNOLDS, A. P.; IGLESIA, B. *A Multiobjective GRASP for Partial Classification*. Springer-Verlag, 2008.

ROGALSKA, M.; BOZEJKO, W.; HEJDUCKI, Z. *Time/Cost Optimization Using Hybrid Evolutionary Algorithm in Construction Project Scheduling*. Automation in Construction, v. 18, p. 24-31, 2008.

SALUKVADZE, M. E. *On the Existence of Solution in Problems of Optimization under Vector Valued Criteria*. Journal of Optimization Theory and Applications, 12(2):203–217, 1974.

SERAFINI, P. *Simulated Annealing for Multiobjective Optimization Problems*. Proceedings of the 10ª International Conference on MCDM, Taipei, 19 a 24 de Julio de 1992, p. 221-248.

SLOWINSKI, R. *Multiobjective Project Scheduling Under Multiple-category Resource Constraints.* Advances in Project Scheduling, Elsevier, Amsterdam, 1989.

SLOWINSKI, R. *Multiobjective Network Scheduling with Efficient Use of Renewable and Nonrenewable Resources*. European Journal of Operational Research, v. 7, p. 265-273, 1981.

STEUER, R. E. *Multiple Criteria Optimization: Theory, Computation, and Optimization*. Wiley, New York, 1986.

THOMAS, P. R.; SALHI, S. *A Tabu Search Approach for the Resource Constrained Project Scheduling Problem*. Journal of Heuristics, v. 4, p. 123-139, 1998.

TSENG, L.; CHEN, S. *A Hybrid Metaheuristic for the Resource-constrined Project Scheduling Problem*. European Journal of Operational Research, v. 175, p. 707-721, 2006.

ULUNGU, E.L., TEGHEM, J.; OST, C. *Efficiency of Interactive Multi-objective Simulated Annealing Through a Case Study*. Journal of the Operational Research Society, v.49, p. 1044-1050, 1998.

VELDHUIZEN, D. *Multi-objective Evolutionary Algorithms: Classifications, Analyses and New Innovations*. PhD Thesis, Faculty of the Graduate School of Engineering of the Air Force Institute of Technology, 1999.

VIANA, A,; SOUSA, J. P. *Using Metaheuristics in Multiobjective Resource Constrained Project Scheduling*. European Journal of Operational Research, v. 120, p. 359-374, 2000.

VIANNA, D. S.; ARROYO, J. E. C. *A GRASP Algorithm for the Multi-objective Knapsack Problem*. Annals of the XXIV International Conference of the Chilean Computer Science Society, p. 69-75, 2004.

ZITZLER, E.; LAUMANNS, M.; THIELE, L. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm.* Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, 2001.

ZITZLER, E.; DEB, K.; THIELE, L. *Comparison of Multi-objective Evolutionary Algorithms: Empirical Results*. Evolutionary Computation, v. 8, p. 173-195, 2000.

ZITZLER, E.; THIELE, L. *Multi-objective Optimization Using Evolutionary Algorithms – a Comparative Case Study*. Springer, 1998.