COPPE/UFRJ

PROGRAMAÇÃO DE PROJETOS SOB RESTRIÇÕES DE RECURSOS E INCERTEZAS COM UTILIZAÇÃO DE LÓGICA *FUZZY*

Humberto Henriques de Arruda

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Produção, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Produção.

Orientador: Virgílio José Martins Ferreira Filho

Rio de Janeiro

Dezembro de 2009

PROGRAMAÇÃO DE PROJETOS SOB RESTRIÇÕES DE RECURSOS E INCERTEZAS COM UTILIZAÇÃO DE LÓGICA FUZZY

Humberto Henriques de Arruda

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE PRODUÇÃO.

Aprovada por:

_____
Prof. Virgílio José Martins Ferreira Filho, D.Sc.

_____
Prof. Laura Silvia Bahiense da Silva Leite, D.Sc.

_____
Prof. Leonardo Junqueira Lustosa, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
DEZEMBRO DE 2009

Arruda, Humberto Henriques de

Programação de projetos sob restrições de recursos e incertezas com utilização de lógica *fuzzy*/ Humberto Henriques de Arruda. – Rio de Janeiro: UFRJ/COPPE, 2009.

IX, 97 p.: il.; 29,7 cm.

Orientador: Virgílio José Martins Ferreira Filho

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia de Produção, 2009.

Referências Bibliográficas: p. 70-74.

1. *Scheduling* de projetos. 2. Incertezas. 3. Lógica *fuzzy*. I. Ferreira Filho, Virgílio José Martins. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Produção. III. Título.

Agradecimentos

A Deus, pela vida, pela saúde e por ter concedido a graça de chegar até aqui;

À minha mãe Maria José Henriques, por todo amor, apoio e confiança em mim depositados;

Aos meus familiares e amigos, pelo apoio e pela compreensão nos momentos de ausência;

Ao professor Virgílio José Martins Ferreira Filho pelos ensinamentos acadêmicos, pela orientação concedida e tempo dedicado ao acompanhamento deste trabalho;

Aos professores Laura Silvia Bahiense da Silva Leite e Leonardo Junqueira Lustosa pelos ensinamentos acadêmicos e por aceitarem participar da banca examinadora;

Aos meus superiores hierárquicos pela flexibilidade concedida para a realização deste mestrado;

Aos colegas de turma, pela troca saudável de informações, bem como pela ajuda oferecida durante a elaboração deste trabalho;

Ao pessoal de apoio do Programa de Engenharia de Produção, especialmente Pedro e Andréia, pela ajuda ao longo do curso;

À Indústria de Material Bélico do Brasil – IMBEL – pela oportunidade concedida para a realização deste mestrado.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

PROGRAMAÇÃO DE PROJETOS SOB RESTRIÇÕES DE RECURSOS E INCERTEZAS COM UTILIZAÇÃO DE LÓGICA *FUZZY*

Humberto Henriques de Arruda

Dezembro/2009

Orientador: Virgílio José Martins Ferreira Filho

Programa: Engenharia de Produção

O problema de programação de projetos com restrições de recursos (*Resource Constrained Project Scheduling Problem* - RCPSP) é um dos problemas mais importantes na área de programação de projetos. A limitação real de recursos é determinante no momento do planejamento. Além da complexidade computacional do problema em situações reais, existe a dificuldade de se estimar tempo e recursos necessários para a realização das atividades de um projeto. Sendo assim, é necessário o tratamento das incertezas no planejamento, como durações das atividades e a criação de uma margem de segurança no tempo total do projeto (*makespan*). Com o intuito de considerar as incertezas no RCPSP, foram utilizadas técnicas de solução baseadas em lógica *fuzzy*, mais especificamente, com abordagem das incertezas na duração das atividades. Para lidar com as variações no tempo de duração de cada atividade, foi proposto um *buffer* de projeto. Foram apresentados quatro métodos de solução, sendo dois destes baseados nas seguintes metaheurísticas: algoritmos genéticos e otimização por enxame de partículas (PSO). Para análise de desempenho dos quatro métodos propostos, foram testadas doze instâncias de projetos, com trinta, sessenta, noventa e cento e vinte atividades. Nos testes relativos aos métodos baseados em metaheurísticas, foram consideradas diferentes configurações de parâmetros, em busca da combinação que fornecesse os melhores resultados.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

RESOURCE CONSTRAINED PROJECT SCHEDULING UNDER UNCERTAINTY WITH FUZZY LOGIC

Humberto Henriques de Arruda

December/2009

Advisor: Virgílio José Martins Ferreira Filho

Department: Industrial Engineering

The Resource Constrained Project Scheduling Problem (RCPSP) is a very important problem in the project scheduling field. The real limitation of resources is a key point at the planning time. Besides the computational complexity of the problem about real situations, there is the difficulty to estimate time and the necessary resources to run all the project activities. For this reason, it is essential to handle uncertainties during the planning phase, such as, the length of all sorts of tasks and the establishment of a safety margin at the total time of the project (*makespan*).

In order to consider some uncertainties in the resource constrained project scheduling problem, solution techniques were employed based on fuzzy logic in a special manner like dealing with the uncertainties about the length of tasks, so a project buffer was proposed. Four solution methods were presented, two of them based on the following metaheuristics: genetic algorithms and particle swarm optimization. Twelve instances of project, with thirty, sixty, ninety and one hundred and twenty tasks, were used to perform tests and compare results from the four solution methods. During the tests of metaheuristics-based solution methods some different parameters setups were considered and by searching the setup could provide best results.

1. INTRODUÇÃO

Atualmente, com o crescimento da disseminação da informação através da internet e dos meios de comunicação em geral, pode-se dizer que vivemos em um ambiente mais dinâmico e globalizado do que há duas ou três décadas. Por conta do aumento da velocidade com que as mudanças ocorrem, as organizações precisam se manter competitivas e têm se preocupado em aumentar a produtividade e o aproveitamento de recursos. Com base nesta procura por melhorias, muitos estudos com foco em gerenciamento de projetos e gestão de recursos vêm sendo publicados.

Segundo o PMBOK (Project Management Body of Knowledge) PMI (2000), as organizações executam trabalho. Este, por sua vez, é composto por serviços continuados e projetos, podendo haver superposição entre eles. As semelhanças entre serviços e projetos estão a seguir:

i. Ambos são executados por pessoas;
ii. Ambos são restringidos por recursos limitados
iii. Ambos devem ser planejados, executados e controlados.

Os serviços continuados e os projetos diferem porque estes são temporários e únicos, enquanto que aqueles são repetitivos e contínuos. Assim, um projeto pode ser definido como um empreendimento temporário, com o objetivo de criar um produto ou um serviço único, PMI (2000).

Analisando a definição de projeto, o termo "temporário" significa que o projeto tem datas de início e fim definidas. O projeto pode ser encerrado quando alcança seus objetivos ou quando é avaliado que não será mais possível atingi-los. Entretanto, o projeto não precisa ter curta duração. O importante é frisar que a duração é finita.

Ainda explorando a definição de projeto, ao citar a criação de um produto ou serviço único, cabe lembrar que, embora já exista uma infinidade de casas ou edifícios, cada nova unidade é um produto único, com determinado proprietário, local e construtora, por exemplo.

Após o projeto ser concebido, a primeira fase é o planejamento. Durante esta fase, é analisada a viabilidade econômica e a disponibilidade de recursos deste projeto. É comum a suspensão ou a extinção de alguns projetos nesta fase inicial, devido à falta de recursos para sua execução. Um dos componentes importantes nesta análise do planejamento é o cronograma, com os recursos utilizados em cada etapa, que é chamado de programação do projeto, ou simplesmente *schedule*.

O *schedule* consiste na seqüência das atividades a serem executadas, além da utilização dos recursos em cada uma dessas atividades.

A literatura de Pesquisa Operacional é mais ampla no estudo de problemas com *scheduling* (programação) de máquinas, onde as durações das atividades são conhecidas e constantes. Entretanto, ao tratar do assunto de programação de projetos, deve-se levar em consideração o tratamento de incertezas, que podem estar relacionadas, mais freqüentemente, aos recursos necessários ou às durações das atividades. Neste caso, existem muitos estudos publicados, com modelos e técnicas de resolução propostas. Uma apresentação das técnicas de modelagem e solução é feita no capítulo 3.

Nos últimos anos, pode-se observar o aumento do número de projetos de tecnologia da informação, tanto nos países desenvolvidos como nos países em desenvolvimento. Estes projetos podem ser predominantemente lógicos, como no caso dos sistemas computacionais, ou com uma parte física considerável, como em equipamentos de comunicação e de informática.

Um aspecto observado nos projetos na área de tecnologia da informação é a ausência de uma base confiável de dados históricos, uma vez que projetos nesta área raramente possuem similares anteriores, como no caso da construção civil ou na construção de instalações para exploração de petróleo. Quando há uma base histórica confiável, o responsável pela criação da programação base do projeto pode utilizar técnicas estocásticas para determinar as durações das atividades. Porém, esta não é a situação mais comum nas empresas de tecnologia.

Por tudo o que foi apresentado, o presente trabalho se propõe a comparar técnicas para programação de projetos utilizando a abordagem possibilística (lógica *fuzzy*), em vez da abordagem probabilística (que depende de dados históricos). É importante ressaltar que, para maior confiabilidade na programação possibilística, é necessário o conhecimento de profissionais experientes, que possam determinar, *a priori*, as possíveis durações de determinada atividade e seus recursos necessários.

O objetivo geral desta dissertação é discutir métodos de solução para o problema de programação de projetos com restrições de recursos e incertezas, descrevendo variações no tratamento das incertezas. Os objetivos específicos são apresentar algumas técnicas presentes na literatura acadêmica, executar testes com uma coletânea de instâncias, comparar e analisar os resultados encontrados por estas diferentes metodologias.

O presente estudo está dividido em oito capítulos e dois apêndices. Este capítulo de introdução apresenta os objetivos e as causas de motivação para sua realização.

No capítulo 2, é apresentado o problema de programação de projetos com restrições de recursos e incertezas. Inicialmente, são feitas algumas considerações sobre a teoria de *scheduling* de máquinas, uma vez que existem semelhanças entre a programação exata de máquinas e a programação de projetos com incertezas.

O capítulo 3 mostra uma revisão sobre as técnicas para o tratamento das incertezas e possíveis métodos para a solução do problema. São citadas as diferentes abordagens publicadas, como, por exemplo, a utilização de lógica *fuzzy* e de metaheurísticas, que são o foco deste estudo.

O capítulo 4 contém o detalhamento das duas metaheurísticas que constituem grande parte do presente estudo: algoritmos genéticos e otimização por enxame de partículas. É feita uma apresentação teórica sobre os conceitos e os detalhes que compõem estas metaheurísticas.

No capítulo 5, o problema a ser resolvido é definido, com a devida formulação matemática e formalização dos dados de entrada e das variáveis que devem ser encontrados. Além disso, apresenta o algoritmo de seqüenciamento de atividades, que é uma importante peça em todo este estudo. É o algoritmo que fornece, a partir das durações, das relações de precedência, dos recursos necessários e da quantidade de recursos disponíveis, a programação das atividades, com seus respectivos instantes de início e fim. Este algoritmo é chamado repetidamente em todas as rotinas implementadas neste estudo.

Ainda no capítulo 5, são apresentados dois métodos baseados em lógica *fuzzy* FPS-G4 e FPS-SIM, são detalhadas também duas abordagens metaheurísticas, baseadas em um algoritmo genético e um algoritmo de otimização por enxame de partículas (*Particle Swarm Optimization* – PSO). As atividades são modeladas também como números *fuzzy* e, a partir da solução encontrada pelos algoritmos metaheurísticos, é determinado um buffer de projeto destinado a lidar com as incertezas.

No capítulo 6, são descritos os experimentos realizados, a partir de projetos de 30, 60, 90 e 120 atividades. São mostrados a modelagem de recursos, as configurações de parâmetros e os resultados computacionais, além dos gráficos e das análises destes.

As conclusões e sugestões de trabalhos futuros estão no capítulo 7.

O capítulo 8 contém as referências bibliográficas.

O APÊNDICE A é composto por um estudo mais aprofundado sobre a programação *fuzzy* de projetos, com maior detalhamento matemático na comparação de conjuntos *fuzzy*.

O APÊNDICE B traz todas as instâncias de projetos com sessenta, noventa e cento e vinte atividades que foram utilizadas nos testes computacionais.

2. O PROBLEMA DE PROGRAMAÇÃO DE PROJETOS COM RESTRIÇÕES DE RECURSOS E INCERTEZAS

2.1 INTRODUÇÃO

O desenvolvimento do presente trabalho foi baseado na missão de encontrar uma solução (programação ou Schedule) que minimize o tempo total de um projeto, respeitando as relações de precedência entre as atividades e as restrições de recursos disponíveis. Este problema é conhecido na literatura acadêmica como *Resource Constrained Project Schedule Problem*, ou simplesmente RCPSP.

Entretanto, a modelagem teórica e algumas técnicas de solução do RCPSP são originadas da teoria de *Scheduling* de máquinas. Sendo assim, cabe uma breve explanação sobre a teoria de *Scheduling* aplicada a fábricas, onde a intenção é otimizar a utilização das máquinas na planta fabril.

Segundo Herroelen e Demeulemeester (1994), tanto os problemas de *scheduling* como os de seqüenciamento são referentes à alocação ótima de recursos ao longo do tempo. O *scheduling* está focado na determinação dos instantes em que as atividades começam a ser desenvolvidas, enquanto que o seqüenciamento se preocupa em determinar o ordenamento dessas atividades. No contexto de *scheduling* determinístico de máquinas, o recurso é uma máquina para processar as tarefas.

Usualmente, na literatura em geral, as atividades são chamadas de *jobs*. Existem dois problemas clássicos do *scheduling* de máquinas: *job-shop* e *flow-shop*. No *job-shop*, não existem relações tecnológicas de precedência. No *flow-shop*, o ordenamento de todos os *jobs* é o mesmo.

Para avaliar o desempenho de determinada programação, podem ser utilizados diferentes critérios. Os mais comuns são: minimização do tempo máximo de fluxo (F_{max}), minimização do tempo total (C_{max} ou *makespan*), minimização do tempo médio de fluxo ($\overline{F}$) e minimização do tempo total médio ($\overline{C}$). Podem ainda ser usados critérios baseados em datas de entrega ou tempo de espera das atividades. Para mais detalhes sobre as medidas de desempenho, recomenda-se o capítulo 1 de Simon French e D.Phil (1982).

2.2 ALGUMAS CONSIDERAÇÕES SOBRE A TEORIA DE *SCHEDULING*

Como já foi apresentado, existem muitas semelhanças entre os problemas de programação de projetos e os problemas de programação de máquinas. A seguir, é apresentado o método de classificação de problemas de programação de máquinas, uma vez que a literatura acadêmica nesta área é consideravelmente mais abrangente que na área de programação de projetos.

Seguindo a metodologia proposta em Conway et al. (1967) e usada também por Simon French e D.Phil (1982), os problemas de programação de máquinas podem ser classificados por quatro parâmetros: n/m/A/B.

- n → número de *jobs*;
- m → número de máquinas;
- A → descreve o padrão de fluxo. Se m = 1, A é omitido, uma vez que a máquina só pode processar um *job* por vez. Os possíveis casos de A são: **F**, para *flow-shop* clássico; P, para permutações do caso clássico de *flow-shop* (também chamado de randômico, ou R, como citado em Vasconcellos (2007)); e G, para o caso em que não há restrições tecnológicas;
- B → indica o tipo de medida de desempenho utilizado.

Por exemplo, o rótulo n/3/F/C_{max} representa o caso com n *jobs*, três máquinas, problema *flow-shop* tendo como medida de desempenho a duração total do projeto.

2.3 PROGRAMAÇÃO DE PROJETOS (*PROJECT SCHEDULING*)

O problema de programação de projetos consiste em determinar o *schedule* de alocação de recursos, respeitando as restrições tecnológicas e atingindo alguma medida de desempenho. Nas situações reais, geralmente haverá incertezas na programação dos projetos, devido à falta de precisão das durações das atividades do projeto Ke e Liu (2005).

O tema de programação de projetos é um dos mais estudados nos últimos anos devido à sua demanda prática. Podemos verificar que os modelos propostos ultimamente têm se tornado cada vez mais realistas. Existem diversas variações de abordagens propostas pelos diferentes autores que têm se dedicado ao estudo do tema.

Para que seja feita uma comparação entre as alternativas de decisão de um projeto, podem ser utilizados vários critérios. Segundo Vasconcellos (2007), os critérios mais comuns e que possuem modelos estudados são:

1. duração total;
2. custo total;
3. relação custo x benefício;
4. medida do risco do projeto;
5. valor presente líquido.

Cabe ressaltar que, qualquer que seja o critério selecionado, existem restrições sobre o uso dos recursos (financeiros ou de pessoal) disponíveis. Além disso, para que o planejamento do projeto possa ser feito, é essencial que as durações das atividades sejam determinadas ou, no mínimo, bem estimadas. A modelagem das entidades do projeto será feita a seguir.

2.3.1 ATIVIDADES

O projeto pode ser representado como uma rede, com um conjunto discreto e finito de atividades, representando as relações de precedência de acordo com uma das opções a seguir:

- Atividade no arco (*Activity on Arc* – AOA) – onde cada arco descreve uma atividade e cada nó representa o instante de início ou término das atividades;
- Atividade no nó (*Activity on Node* – AON) – onde cada nó representa a atividade e o arco entre dois nós descreve a precedência entre as atividades associadas a tais nós.

A Figura 1, de Ke e Liu (2005), ilustra a representação AON:

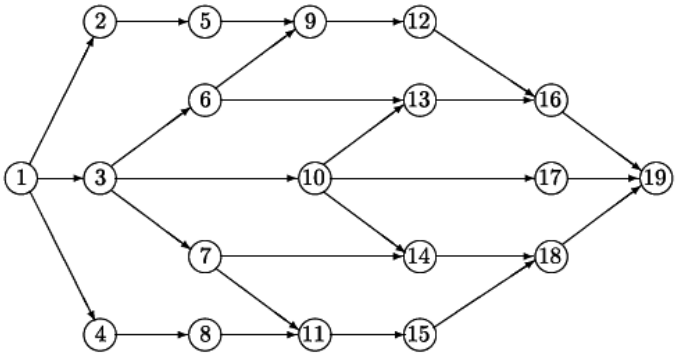


Figura 1 - Projeto

Nesta figura, o nó 1 representa o início do projeto, enquanto o nó 19 representa sua finalização. Os nós intermediários representam as atividades necessárias para o desenvolvimento do projeto. No desenvolvimento do presente estudo, foi utilizada a modelagem AON.

2.3.2 DURAÇÕES

Como o planejamento do projeto depende diretamente das durações das atividades, é necessário que elas estejam bem modeladas. Inicialmente, na década de 1950, surgiram duas metodologias importantes para a determinação da duração global de um projeto: o Critical Path Method (CPM) e o Program Evaluation and Review Technique (PERT). Nas duas metodologias, o que é feito é uma estimativa da duração das atividades, sendo possível, então, com as relações de precedência atendidas, a determinação do tempo total do projeto. Usando o CPM, somente o valor mais provável é usado para a definição do tempo total do projeto. Na técnica PERT, para cada atividade, são estimados três valores de duração: o otimista, o mais provável e o pessimista. Após isso, o valor utilizado para a determinação do tempo total do projeto é dado por:

$$\text{Valor\_estimado} = \frac{\text{prev\_otimista} + 4*\text{prev\_maisProvável} + \text{prev\_pessimista}}{6}$$

.Obviamente, pode-se concluir que a técnica PERT fornece uma perspectiva mais realista do projeto, com a introdução de uma medida da incerteza na estimativa das durações.

Também deve ser observada a possibilidade de preempção, que significa que determinado recurso pode pausar a realização de uma atividade para o início de outra atividade. Naturalmente, a possibilidade de haver preempção traz mais flexibilidade e realismo ao planejamento do projeto.

Durante o ano de 1959, foram publicados dois estudos que disseminaram as técnicas citadas anteriormente: Malcom et al. (1959) para o PERT e Kelley e Walker (1959) para o CPM. A partir deste ano, o problema de programação de projetos passou a receber maior atenção dos pesquisadores ao redor do mundo. Porém, embora sejam abordagens muito empregadas, existem simplificações que distanciam o modelo teórico do que acontece na prática. Um grande problema é a hipótese que os recursos são ilimitados, o que não coincide com a realidade. Assim, houve a necessidade de adaptações para que o modelo teórico se aproximasse da realidade.

Embora a técnica PERT já insira uma medida de incertezas nas durações das atividades, ainda o faz de forma muito simplista, gerando a necessidade de modelar de forma mais próxima da realidade estas incertezas. Dois tipos de modelagem de durações estão disponíveis na literatura: a modelagem probabilística (mais usual) e a modelagem possibilística (que é o objeto de interesse deste estudo).

2.3.2.1 MODELAGEM PROBABILÍSTICA DAS DURAÇÕES

Uma das formas mais utilizadas para se tratar as incertezas na determinação das durações das atividades é a modelagem baseada em probabilidades. Normalmente, as durações das atividades de um projeto são assumidas como estocásticas.

Ke e Liu (2005) classificaram os modelos de problemas estocásticos de programação de projetos como a seguir:

- Modelo de custo esperado:

  É amplamente usado para resolver vários tipos de problemas práticos. Possibilita a tomada de decisão baseada na minimização do custo esperado do projeto, sujeito a restrições em que o tempo total esperado obedeça à data de entrega do projeto.

- Modelo α-custo

  O conceito de *Chance-constrained Problem* (CCP) foi introduzido por Charnes e Cooper (1959), com objetivos de satisfazer as restrições de chance com, pelo menos, alguns níveis de confiança estabelecidos. Ke e Liu (2005) formularam o problema de programação de projetos, estabelecendo um modelo α-custo baseado em CCP estocástica.

- Modelo de maximização da probabilidade

  O objetivo deste modelo é maximizar a probabilidade de que o custo total do projeto não ultrapasse o orçamento previsto. Além disso, existem restrições para que a probabilidade de o tempo total do projeto não ultrapassar a data de entrega seja maior ou igual a determinado nível de confiança α.

2.3.2.2 MODELAGEM POSSIBILÍSTICA DAS DURAÇÕES

Em muitos casos, devido a pouca disponibilidade ou ausência de dados históricos confiáveis, o uso de modelos probabilísticos pode não ser indicado ou ser pouco eficiente. Uma alternativa para este problema é o uso de modelos possibilísticos, onde um profissional experiente no assunto faz projeções sobre as durações das atividades. Pode ser citado como exemplo em que os dados históricos, em geral, não são confiáveis, o desenvolvimento de software, onde o tempo necessário para a execução de determinada tarefa muitas vezes é bem diferente do tempo planejado devido à necessidade de correções no código gerado, testes e integração com as outras partes do projeto após as mudanças no código. As tarefas relacionadas à estrutura do projeto podem levar a incertezas ainda maiores que as tarefas de codificação, uma vez que podem impactar toda a vida do projeto.

Além dos aspectos citados, a modelagem probabilística pode ser menos apropriada devido à dificuldade de se encontrar uma distribuição de probabilidade para erros humanos, que usualmente acontecem ao longo dos projetos. Então, a modelagem baseada em possibilidades e com formulação matemática, pode ser mais apropriada. Neste caso, segundo Ozdamar e Alanya (2001), a utilização de lógica *fuzzy* é indicada, porque ela lida com a imprecisão de forma semelhante à utilizada pelos seres humanos, além de fornecer intervalos de fácil manipulação para as durações do projeto, o que permite que vários cenários sejam apresentados e torna o planejamento mais adequado.

A utilização da lógica *fuzzy* permite que a representação das durações das atividades seja feita com menos dependência do passado. A lógica *fuzzy* relaciona uma função de pertinência do elemento ao conjunto. Sendo assim, esta função pode ser dada pelo grau de pertinência de certa duração ao conjunto de valores possíveis para a duração da atividade. Como citado por Zimmermann (1991), um método popular é identificar a pertinência em uma escala de porcentagem dado um conjunto de durações. Quanto mais alto o grau de pertinência, mais o valor de duração pertence ao conjunto de durações possíveis.

Segundo Dubois e Prade (1987), uma representação bastante difundida para números fuzzy é dada por números LR. O uso de números *fuzzy* desta forma é incentivado por facilitar a manipulação matemática e a aquisição de informações dos profissionais responsáveis pelo desenvolvimento.

Um conjunto *fuzzy* é uma função que mede o grau de pertinência a um conjunto. O conjunto A no conjunto base X pode ser descrito como uma função de pertinência μ_A: $X \to \{0,1\}$ com $\mu_A(x) = 1$ se $x \in A$ e $\mu_A(x) = 0$ se $x \notin A$. Podemos estender o modelo citado para que a função mapeie no intervalo [0,1]. Neste modelo, um alto valor na função de pertinência significa grande possibilidade, enquanto que um baixo valor na função de pertinência significa uma pequena possibilidade. Sendo assim, após os valores da função de pertinência serem escritos para todos os elementos do conjunto, pode-se formar um conjunto de pares ordenados $\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$, onde $\mu_{\tilde{A}}(x)$, $0 \le \mu_{\tilde{A}}(x) \le 1$, é chamada de função ou grau de pertinência de *x* em $\tilde{A}$.

Segundo Hapke et al. (1999), a forma precisa de um número *fuzzy* é difícil de ser descrita apenas por um profissional experiente. Rommelfanger (1990) propôs uma maneira prática para definir funções de pertinência adequadas. Ele sugeriu que o profissional experiente expressasse suas previsões mais pessimistas e mais otimistas sobre determinado parâmetro de incerteza, especificando níveis de pertinência em $\Re$. Assim, o menor intervalo $\left[\underline{m}, \overline{m}\right]$, para os quais $\mu(x) = 1$, significa que x certamente pertence ao conjunto de valores possíveis. Existe um intervalo maior $\left[\underline{m}^{\lambda}, \overline{m}^{\lambda}\right]$, que contém $\left[\underline{m}, \overline{m}\right]$ e representa os valores de *x* que têm boas chances de pertencer ao conjunto de valores possíveis. Após este, existe um terceiro intervalo $\left[\underline{m}^{\varepsilon}, \overline{m}^{\varepsilon}\right]$, que envolve o segundo, representando valores de *x* que têm poucas chances de pertencer ao conjunto de valores possíveis. Os valores de *x*, com $\mu(x) < \varepsilon$, têm muito poucas chances de pertencer ao conjunto de valores possíveis. Usando a representação de seis pontos, um número *fuzzy* $\tilde{M}$ é representado pela lista de símbolos $\tilde{M} = \left(\underline{m}^{\varepsilon}, \underline{m}^{\lambda}, \underline{m}, \overline{m}, \overline{m}^{\lambda}, \overline{m}^{\varepsilon}\right)$, como mostrado na Figura 2.

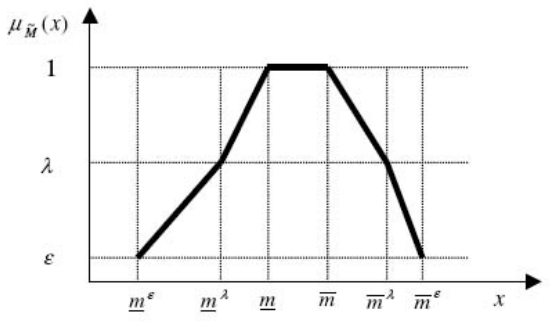


Figura 2 - Número Fuzzy na representação de seis pontos

O resultado de um cálculo *fuzzy* é uma programação *fuzzy*, que mostra números *fuzzy* para os tempos de início e fim das atividades. Logo, uma programação *fuzzy* na verdade pode ser visto como a composição de várias programações comuns.

Na Figura 2, μ é o valor de pertinência da duração da tarefa, p. As durações p_1 e p_6 possuem $\mu = \varepsilon$; p_2 e p_5 possuem $\mu = \lambda$; e p_3 e p_4 possuem $\mu = 1.0$.

Mais algumas maneiras de utilizar a modelagem baseada na teoria de possibilidades serão mostradas na seção 3.3.

2.3.3 RECURSOS

Na maioria dos estudos publicados, os recursos necessários para a realização de determinada atividade são conhecidos.

Em geral, nos problemas de Gerenciamento de Projetos, existem três classes de recursos:

i. Renováveis: existe uma determinada quantidade do recurso, disponível a cada período. Se o recurso não for utilizado totalmente em cada unidade de tempo, a sobra não pode ser reaproveitada na unidade de tempo seguinte. Exemplo: horas de trabalho de um funcionário.
ii. Não renováveis: uma determinada quantidade do recurso está disponível para todo o projeto. O uso do recurso pode ser feito de forma gradativa, mas não existe a possibilidade de adicionar recursos ao longo do projeto. Exemplo: matéria-prima.
iii. Duplamente restrito: os recursos nesta categoria estão incluídos nas duas categorias de restrição acima.

Para a formulação matemática dos problemas de programação de projetos, diferentes maneiras de utilização dos recursos podem ser empregadas. De acordo com Vasconcellos (2007), as formulações mais utilizadas são:

i. Recursos únicos ou múltiplos: usualmente no mundo real, pode haver vários tipos de recursos, como engenheiros, técnicos e auxiliares, por exemplo. Entretanto, é muito comum existir a possibilidade de simplificar o modelo e interpretar que muitos bens ou serviços tornam-se disponíveis com a utilização do recurso financeiro. Neste caso, só haveria um tipo de recurso: o financeiro.

ii. Requisitos variáveis ou constantes: normalmente, os recursos necessários para cada atividade são conhecidos e constantes. Entretanto, podem ser encontrados modelos em que as durações variam de acordo com os recursos empregados nas atividades. Para a modelagem onde os recursos podem ser modificados, possibilitando maior velocidade devido à alocação de mais recursos, recomenda-se a leitura de Demeulemeester et al. (2000).

iii. Modo único ou múltiplo de produção: mais aplicado ao estudo de programação de máquinas, onde determinada instalação fabril pode produzir um ou mais tipos de produtos.

iv. Correlação estocástica entre duração e requisitos de recursos: normalmente, devido ao surgimento de interrupções ou dificuldades não previstas inicialmente, um aumento na duração de determinada atividade implica diretamente em um aumento no consumo de recursos (alocados por mais tempo).

2.4 PROBLEMA DE PROGRAMAÇÃO DE PROJETOS COM RESTRIÇÃO DE RECURSOS

O problema de programação de projetos com restrição de recursos (Resource Constrained Project Scheduling Problem – RCPSP) pode ser modelado como a seguir Talbot (1982):

$$\min\left\{\max F_i \mid i = 1,2...,N\right\} \tag{2.1}$$

Sujeito a

$$F_j \leq F_i - D_i, \forall j \in P_i, i = 1,2,...,N \tag{2.2}$$

$$\sum_{A_t} r_{ik} \leq R_k, k = 1,2,...,K; t = S_1.S_2,...,S_N \tag{2.3}$$

Onde:

N é o número de atividades envolvidas no projeto;

F_i é o instante de conclusão da atividade a_i;

D_i é a duração da atividade a_i;

P_i é o conjunto das atividades necessariamente anteriores à atividade a_i (predecessoras de a_i);

R_k é a quantidade disponível do recurso k;

K é o número de tipos de recursos;

r_{ik} é a quantidade de recursos do tipo k necessários para a execução da atividade a_i;

A_t é o conjunto das atividades em processamento no instante t, e

S_i ($= F_i - D_i$) é o instante em que a atividade a_i começa a ser processada.

A equação (2.1) representa o objetivo, enquanto as equações (2.2) e (2.3), respectivamente, representam as restrições de precedência e de recursos.

Vale ressaltar que a formulação acima é inicial. No capítulo 5, será apresentada uma formulação completa, que foi utilizada neste estudo.

3. REVISÃO BIBLIOGRÁFICA DE MÉTODOS PARA ABORDAGEM DO RCPSP COM INCERTEZAS

Segundo Herroelen e Leus (2005), cinco abordagens podem ser identificadas para tratar de incertezas na definição de uma programação, quando a rede de precedência das atividades é determinística: programação reativa (*reactive scheduling*), programação estocástica (*stochastic scheduling*), programação com lógica fuzzy, programação robusta ou pró-ativa (*proactive scheduling*) e análise de sensibilidade. No fim deste capítulo, foi introduzida uma seção para mostrar apenas os estudos que utilizam uma das seguintes metaheurísticas: *Simmulated Annealing* (SA), Algoritmos Genéticos, Busca Tabu e *Particle Swarm Optimization* (PSO).

3.1 PROGRAMAÇÃO REATIVA

A abordagem reativa não se preocupa em tratar as incertezas durante a montagem da programação inicial, se concentrando em recalcular e reajustar o *schedule baseline* cada vez que algum evento inesperado ocorre. O *schedule baseline* é a programação que o projeto deve seguir, também usada para medir o progresso em sua execução. Para mais detalhes, são recomendados os seguintes trabalhos: Sabuncuoglu e Bayiz (2000), Szelke e Kerr (1994), Vieira et al. (2003).

Podemos analisar melhor a programação reativa segundo os dois extremos de estratégias para manter a consistência de sua solução. De um lado, está a estratégia de usar técnicas simples de reparo da programação inicial, apenas corrigindo os problemas causados pelos eventos inesperados que acontecerem. Como exemplo de uso de uma regra simples de controle, está a *right-shift rule* (Sadeh et al. (1993) e Smith (1994)). A técnica citada simplesmente realoca, em um instante posterior, as tarefas afetadas pelo evento inesperado. É importante ressaltar que este procedimento pode levar a soluções ruins, uma vez que as atividades não são seqüenciadas novamente. Ou seja, devido à realocação de algumas atividades, poderia ser calculada uma nova programação para todo o restante do projeto.

Por outro lado, com uma linha de ação mais custosa, porém, mais confiável, está a determinação de uma nova programação de todo o projeto, a partir do instante em que ocorreu o evento inesperado. Uma medida de impacto no planejamento total é a

comparação entre a duração prevista inicialmente e a esperada depois da reprogramação.

3.2 PROGRAMAÇÃO ESTOCÁSTICA

A programação estocástica utiliza as durações das atividades com incertezas, de forma a minimizar o tempo esperado do projeto mantendo as relações de precedência entre as atividades e as restrições dos recursos. O tema é discutido com detalhes no capítulo 9 do livro de Demeulemeester e Herroelen (2002).

Muitos estudos foram desenvolvidos de forma dedicada ao problema de programação de projetos estocástico com restrições de recursos (SRCPSP). Entre eles, são recomendados Fernandez (1995), Fernandez et al. (1996) e Fernandez et al. (1998).

O SRCPSP tem como objetivo programar as atividades, com durações com incertezas, de forma a minimizar o tempo total esperado para o projeto. O projeto é descrito como uma rede do tipo AON, possui n atividades e **d** é o vetor de durações aleatórias ($d_1, d_2, ..., d_N$), onde d_i representa a duração da atividade ***i***. A partir de uma amostra de **d**, composta por ($d_1, d_2, ..., d_N$) e respeitando o limite de recursos disponíveis a cada período, são gerados *schedules*.

Também é recomendado o artigo de Ke e Liu (2005). A solução apresentada é uma simulação estocástica e um algoritmo genético integrados para a criação de um algoritmo híbrido que resolve os modelos citados na seção 2.3.2.1.

3.3 SCHEDULING COM LÓGICA *FUZZY*

Como já abordado na seção 2.3.2.2, o grande motivo para o uso de lógica *fuzzy* na programação de projetos é a falta de dados históricos para a determinação da distribuição probabilística das durações das atividades. Sendo assim, profissionais com experiência no assunto do projeto estimam as durações das atividades. Como essas estimativas podem ser vagas e imprecisas, em vez de usar distribuições de probabilidade, a literatura recomenda o uso de números *fuzzy*, de forma a representar uma função de pertinência baseada teoria de possibilidade.

O estudo do modelo *fuzzy* para o RCPSP começou com Hapke et al. (1994). Os autores estabeleceram um método para gerar as programações otimistas e pessimistas, baseadas nos α-cortes na função de pertinência $\mu(x)$, que originam um número fuzzy

trapezoidal. A agregação de valores otimistas e pessimistas de um critério de minimização para todos os α-cortes fornece um resultado *fuzzy*. Esta técnica de solução será detalhada na seção 5.3.4.

Ozdamar e Alanya (2001) abordaram projetos de desenvolvimento de software e oferecem uma formulação matemática não-linear mista para o problema e propuseram uma heurística para solução. Os autores também ilustraram o uso de heurísticas com quatro regras de prioridade: regra da menor folga, do tempo de último término, maior número de sucessores imediatos e mínimo risco.

Wang (2004) propôs uma metodologia robusta baseada na teoria de conjuntos *fuzzy* para lidar com a incerteza em projetos de desenvolvimento de produtos. Os parâmetros imprecisos de tempo envolvidos no projeto são representados por conjuntos *fuzzy*. Uma medida da robustez da programação baseada em teoria qualitativa da possibilidade é proposta para guiar a busca por uma programação robusta, ou seja, a programação que tem a melhor performance no pior caso. Finalmente, um algoritmo genético é desenvolvido para resolver o problema com um desempenho aceitável.

Ke e Liu (2007) consideraram o problema de programação de projetos com incerteza mista de aleatoriedade e lógica *fuzzy*, com as durações das atividades sendo variáveis *fuzzy* aleatórias. São propostos três tipos de modelos para diferentes requisitos gerenciais: *fuzzy* como modelo de minimização do custo esperado, modelo de minimização de custo (α, β) e o modelo de maximização das chances. Simulações *fuzzy* aleatórias para algumas funções de incerteza são incorporadas no algoritmo genético, de forma a alcançar um algoritmo híbrido inteligente.

3.4 PROGRAMAÇÃO PRÓ-ATIVA

Os estudos de programação pró-ativa se concentram em três tipos: técnicas de redundância, técnicas de programação robusta e múltiplas programações.

3.4.1 TÉCNICAS DE REDUNDÂNCIA

A tolerância a falhas já foi bastante abordada no âmbito de programação de máquinas. Basicamente, existem dois procedimentos para a tolerância a falhas: redundância de tempo e redundância de recursos.

A redundância de tempo se baseia no fato de que podemos estimar a duração de uma tarefa e prever que, havendo algum impedimento, a duração pode ser aumentada de algum fator. Logo, se trabalha com a duração aumentada, já levando em conta estas possíveis interrupções. Para proteção temporal, a leitura de Gao (1995) é recomendada.

A redundância de recursos pode ser proibitiva em termos práticos, uma vez que, se todos os recursos forem dobrados, o projeto teria o dobro do custo inicial. Porém, podem ser calculados limites inferiores e métodos de ajustes de tempo no RCPSP. Carlier e Néron (2007) definem as funções de redundância de recursos que podem ser usadas para determinar limites inferiores para o RCPSP.

3.4.2 TÉCNICAS DE PROGRAMAÇÃO ROBUSTA

Dizemos que uma programação é robusta quando uma alteração nas restrições causa pouca alteração na programação inicial. Existem técnicas de medição de robustez de programação de máquinas, onde são contadas todas as alterações que ocorreriam na programação inicial. Para maiores detalhes, são recomendadas as seguintes leituras: Abbasi et al. (2006), Chtourou e Haouari (2008) e Van de Vonder et al. (2008).

Herroelen e Leus (2004) desenvolveram modelos de programação matemática para a geração de schedules estáveis. Lambrechts et al. (2008) propõem uma abordagem de programação robusta, utilizando busca tabu. Eles medem a robustez, ou a estabilidade, de uma programação pelo desvio entre os tempos de início das atividades nas situações planejada e realizada durante a execução do projeto.

3.4.3 MÚLTIPLAS PROGRAMAÇÕES (PLANOS DE CONTINGÊNCIA)

A abordagem de planos de contingência é baseada na geração de múltiplos *baselines* (ou fragmentos), com o objetivo de responder de forma ótima a possíveis interrupções. A idéia é determinar antecipadamente possíveis programações que devem ser seguidas em caso de necessidade. Com esta determinação anterior, quando algum evento inesperado ocorrer, basta utilizar a programação (ou fragmento) correspondente àquela situação. Assim, o foco está em flexibilidade e robustez. A aplicação desta abordagem é especialmente indicada nos casos em que a programação reativa esteja sendo utilizada, já que permite a troca em tempo real de programação a ser executada, descartando a

programação inicial (ou fragmento dela) e ativando a uma programação previamente estudada, já considerando a hipótese de ocorrer aquela interrupção.

Mauguière et al. (2002) e Aloulou et al. (2002) utilizam a idéia de criar para cada recurso uma seqüência de grupo, ou seja, um conjunto de grupos de operações, total ou parcialmente ordenado, e considerar todas as programações obtidas pela ordenação aleatória das operações dentro de cada grupo. Eles abordam esta idéia no contexto de uma máquina apenas.

3.5 ANÁLISE DE SENSIBILIDADE

Grande parte dos esforços de pesquisa recente está concentrada em responder a questões de análise de sensibilidade de problemas de programação de uma máquina. Entre os estudos na área, destacam-se: Hall e Posner (2000a) e Hall e Posner (2000b). Podemos interpretar a análise de sensibilidade como o esforço em achar respostas para perguntas do tipo “e se...?”. Os pesquisadores procuram encontrar soluções para problemas polinomiais e intratáveis de programação de máquinas. As perguntas são do tipo: (i) Quais os limites de alteração de um parâmetro para que a solução permaneça ótima? (ii) Dada uma alteração em determinado parâmetro, qual é o custo e qual é a nova solução ótima? (iii) Até quando a linha de base permanece ótima? (iv) Que tipos de análise de sensibilidade são úteis para identificar robustez e flexibilidade de determinada programação?

Embora existam muitos estudos sobre análise de sensibilidade na área de programação de máquinas, ainda existe muito espaço para mais estudos sobre isso na área de programação de projetos.

3.6 UTILIZAÇÃO DE METAHEURÍSTICAS

Existem muitos trabalhos importantes sobre a utilização de metaheurísticas em métodos de solução de problemas de programação de projetos com restrições de recursos. Por isso, estes trabalhos foram separados nesta seção. Entre eles, podemos citar os estudos sobre:

3.6.1 SIMULATED ANNEALING

Mika et al. (2005) abordam o RCPSP com fluxo de caixa descontado. O projeto é representado como uma rede de atividades nos nós (AoN). Um fluxo de caixa positivo é associado a cada atividade. O objetivo é maximizar o valor presente de todos os fluxos de caixa do projeto. Duas metaheurísticas são utilizadas para as buscas locais: SA e busca tabu.

3.6.2 ALGORITMOS GENÉTICOS

Vasconcellos (2007) desenvolveu uma ferramenta de apoio a decisão, utilizando AG e gerenciamento de riscos. Foi desenvolvido um algoritmo para solucionar um problema real, no planejamento de operações em poços de petróleo de um mesmo campo, desde a perfuração até a etapa final de preparação para produzir, com o objetivo que os poços entrassem o mais rápido possível em operação. O resultado obtido com o algoritmo genético foi comparado a soluções obtidas com algoritmos exatos (*Branch and Bound*), programação por restrições e *GRASP*.

Alba e Chicano (2007) utilizaram algoritmos genéticos para resolver diferentes cenários de projetos de software. Com o uso de um gerador de instâncias, eles puderam direcionar os estudos na influência que os atributos mais importantes do problema têm na solução.

Chang et al. (2001; Chang et al. (2008) aplicam e discutem o uso de algoritmos genéticos na resolução de problemas de projetos de desenvolvimento de softwares. Eles descrevem um novo modelo, baseado em AG, que produz programações ótimas ou próximas do ótimo, levando em conta as diferentes habilidades dos funcionários. Esta abordagem utiliza níveis de capacidade dos funcionários para executar tarefas diferentes e também o custo fixo (salário) e o custo variável (como horas extras, por exemplo) deles.

Kim et al. (2003) desenvolveram um algoritmo genético híbrido com um controlador de lógica *fuzzy* (CLF) para resolver o RCPSP. A abordagem foi baseada no projeto de operadores genéticos com CLF e inicialização com método serial, que tem se mostrado superior para problemas RCPSP de grande escala.

Valls et al. (2008) propuseram um algoritmo genético híbrido para a resolução do RCPSP, utilizando algumas mudanças em relação ao paradigma de AG: operador de

crossover específico para o RCPSP, operador incremental local que foi usado para todas as programações geradas, uma nova maneira de selecionar os pais a serem combinados e uma estratégia de duas fases, onde a segunda fase reinicia a evolução a partir de uma vizinhança da população de melhor programação da primeira fase.

Long e Ohsato (2007) propuseram a determinação inicial de um intervalo para as durações das atividades. Em seguida, um algoritmo genético é utilizado para a determinação dos valores ótimos de durações. Posteriormente, um profissional com experiência na área transforma o intervalo de valores possíveis para as durações em um número fuzzy. Neste caso, torna-se possível a criação de uma margem de segurança para o tempo total do projeto. Tal modelagem será detalhada e comentada posteriormente na seção 4.1 e, no capítulo 6, com resultados computacionais.

3.6.3 BUSCA TABU

Como já citado anteriormente, Mika, Waligóra et al. (2005) utilizaram *Simulated Annealing* e Busca Tabu como métodos de busca local para a abordagem de RCPSP com fluxo de caixa descontado.

Também já citados anteriormente, Lambrechts, Demeulemeester et al. (2008) se concentraram em estudos de programação pró-ativa, utilizando a geração de robustas *baselines*, com o máximo de proteção possível contra interrupções que podem acontecer durante a execução do projeto. Eles utilizaram procedimento de busca tabu, com função objetivo baseada em folga livre.

Pan et al. (2007) propuseram uma alternativa eficiente para a resolução de RCPSP. Em vez de utilizar um algoritmo tradicional de Busca Tabu, o estudo se concentrou em desenvolver um modelo de Busca Tabu pela modificação na maneira de encontrar a solução inicial.

3.6.4 *PARTICLE SWARM OPTIMIZATION* (PSO)

Zhang et al. (2006) propuseram a utilização de PSO para a resolução do RCPSP. As prioridades das atividades do projeto são representadas por partículas e, então, estas partículas são transformadas em programações viáveis, de acordo com as restrições de recursos e de precedência.

Zhang et al. (2005) mostraram o uso de PSO na solução do RCPSP de duas formas: representação baseada em prioridades e representação baseada em permutações. A primeira representação é a mesma citada no parágrafo anterior. A representação baseada em permutações modela uma partícula como uma seqüência de números inteiros distintos, de 1 a *N* (onde *N* é o número total de atividades). Cada seqüência (partícula) representa uma ordem para a execução das atividades. Para atender às restrições de precedência, uma atividade só pode aparecer na partícula depois de suas predecessoras. O objetivo é minimizar o tempo total do projeto. Esta técnica será explicada em detalhes na seção 4.2.

4. REVISÃO TEÓRICA DAS METAHEURÍSTICAS UTILIZADAS NESTE TRABALHO

Neste capítulo, é apresentada uma revisão teórica das metaheurísticas que fazem parte dos métodos de solução propostos nesta dissertação.

4.1 ALGORITMO GENÉTICO

O conceito de algoritmo genético foi introduzido por Holland (1975), com o intuito de resolver problemas de otimização difíceis de resolver por métodos exatos, especialmente os problemas de difícil modelagem matemática como aqueles que apresentam com diversos parâmetros ou características que precisam ser combinadas em busca da melhor solução; problemas com muitas restrições ou condições que não podem ser representadas matematicamente; e problemas com grandes espaços de busca Pacheco (2005).

O algoritmo genético reproduz a evolução natural de uma população através da mutação de indivíduos e do cruzamento entre dois indivíduos desta população. É gerada uma população inicial, que servirá de base para as próximas gerações. É utilizada uma função de avaliação, para determinar as melhores soluções dentre os indivíduos da população. Cada geração seguinte é determinada com o uso dos operadores de cruzamento e mutação.

Cada indivíduo é representado por um cromossomo, que representa uma potencial solução para o problema em questão. Os cromossomos são vetores *1xN* (Figura 3), onde N é o número total de atividades. A população é composta de M cromossomos, sendo representada por uma matriz *MxN*.

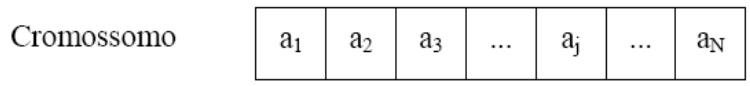


Figura 3 - Cromossomo do algoritmo genético

Em cada iteração, com as durações das atividades representadas no cromossomo, para toda a população é calculada a duração total e os cromossomos são classificados de acordo com os valores encontrados. Após esta classificação, ocorrem os cruzamentos e as mutações em parte da população. A seguir, inicia-se uma nova iteração.

No cruzamento, são selecionados dois cromossomos, chamados de pais, e um ponto de corte (escolhido aleatoriamente a cada cruzamento). Os pais são recombinados, de forma a concatenar as partes dos dois pais antes e depois do ponto de corte.

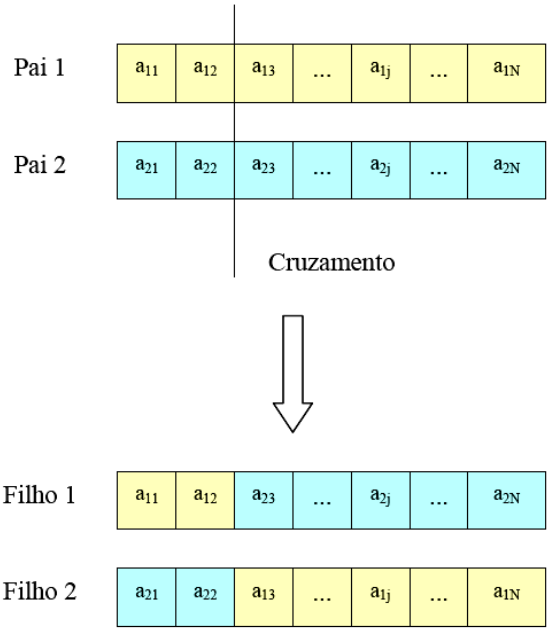


Figura 4 - Cruzamento em um algoritmo genético

Na mutação, o que ocorre é a escolha dos cromossomos que sofrerão a mutação e, em cada um escolhido, é feita a mudança do valor da duração de uma atividade, escolhida aleatoriamente.

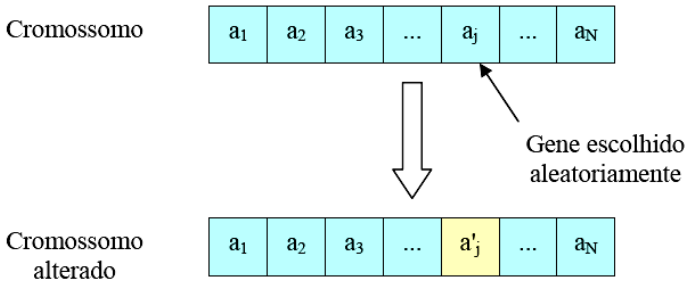


Figura 5 - Mutação em um algoritmo genético

Para determinar as quantidades de cromossomos que sofrerão cruzamento e mutação, são utilizados dois parâmetros: taxa de cruzamento e taxa de mutação.

A taxa de cruzamento é um número fixo no intervalo (0,1), que indica a porcentagem de cromossomos que serão cruzados. Esta quantidade de cromossomos cruzados é constante ao longo das iterações do algoritmo genético. A taxa de mutação também é um número fixo no intervalo (0,1), que representa a porcentagem de

cromossomos que serão mudados. No artigo original (Long e Ohsato (2007)), foi utilizado o valor 0,03. Porém, durante a implementação realizada para este estudo, a taxa de mutação foi considerada variável. Esta alteração permitiu que a população ficasse mais heterogênea, ou seja, com maior diversidade entre os cromossomos. O método de alteração da taxa de mutação foi baseado no número de iterações do algoritmo genético sem que houvesse alteração da melhor solução. Quando ocorrem P iterações sem que a melhor solução seja alterada, a taxa de mutação é multiplicada por um fator q.

Quando o número máximo de iterações é atingido, a melhor solução encontrada não pode ser vista como a solução ótima, uma vez que o algoritmo genético é uma metaheurística. Porém, em geral, é uma solução perto da solução ótima. A programação resultante é uma programação determinística, com as durações adequadas das atividades $t^*(j), j = 1, ..., N$ e, também, com os instantes marcados para as atividades começarem $AS(j), j = 1, ..., N$.

A Figura 6 mostra o resumo do procedimento de um algoritmo genético.

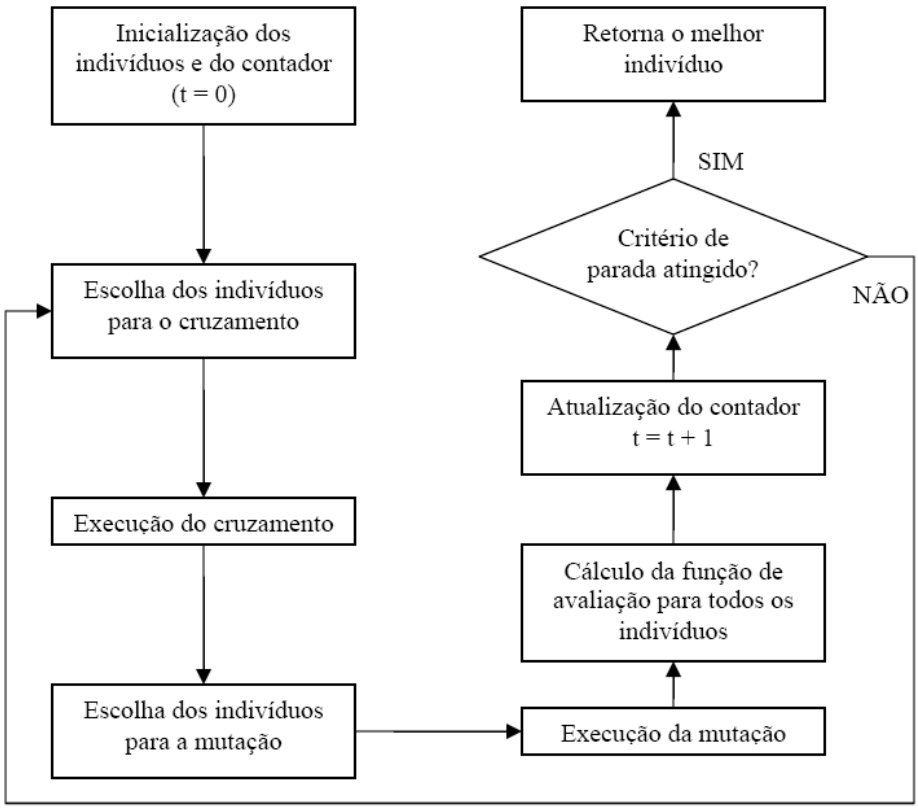


Figura 6 - Procedimento de um algoritmo genético

4.2 OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS

A otimização por enxame de partículas (*Particle Swarm Optimization - PSO*, em inglês) foi proposto em Eberhart e Kennedy (1995). É um método de busca que visa à imitação dos movimentos de indivíduos em um enxame durante a procura por um objetivo (por exemplo, alimento). No PSO, um enxame de M indivíduos, chamados de partículas, é usado para buscar a solução para determinado problema.

A matriz de posições é composta por M linhas, onde a posição da j-ésima partícula no instante t pode ser escrita como: $X_j(t) = \{x_{j1}(t), x_{j2}(t), ..., x_{jN}(t)\}$. A posição $\boldsymbol{X_j(t)}$ de uma partícula representa a solução dada por aquele indivíduo para o problema.

A matriz de velocidades é definida de forma semelhante. Também é uma matriz *MxN*, onde a velocidade da j-ésima partícula no instante t pode ser escrita como: $V_j(t) = \{v_{j1}(t), v_{j2}(t), ..., v_{jN}(t)\}$. A velocidade $\boldsymbol{V_j(t)}$ da partícula $\boldsymbol{j}$ direciona para onde será dado seu próximo passo. Durante as iterações do algoritmo, a posição da partícula no instante t+1 é calculada de acordo com sua posição e sua velocidade no instante t. Pelo fato de a partícula se mover de uma posição para a outra, ela está fornecendo outra possível solução para o problema.

O aspecto cognitivo do PSO está na melhor posição pessoal de uma partícula, definida como a posição que fornece a melhor função objetivo entre todas as posições visitadas pela partícula. Cada vez que a partícula chega a uma posição que fornece um valor melhor que as posições anteriores para a função objetivo, a melhor posição pessoal $\boldsymbol{B_i^L}$ é atualizada.

O PSO também utiliza o fato de que, em um grupo, o comportamento dos vizinhos influencia no movimento de cada indivíduo. Sendo assim, quando qualquer partícula do enxame atinge uma posição que fornece o melhor valor para função objetivo já alcançado, esta informação é repassada para todas as outras partículas e a melhor posição global $\boldsymbol{B^G}$ é atualizada.

As melhores posições pessoal e global são usadas para a atualização da velocidade de cada partícula. Em cada passo da iteração, a velocidade é calculada baseada em três aspectos: inércia, aprendizado cognitivo e aprendizado social.

O fator de inércia, $\boldsymbol{w}$, é utilizado para controlar o impacto dos valores anteriores de velocidade. Valores muito altos para o fator de inércia fazem com que a partícula tenha

maior resistência a movimentos bruscos de sua posição atual. O termo cognitivo induz a partícula a seguir a direção de sua melhor posição pessoal, a partir de sua posição atual.

O termo relativo ao comportamento social induz a partícula a buscar, a partir de sua posição atual, a melhor posição global do enxame. Além de todos os termos citados anteriormente, existe a inserção de fatores aleatórios no cálculo para atualização da velocidade. Esta inserção é feita para que os movimentos fiquem menos previsíveis. Assim, duas partículas que tenham posições atuais e melhores posições pessoais parecidas podem se mover em direções diferentes.

Agora, definem-se as equações para atualização das posições e das velocidades das partículas:

$$V_j(t+1) = w.V_j(t) + c_1.r_1.(B_j^L - X_j(t)) + c_2.r_2.(B^G - X_j(t)) \quad (4.1)$$

$$X_j(t+1) = V_j(t+1) + X_j(t) \quad (4.2)$$

Nas equações (4.1) e (4.2), j = 1, 2, ..., M representam os indivíduos do enxame. Os fatores de aprendizado $\boldsymbol{c_1}$ e $\boldsymbol{c_2}$ são constantes positivas, no intervalo (0,1), $\boldsymbol{r_1}$ e $\boldsymbol{r_2}$ são valores aleatórios também no intervalo (0,1) e $\boldsymbol{w}$ é o fator de inércia. $B_j^L = \{x_{j1}^L, x_{j2}^L, ..., x_{jN}^L\}$ é a melhor posição já atingida pela j-ésima partícula e $B^G = \{x_1^G, x_2^G, ..., x_N^G\}$ é a melhor posição já atingida por todo o enxame, ambas até a iteração $\boldsymbol{t}$.

Pode-se, então, resumir o procedimento do PSO como ilustrado na Figura 7:

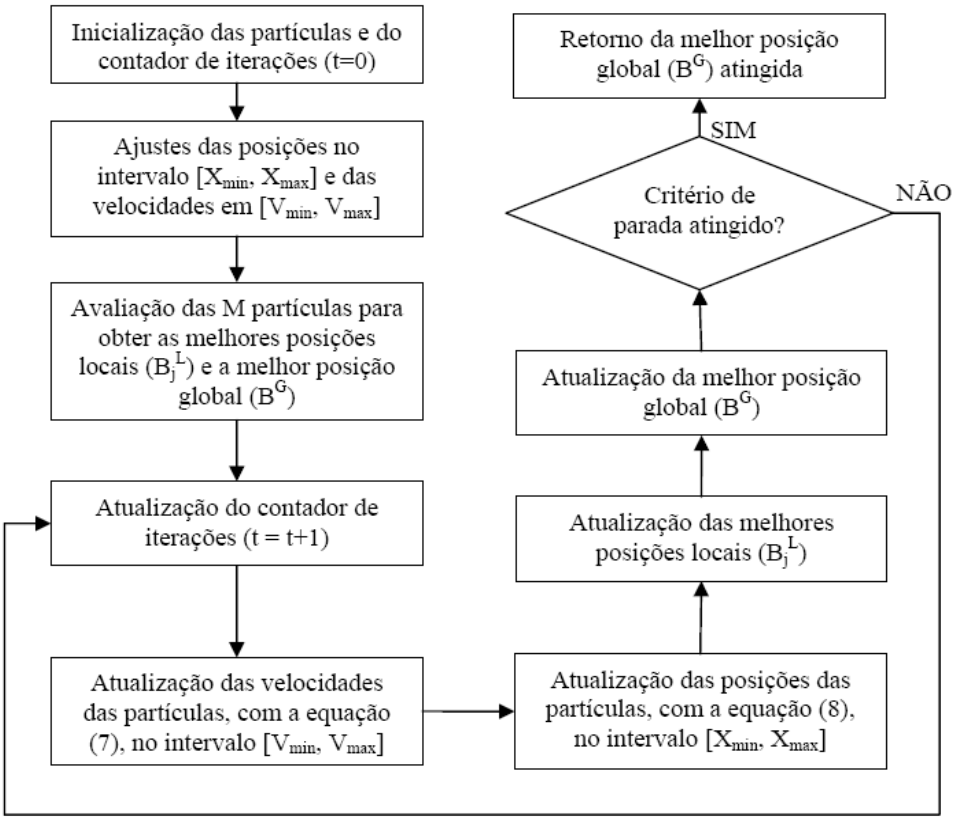


Figura 7 - Procedimento de um algoritmo de PSO

5. O PROBLEMA DE PROGRAMAÇÃO DE PROJETOS COM RESTRIÇÕES DE RECURSOS E INCERTEZAS E METODOLOGIAS PARA SUA SOLUÇÃO

A resolução do problema de programação de projetos com restrições de recursos e incertezas pode ser vista como encontrar a menor duração total do projeto, encontrando a duração adequada de cada atividade, além do instante em que cada atividade deve ser finalizada, tendo como entradas as relações de precedência entre as atividades, os recursos necessários para cada atividade, a disponibilidade total de recursos e o intervalo válido para a duração de cada atividade. Neste estudo, o problema de programação de projetos foi modelado como uma rede com atividades nos vértices, onde cada nó da rede representa uma atividade e as arestas representam as dependências entre as atividades.

5.1 FORMULAÇÃO MATEMÁTICA

Seja a seguinte notação:

Índices:

i: i = 1, ..., *N* indicam as atividades,

k: k = 1, ..., *K* indicam os recursos;

Parâmetros:

Cada atividade ***i*** deve ter sua duração p_i escrita como um número fuzzy com dois α-cortes, ou seja, p_i = (***a(i)***, ***b(i)***, ***c(i)***, ***d(i)***).

N é o número de atividades do projeto;

K é a quantidade de tipos diferentes de recursos;

R_k é a quantidade disponível de recursos do tipo ***k,*** k = 1, ..., K;

r_{ik} é a quantidade de recursos do tipo ***k*** necessários para a atividade ***i***, calculada pela razão entre a quantidade total necessária e o tempo de execução da tarefa

$(r_{ik} = \frac{R_{ik}}{t*(i)})$, i = 1, ..., N e k = 1, ..., K;

Conjuntos:

P_i é o conjunto de atividades predecessoras da atividade ***i,*** i = 1, ..., N; e

A_t é o conjunto das atividades em andamento no instante ***t***, t = 1, ..., max ***Fim(i)***

Variáveis de decisão:

Fim(i) é a variável de decisão que mostra o instante em que a atividade ***i*** é concluída, $i = 1, ..., N$ e

t*(i) é a variável de decisão que denota a duração da atividade ***i***, $i = 1, ..., N$;

O problema pode, então, ser formulado da seguinte forma:

$$MIN\left\{MAX\ Fim(i) \mid i = 1,2,...,N\right\} \tag{5.1}$$

Sujeito a

$$b(i) \le t^*(i) \le c(i); i = 1,2,...,N \tag{5.2}$$

$$Fim(j) \le Fim(i) - t^*(i); \forall j \in P_i; i = 1,2,...,N \tag{5.3}$$

$$\sum_{A_t} r_{ik} \le R_k, k = 1,2,...,K; t = 1,2,...,\max Fim(i); i = 1,2,...,N; \text{onde } r_{ik} = \frac{R_{ik}}{t^*(i)} \tag{5.4}$$

Todas as variáveis de decisão são inteiras e não negativas (5.5)

A equação 5.1 descreve o objetivo do problema, que é a minimização do tempo total do projeto. As equações 5.2 representam que a duração adequada da atividade, que deve ser determinada, está em um intervalo [*b(i), c(i)*] determinado pelo gerente do projeto. As equações 5.3 mostram que cada atividade só pode começar quando estiverem concluídas todas as atividades de que ela depende. As equações 5.4 representam as restrições de recursos disponíveis a cada período.

5.2 SEQÜENCIAMENTO DETERMINÍSTICO DE ATIVIDADES

Um elemento central na solução do problema de programação com restrições de recursos e incertezas é um procedimento que permita seqüenciar as atividades nas situações onde são conhecidos:

i. Relações de precedência entre as atividades;
ii. Recursos necessários para a realização de cada atividade;
iii. Duração de cada atividade;
iv. Quantidade de recursos disponíveis.

Por isso, é útil apresentar em destaque a heurística que, a partir das informações listadas acima, obtém a programação do projeto. Cabe ressaltar que, no presente estudo, o único fator de incerteza considerado está na duração das atividades. Logo, nas técnicas descritas posteriormente, apenas os valores de duração não serão considerados conhecidos *a priori*.

Com as informações conhecidas, é feito o seqüenciamento das atividades, de forma a encontrar a duração total do projeto. Este seqüenciamento segue o algoritmo a seguir:

Algoritmo Seqüenciamento de Atividades:

t ← 0; A(t) ← {}; H(t) ← {}; Q(t) ← {};

Q(t) ← atividades sem predecessor;

Enquanto Q(t) não for vazio

A(t) ← atividades concluídas;

Q(t) ← atividades que podem começar;

H(t) ← lista de prioridades;

Agendar as atividades, de acordo com H(t), enquanto houver recursos disponíveis;

Atualizar recursos;

t ← t+1;

Fim_Enquanto

Quando todas as atividades tiverem sido executadas, o algoritmo retorna o tempo total do projeto.

A lista de prioridades H(t) pode ser ordenada por diferentes critérios. Cada atividade ***i*** possui um valor de prioridade $\boldsymbol{h_i}$ associado e, na ordenação da lista, as atividades com menor valor $\boldsymbol{h_i}$ têm maior prioridade de execução. Alguns critérios sugeridos por Hapke, Slowinski et al. (1994) são:

i. $\boldsymbol{h_i}$ = início mais cedo da atividade **i**, determinado pelo CPM;
ii. $\boldsymbol{h_i}$ = término mais cedo da atividade **i**;
iii. $\boldsymbol{h_i}$ = início mais tarde da atividade **i**;
iv. $\boldsymbol{h_i}$ = término mais tarde da atividade **i**;
v. $\boldsymbol{h_i}$ = folga entre o início mais tarde e o mais cedo da atividade **i**;
vi. $\boldsymbol{h_i}$ = duração da atividade **i**;
vii. $\boldsymbol{h_i} = \boldsymbol{r_{ik}}$, onde $\boldsymbol{r_{ik}}$ é a quantidade de recursos por período do tipo **k** necessários para a realização da atividade **i**.

Segundo os testes citados pelos autores, não existe um critério que tenha melhores resultados do que os outros critérios de forma global. Logo, nenhum critério é reconhecidamente mais indicado que os outros. Nesta dissertação, o critério utilizado foi o de menor folga no CPM, item *v* da lista.

5.3 PROGRAMAÇÃO DE PROJETOS *FUZZY*

A Programação de Projetos *Fuzzy* (*Fuzzy Project Scheduling - FPS*) é apresentada em Hapke, Slowinski et al. (1994), e explicada com detalhes no APÊNDICE A do presente estudo. A seguir, serão apenas apontados os aspectos relevantes para o entendimento das técnicas propostas de solução.

Como já apresentado na seção 3.3, os números *fuzzy* podem ser representados por seis pontos, com α-cortes, baseados na função de pertinência. Os três passos propostos para tratar as incertezas são:

i. modelar as incertezas;
ii. transformar o problema com incertezas em um problema determinístico associado;
iii. solucionar o problema determinístico associado, verificar os resultados e o possível retorno a (ii) para revisão de parâmetros.

5.3.1 MODELAGEM DAS INCERTEZAS

Um método adequado de se modelar as incertezas é, segundo Rommelfanger (1990), definir níveis de corte da função de pertinência. Para isso, ele propôs a seguinte divisão:

- α = 1: $\mu(x) = 1$ significa que o valor de x certamente pertence ao conjunto de valores possíveis;
- α = λ: $\mu(x) > \lambda$ significa que os valores de x com $\mu(x) \geq \lambda$ apresentam boas chances de pertencer ao conjunto de valores possíveis;
- α = ε: $\mu(x) < \varepsilon$ significa que valores estes valores de x têm poucas chances de pertencer ao conjunto de valores possíveis.

No presente estudo, por questões práticas, foram utilizados apenas dois cortes. Os valores utilizados foram α = 0,80 e α = 0,20. Assim, valores que apresentam mais de

80% de chances de pertencerem ao conjunto de valores possíveis estão no nível α = 0,80, e os valores que apresentam até 20% de chances de pertencerem ao conjunto válido estão no nível α = 0,20.

5.3.2 TRANSFORMAÇÃO EM PROBLEMA DETERMINÍSTICO

Como cada número *fuzzy* é definido por dois α-cortes, e cada α-corte determina dois valores de durações de atividades, pode-se concluir que cada número *fuzzy* representa quatro possíveis valores de duração.

Com estes quatro valores de duração para cada atividade, a cada problema de programação de projetos com incerteza estão associados quatro problemas determinísticos.

5.3.3 SOLUÇÃO DO PROBLEMA DETERMINÍSTICO

Na fase de solução do problema, existem, então, quatro programações diferentes, que devem ser ordenadas para a apresentação de um único resultado de programação *fuzzy*.

Hapke, Slowinski et al. (1994) propõem um método de solução, onde os quatro resultados encontrados são combinados em um número *fuzzy* final. Este método de solução é descrito em detalhes na seção 5.3.4. Além deste método, esta dissertação propõe o método descrito na seção 5.3.5.

5.3.4 MÉTODO FPS-G4

Uma maneira rápida e de baixo custo computacional para resolver o problema proposto é a decomposição deste em diferentes problemas determinísticos, porém, com os valores das durações das atividades agrupados pela sua posição no número *fuzzy* daquela atividade.

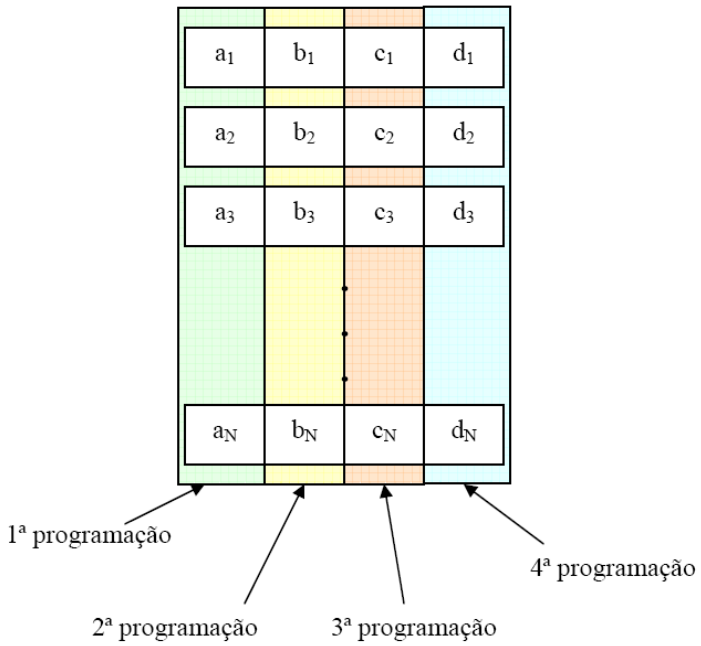


Figura 8 – Método de solução FPS-G4

Cada duração p_i de atividade é representada por quatro valores diferentes (***a(i)***, ***b(i)***, ***c(i)***, ***d(i)***). O agrupamento deve seguir os índices. Logo, um problema determinístico a ser resolvido é composto pelas durações ***a(i)***, para todo *i*. O próximo problema determinístico é composto pelas durações ***b(i)***, para todo *i*. Assim sucessivamente, gerando programações com durações ***c(i)*** e ***d(i)***, para todo *i*.

Este método de solução não leva em conta o fato de que as atividades são eventos independentes, de forma que uma atividade ter sido realizada de acordo com a expectativa mais otimista não implica no fato que a atividade seguinte também deve ser realizada no menor tempo possível.

Não é difícil perceber que, se forem utilizados os valores mais otimistas (***a(i)***, para todo *i*), a programação encontrada é a mais otimista possível. Da mesma forma, se forem utilizados os valores mais pessimistas (***d(i)***, para todo *i*), a solução encontrada é a mais pessimista possível. Entretanto, os valores intermediários podem assumir qualquer comportamento.

A partir da duração total do projeto escrita como um número *fuzzy*, existe a necessidade de transformar este número *fuzzy* em um valor que possa ser analisado e comparado com outras possíveis soluções. A transformação da duração *fuzzy* do projeto em um único valor utiliza a idéia de duração com alto grau de concordância, descrita com detalhes na seção 5.6.2.

A motivação deste método é a sua simplicidade e sua rapidez de resposta.

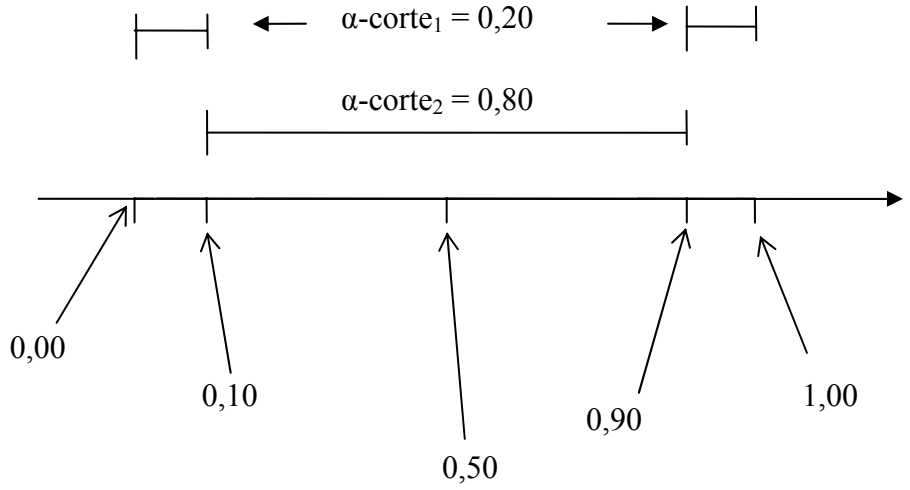
5.3.5 MÉTODO FPS-SIM

A solução imediata abordada em 5.3.4 ignora o fato de que as atividades são independentes. Agora, a proposta é exatamente tratar esta independência.

Como cada atividade tem sua duração prevista por quatro valores (***a(i)***, ***b(i)***, ***c(i)***, ***d(i)***), este método propõe que sejam sorteados os valores das durações das atividades entre esses valores possíveis. Porém, os sorteios são independentes para cada atividade.

A proposta é dividir o intervalo (0,1) em partes proporcionais às chances de determinado valor de duração de atividade acontecer. Para isso, é necessário saber os valores de cada α-corte. A seguir, será apresentado um exemplo numérico de divisão do intervalo.

Considere os valores de α-cortes iguais a 0,20 e 0,80. Neste caso, os cortes são proporcionais a 1 e 4. Com isso, dividindo o intervalo em partes iguais, a unidade básica tem comprimento de 0,10. Assim, os α-cortes teriam comprimento de 0,10 (α-corte = 0,20) e 0,40 (α-corte$_2$ = 0,80).



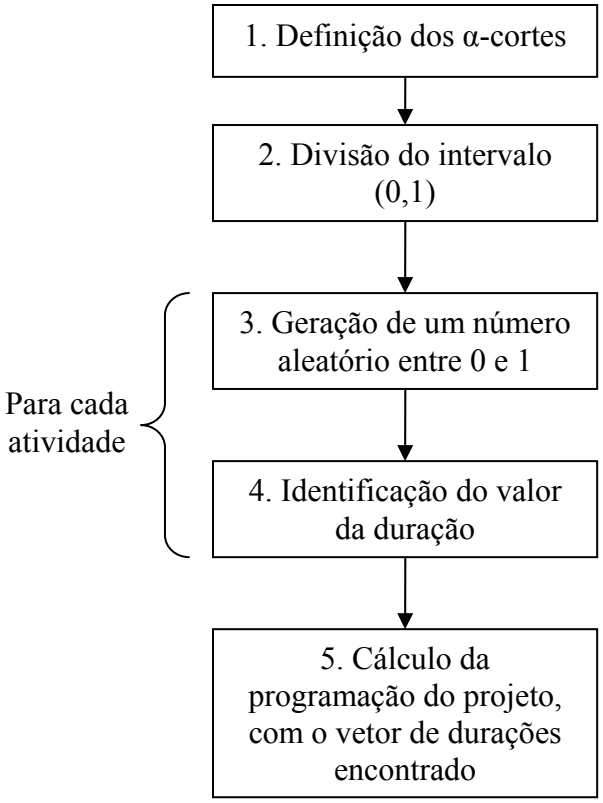
Com esta divisão do intervalo (0,1), a determinação do valor de duração da atividade a ser utilizado pode ser resumida assim:

r = número aleatório

$$p(i) = \begin{cases} a(i), \text{ se } 0{,}00 < \text{r} < 0{,}10 \\ b(i), \text{ se } 0{,}10 \leq \text{r} < 0{,}50 \\ c(i), \text{ se } 0{,}50 \leq \text{r} < 0{,}90 \\ d(i), \text{ se } 0{,}90 \leq \text{r} < 1{,}00 \end{cases}$$

Fazendo o procedimento descrito acima para todas as atividades, tem-se um vetor de valores de durações de atividades. Com este vetor, realiza-se o cálculo da programação do projeto.

A seguir, um resumo dos passos do procedimento.



O procedimento descrito possibilita a obtenção de uma programação determinística para cada conjunto de durações sorteadas. Com a repetição deste procedimento por um número suficientemente alto de vezes, pode-se determinar uma curva de distribuição da duração total do projeto. Este procedimento permite que sejam utilizados tratamentos estocásticos para a duração total do projeto.

Nesta dissertação, para a comparação dos resultados atingidos pelo método FPS-SIM, foi calculada a média aritmética dos valores de duração do projeto encontrados no processo de simulação.

5.4 ABORDAGENS METAHEURÍSTICAS

A programação de projetos, utilizando um algoritmo genético e *buffer* é proposta por Long e Ohsato (2007). As durações das atividades são previamente definidas por intervalos pelo gerente do projeto. Dentro destes intervalos, um método baseado em uma metaheurísica encontra a melhor programação, com o objetivo de minimizar a

duração total do projeto, respeitando as restrições de recursos. Com o resultado inicial, o gerente pode caracterizar as durações das atividades como números *fuzzy* para tratar as incertezas. Em vez de calcular uma programação *fuzzy*, o método propõe um *buffer* para absorver as variações nas durações e obter a programação final do projeto. A interpretação de *buffer* de projeto nesta abordagem é semelhante à interpretação de *buffer* na área de computação, onde ele é uma área intermediária de memória que serve para evitar problemas devidos a diferenças entre a taxa de leitura e a taxa de gravação. No caso de programação de projetos, a interpretação é de proteger a programação de impactos pela diferença entre o desempenho planejado e o desempenho real durante a execução das atividades.

A principal diferença entre as abordagens metaheurísticas propostas e o método FPS-SIM é a saída de cada um. Enquanto no FPS-SIM a saída é um conjunto de valores de durações totais, de onde é possível calcular a média, nas abordagens metaheurísticas a saída é um valor de duração total acrescido do tamanho do *buffer* de projeto.

No presente estudo, foram implementadas duas metaheurísticas, a primeira baseada em um algoritmo genético e a segunda baseada em otimização por enxame de partículas (*Particle Swarm Optimization* - PSO).

5.4.1 MODELAGEM *FUZZY* PARA AS DURAÇÕES DAS ATIVIDADES

As durações das atividades serão representadas utilizando o modelo de números *fuzzy* trapezoidais (*a(i), b(i), c(i), d(i)*), com dois α-cortes. A função de pertinência μ tem valores no intervalo [0,1] e é usada para expressar o quanto um valor de tempo pertence ao conjunto de durações. Os limites iniciais para cada duração [*b(i)*,*c(i)*] são usados na modelagem *fuzzy* também. Um dos objetivos deste método é encontrar os valores adequados de durações de atividades (***t*(i)***).

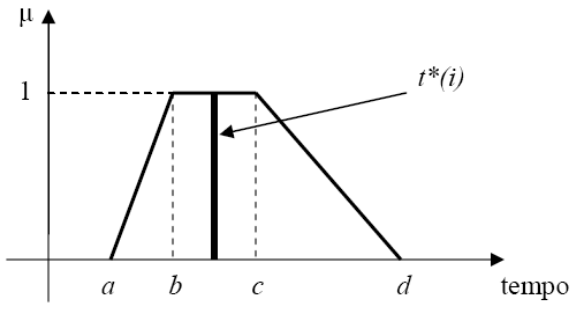


Figura 9 - Duração adequada (t*)

5.4.2 METODOLOGIA DE SOLUÇÃO

Para abordar o problema de programação de projetos com restrições de recursos e incertezas utilizando metaheurísticas (AG e PSO), o procedimento adotado está descrito a seguir:

i. Determinação das durações adequadas das atividades

A partir deste momento, os algoritmos baseados em metaheurísticas devem encontrar os valores adequados de duração para cada atividade, dentro do intervalo [b(i), c(i)], de forma a respeitar as restrições de recursos e de precedência, minimizando a duração do projeto.

Para isso, o algoritmo utilizado é mostrado a seguir. Vale ressaltar que a diferença entre o PSO e o algoritmo genético está apenas na avaliação e na evolução da população.

Algoritmo Determinação de durações adequadas

Gerar a população inicial;
Enquanto critério de parada não é atingido
Para todo indivíduo da população
Decodificar o cromossomo (ou partícula) para obter as durações;
Seqüenciar as atividades;
Avaliar o indivíduo;
Fim_Para
Evoluir a população;
Fim_Enquanto

Os passos deste algoritmo são mais detalhadamente descritos na seção 5.5.

ii. Determinação do tamanho do *buffer* de projeto. A programação determinada no item (i) continua valendo, porém, adicionando-se este *buffer*. O tamanho do *buffer* é determinado com cálculos envolvendo números *fuzzy*. Para maiores detalhes, ver seção 5.6.

5.5 MODELAGEM E EXECUÇÃO DO ALGORITMO

A modelagem dos indivíduos foi feita da mesma forma para o PSO e o algoritmo genético. Cada indivíduo é um vetor de N-dimensional, onde N é o número de atividades do projeto. Cada elemento ***u*** do vetor é um número real, no intervalo (0,1), que vai mapear o intervalo [***b(i),c(i)***] da atividade ***i***.

A decodificação de um gene do cromossomo em uma duração de atividade é feita de acordo com a equação 5.6 a seguir:

$$t(i) = b(i) + x_{ji}\left[c(i) - b(i)\right] \quad (5.6)$$

Na equação (5.6), ***i*** é o índice de atividade (i = 1, ..., N) e ***j*** é o índice de indivíduo (j = 1, ..., M), onde M é o tamanho da população. Ou seja, o elemento $\boldsymbol{x_{ji}}$ é o elemento correspondente à atividade ***i*** no indivíduo ***j***, na matriz de população. Assim, cada indivíduo é convertido em uma configuração possível, com as durações das atividades. Com essas durações, é feito o seqüenciamento das atividades, de forma a encontrar a duração total do projeto, a qual será a avaliação dos indivíduos.

5.6 DETERMINAÇÃO DO *BUFFER* DE PROJETO

O uso de *buffer* de projeto foi introduzido por Goldratt (1997). Segundo Patrick (1999), o uso de *buffers* é bem diferente de gerenciar folgas de tempo. As folgas acontecem aleatoriamente ao longo do planejamento do projeto, enquanto que os buffers são calculados para refletir as incertezas nas durações das atividades. O *buffer* de projeto é baseado na cadeia crítica, formada pelas atividades que não podem sofrer atraso sem implicar em alteração da duração total do projeto. Se houver mais de uma cadeia crítica, será considerada a maior delas. É importante ressaltar que o *buffer* é acrescentado à duração total do projeto tendo em vista aumentar a robustez do planejamento, mesmo que isto cause uma perda de eficiência na solução.

5.6.1 GRAU DE CONCORDÂNCIA

O grau de concordância GC(A,B) é usado para medir a possibilidade de dois números concordarem ao modelar dois eventos *fuzzy* (Kaufman e Gupta (1985),

Loterapong e Moselhi (1996) e Long e Ohsato (2007)). A idéia é medir o quanto um número *fuzzy* ***A*** é parecido com outro ***B***.

O grau de concordância é definido como a seguir:

$$GC(A,B) = \frac{Área(A \cap B)}{Área(A)} \quad (5.7)$$

onde $Área(A) = \int \mu_A(t).dt$ e $Área(A \cap B) = \int \mu_{A \cap B}(t).dt$.

O índice de concordância representa que porcentagem do número *fuzzy* ***A*** está contido nas fronteiras do número *fuzzy* ***B***. Cabe ressaltar que o índice de concordância não é comutativo, uma vez que os números ***A*** e ***B*** determinam áreas diferentes sob suas funções de pertinência.

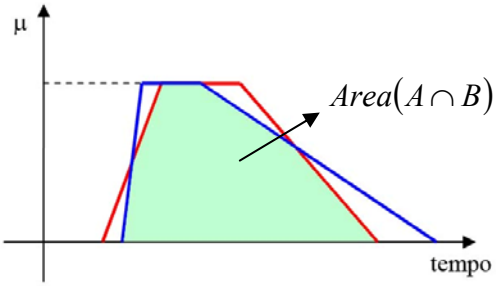


Figura 10 - Grau de concordância entre números *fuzzy*

5.6.2 DURAÇÃO COM ALTO GRAU DE CONCORDÂNCIA

É um valor determinístico de duração que pode ser considerado altamente semelhante, ou quase equivalente, a determinado número *fuzzy*. Em outras palavras, se a duração de uma atividade é determinada por um número *fuzzy*, um valor determinístico poderia ser usado como a duração da mesma atividade, com algum grau de concordância. Utilizando-se o conceito de grau de concordância, é determinado o valor t_H que pode substituir a duração *fuzzy* da atividade.

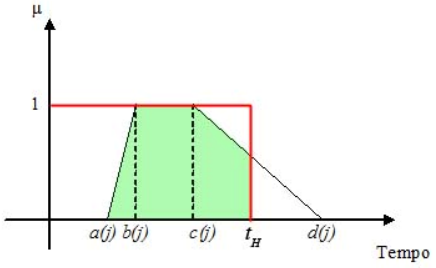


Figura 11 - Duração com alto grau de concordância (t_H)

Neste estudo, foi utilizado o valor 0,9 para o grau de concordância. Ou seja, para cada atividade ***j*** é calculado o valor $t_H(j)$, tal que a área marcada da Figura 11 seja 0,9 vezes a área determinada pelo número *fuzzy* trapezoidal.

5.6.3 CÁLCULO DO TAMANHO DO BUFFER DE PROJETO (BP)

Para a determinação do tamanho do *buffer* de projeto, é necessária a utilização dos tempos de segurança (***ts***). Os tempos de segurança são calculados pela diferença entre a duração com alto grau de concordância ($t_H(i)$) e a duração encontrada pelo algoritmo ($t^*(i)$).

$$ts(i) = t_H(i) - t^*(i) \quad (5.8)$$

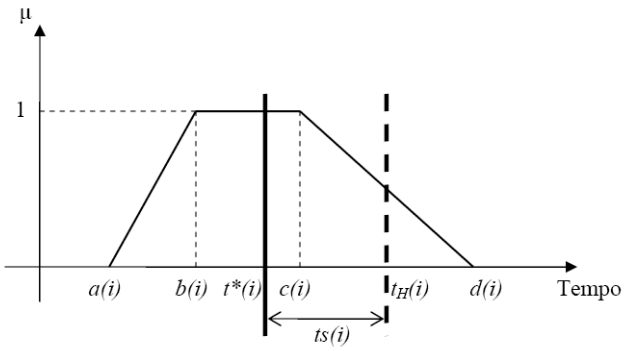


Figura 12 - Representação de t_H(i) e ts(i)

Existem diferentes maneiras de cálculo do tamanho do buffer de projeto. Para maior aprofundamento, é recomendado Tukel, Rom e Eksioglu (2006). Por questões de proximidade com a modelagem de durações, foi utilizado o método proposto por Long e Ohsato (2007), onde se define o tamanho do *buffer* de projeto a partir dos tempos de segurança das atividades da cadeia crítica. Isto acontece porque as variações causadas pelas durações *fuzzy* das atividades tendem a se compensar quando o número de atividades fica muito grande. Chamando a cadeia crítica de CC, a expressão (5.9) mostra o tamanho do *buffer* de projeto.

$$BP = \sqrt{\sum_{i \in CC} ts(i)^2} \quad (5.9)$$

6. EXPERIMENTOS REALIZADOS

Para avaliar as metodologias desenvolvidas para resolver o RCPSP com incertezas, foram feitos experimentos com diferentes instâncias de projetos. Cada projeto foi representado como um grafo orientado, com a atividade sendo representada por um nó e as relações de precedência sendo representadas pelas arestas. Nas seções a seguir, serão detalhados os experimentos e as configurações relevantes para a comparação entre os métodos.

6.1 INSTÂNCIAS

Com o intuito de diversificar os experimentos, foram utilizadas instâncias de 30, 60, 90 e 120 atividades. As instâncias foram retiradas do PSPLib (http://129.187.106.231/psplib/) porém, as durações apresentadas no site citado não são modeladas como números *fuzzy*. Por isso, as durações foram geradas de forma aleatória, através da função *rand* do Matlab, que fornece números aleatórios uniformemente distribuídos entre zero e um. Como as durações devem ser valores inteiros, o valor fornecido pela função rand foi multiplicado por 100.

Foram utilizadas três instâncias de cada tipo (30, 60, 90 e 120 vértices), totalizando 12 exemplos de tamanhos variados, além de diferentes graus de conexão entre os vértices. Todas as instâncias utilizadas são apresentadas no APÊNDICE B.

6.2 MODELAGEM DE RECURSOS

Os recursos necessários para a realização de cada atividade independem da duração desta. Quando a duração da atividade assume um valor maior, a atividade necessita de menos recursos a cada período. Cabe ressaltar que esta consideração leva a um caso particular do problema conhecido na literatura acadêmica como *Discrete Time-Resource Trade-off Problem* (DTRTP), abordado em detalhes por Demeulemeester, De Reick et al. (2000). No caso desta dissertação, a relação entre recursos e duração é assumida como linear, ao invés de discreta como em muitos outros estudos.

Nos testes realizados, os recursos de todas as instâncias utilizadas foram limitados a 70 unidades por período.

6.3 CONFIGURAÇÃO DOS PARÂMETROS

6.3.1 PARÂMETROS PARA AS METODOLOGIAS FPS-G4 E FPS-SIM

Para o método FPS-SIM, foram realizadas 200 simulações de cenários. Lembrando que cada cenário é composto por um vetor de durações de atividades, escolhidas a partir do sorteio entre os quatro valores que formam o número *fuzzy* da duração. Ou seja, cada cenário apresenta uma duração do projeto. Logo, foram calculadas 200 durações para cada instância.

Além disso, os valores utilizados para os α-cortes foram 0,20 e 0,80. Os valores com a função de pertinência acima de 0,80 apresentam alta probabilidade de estarem no conjunto de valores válidos, enquanto que os valores com a função de pertinência abaixo de 0,20 apresentam baixa probabilidade de estarem no conjunto de valores válidos.

Para o método FPS-G4, foi calculado o valor (t_H da seção 5.6.2) com 90% de concordância com o número *fuzzy* que indica a duração do projeto.

6.3.2 PARÂMETROS PARA A METODOLOGIA BASEADA EM UM ALGORITMO GENÉTICO

Para o método baseado em um algoritmo genético, foram alterados os seguintes parâmetros:

i. Número de gerações;
ii. Taxa de cruzamento;
iii. Taxa de mutação inicial;
iv. Número de gerações sem que haja atualização da taxa de mutação.

Foram realizados testes com 33 configurações de parâmetros para o algoritmo genético, de acordo com a tabela 1.

Tabela 1 - Configurações de parâmetros para o algoritmo genético

Nome	População	Taxa de cruzamento	Taxa de mutação inicial	Máximo de gerações sem atualizar a mutação
AG1	100	0,97	0,04	20
AG2	100	0,97	0,02	30
AG3	100	0,96	0,04	30
AG4	100	0,96	0,03	50
AG5	100	0,95	0,07	50
AG6	100	0,95	0,07	30
AG7	100	0,95	0,05	30
AG8	100	0,95	0,04	20
AG9	100	0,95	0,04	30
AG10	100	0,95	0,03	50
AG11	100	0,95	0,02	30
AG12	100	0,94	0,07	30
AG13	100	0,94	0,04	20
AG14	100	0,94	0,03	50
AG15	100	0,93	0,04	30
AG16	100	0,92	0,04	20
AG17	100	0,9	0,04	50
AG18	100	0,9	0,04	30
AG19	100	0,9	0,04	20
AG20	100	0,9	0,04	40
AG21	100	0,9	0,03	50
AG22	100	0,9	0,03	30
AG23	100	0,85	0,04	50
AG24	100	0,85	0,03	50
AG25	100	0,8	0,06	50
AG26	100	0,8	0,05	50
AG27	100	0,8	0,04	50
AG28	30	0,9	0,03	50
AG29	30	0,9	0,03	40
AG30	30	0,9	0,03	30
AG31	30	0,8	0,04	50
AG32	30	0,8	0,03	50
AG33	30	0,8	0,02	50

6.3.3 PARÂMETROS PARA A METODOLOGIA BASEADA EM PSO

Para o método baseado em PSO, foram alterados os seguintes parâmetros:

i. Limite de velocidade;

ii. Fator de inércia (w);

iii. Fator cognitivo (c_1);

iv. Fator social (c_2).

Foram realizados testes com 46 configurações de parâmetros para o algoritmo de otimização por enxame de partículas, de acordo com a Tabela 2.

Tabela 2 - Configurações de parâmetros para o PSO

Nome	vel_lim	w	c_1	c_2
PSO1	0,8	2,0	2,0	1,0
PSO2	0,8	2,0	1,0	1,0
PSO3	0,7	2,0	1,0	0,8
PSO4	0,7	1,5	1,0	1,0
PSO5	0,7	1,0	2,0	1,0
PSO6	0,7	1,0	2,0	1,0
PSO7	0,7	1,0	2,0	0,8
PSO8	0,7	1,0	1,5	1,0
PSO9	0,7	1,0	1,5	1,0
PSO10	0,7	1,0	1,5	0,8
PSO11	0,7	1,0	1,0	2,0
PSO12	0,7	1,0	1,0	1,0
PSO13	0,7	1,0	1,0	0,9
PSO14	0,7	1,0	1,0	0,8
PSO15	0,7	1,0	1,0	0,8
PSO16	0,7	1,0	1,0	0,7
PSO17	0,7	1,0	1,0	0,1
PSO18	0,7	1,0	0,9	2,0
PSO19	0,7	1,0	0,9	1,0
PSO20	0,6	1,0	0,9	0,9
PSO21	0,5	1,0	0,8	1,0
PSO22	0,5	1,0	0,8	1,0
PSO23	0,5	1,0	0,8	0,9
PSO24	0,5	1,0	0,7	0,8
PSO25	0,5	1,0	0,6	0,8
PSO26	0,5	0,9	2,0	1,0
PSO27	0,5	0,9	2,0	0,8
PSO28	0,5	0,9	2,0	0,8
PSO29	0,5	0,9	2,0	0,7
PSO30	0,5	0,9	1,5	1,0
PSO31	0,4	0,9	1,0	2,0
PSO32	0,4	0,9	1,0	1,0
PSO33	0,4	0,9	1,0	0,8
PSO34	0,4	0,9	1,0	0,7
PSO35	0,4	0,9	0,8	1,0
PSO36	0,4	0,8	2,0	2,0
PSO37	0,3	0,8	2,0	0,8
PSO38	0,3	0,8	2,0	0,8
PSO39	0,3	0,8	1,5	1,0
PSO40	0,3	0,8	1,0	2,0
PSO41	0,3	0,8	1,0	1,0
PSO42	0,2	0,8	1,0	0,9
PSO43	0,2	0,8	1,0	0,8
PSO44	0,2	0,8	1,0	0,8
PSO45	0,2	0,5	1,0	2,0
PSO46	0,2	0,5	1,0	1,0

6.4 RESULTADOS COMPUTACIONAIS E ANÁLISES

O computador utilizado para a execução dos testes tem a seguinte configuração: AMD Turion ® X2 Ultra Dual-Core Mobile ZM-80 2,10 GHz e 4 GB de memória RAM.

A apresentação dos resultados será feita de forma separada por tamanho de projeto. Os gráficos das abordagens heurísticas mostram a distribuição dos valores de duração total do projeto separados por α-cortes, sendo apresentados em uma composição como números *fuzzy* (FPS-G4) e em um histograma (FPS-SIM). Para a comparação das abordagens metaheurísticas, são apresentados os histogramas de duração total do projeto, com os resultados atingidos por todas as configurações de cada método

Após a apresentação dos gráficos, existe uma tabela comparativa, com as seguintes informações:

Para o método FPS-G4:

i. Número fuzzy que representa a duração total do projeto;
ii. Valor exato que tem 90% de concordância com o número *fuzzy* do item (i) - $\mathbf{t_H}$;
iii. Tempo de execução.

Para o método FPS-SIM:

iv. Média das 200 durações totais encontradas pelas simulações;
v. Desvio padrão para a amostra do item (iv) e
vi. $\%t_H$ – Porcentagem dos valores da amostra do item (iv) que são inferiores ao valor $\mathbf{t_H}$, mostrado no item (ii);
vii. Tempo de execução.

Para as abordagens baseada em PSO:

viii. Média das durações totais encontradas sem *buffer* de projeto;
ix. Desvio padrão da amostra sem *buffer*;
x. Média das durações totais encontradas com *buffer* de projeto;
xi. Desvio padrão da amostra sem *buffer*;
xii. Média dos tempos de execução.

Para as abordagens baseada em algoritmo genético:

xiii. Média das durações totais encontradas sem *buffer* de projeto;
xiv. Desvio padrão da amostra sem *buffer*;
xv. Média das durações totais encontradas com *buffer* de projeto;

xvi. Desvio padrão da amostra sem *buffer*;

xvii. Média dos tempos de execução.

A seguir, estão os gráficos que mostram as soluções encontradas segundo os quatro métodos apresentados nesta dissertação. Cada instância apresenta duas atividades fictícias, que representam o início e o final do projeto.

6.5 PROJETOS COM 30 ATIVIDADES

6.5.1 APRESENTAÇÃO DA INSTÂNCIA 30_1 E RESULTADOS ALCANÇADOS

Tabela 3 - Instância 30_1

Atividade (i)	Sucessores			Recursos	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	0	0	0	0
2	10			20	13	16	47	93
3	5	9	15	80	3	39	77	80
4	7	13		70	3	12	30	52
5	6	17		80	3	4	9	23
6	13	16	18	80	10	18	29	62
7	8	11	19	50	6	12	19	54
8	27			90	14	20	34	36
9	14	28		20	4	8	22	86
10	12	22		10	6	21	30	33
11	21	26		20	27	30	33	58
12	24			60	3	13	22	42
13	25			50	3	13	17	27
14	25			100	5	11	18	52
15	21			100	4	13	22	54
16	20	26		100	3	4	11	72
17	21	26	29	70	3	5	22	40
18	23			50	9	19	51	54
19	25			70	3	4	4	5
20	31			30	5	25	31	86
21	31			70	3	10	31	91
22	27	29		100	3	3	4	9
23	31			80	15	17	30	49
24	29			80	8	14	24	73
25	30			70	14	22	27	66
26	27			70	25	33	44	50
27	30			10	7	12	17	37
28	30			80	5	9	11	29
29	32			30	6	11	17	89
30	32			80	3	10	22	44
31	32			60	3	3	29	69
32				0	0	0	0	0

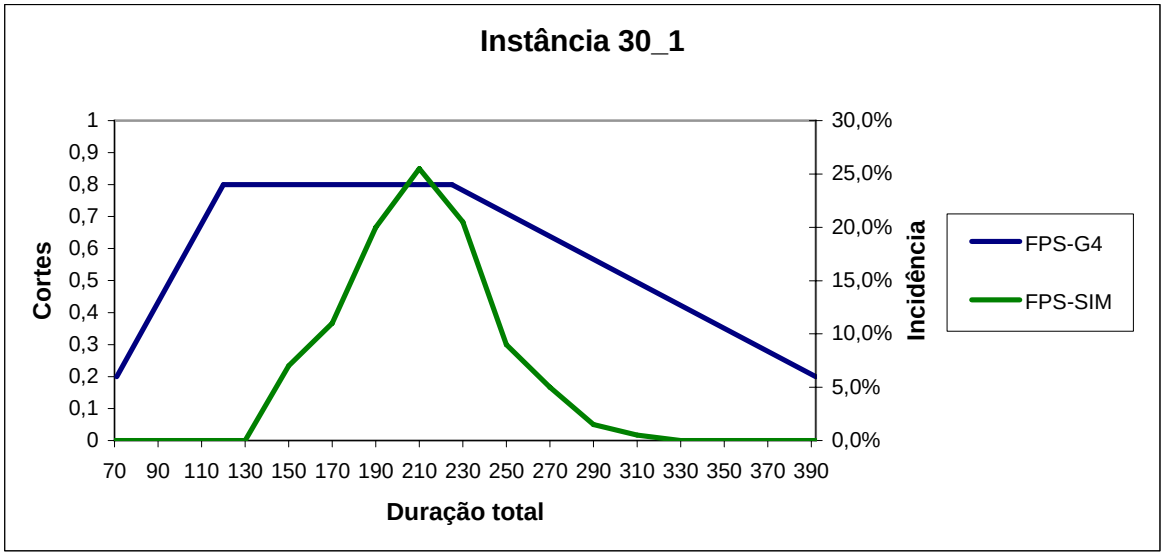


Figura 13 - Soluções de FPS-G4 e FPS-SIM na instância 30_1

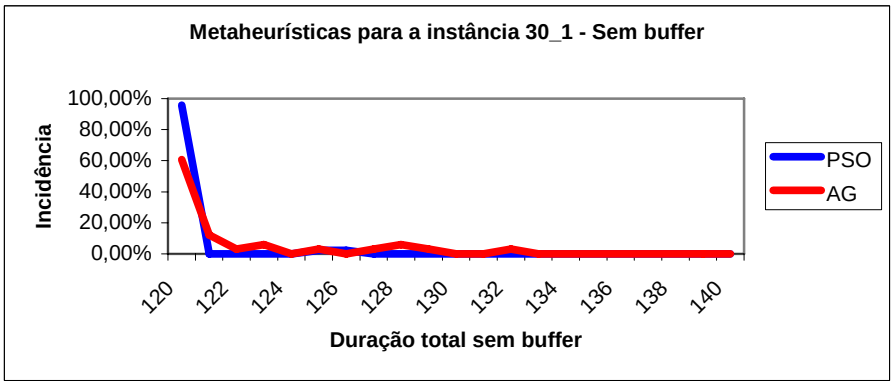


Figura 14 - PSO e AG sem buffer na instância 30_1

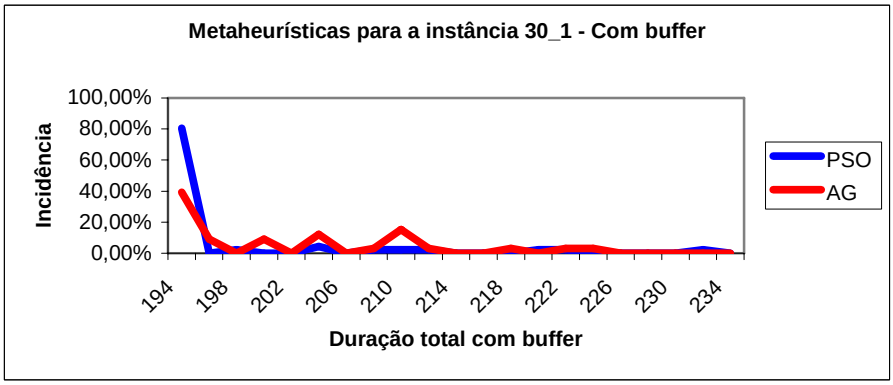


Figura 15 - PSO e AG com buffer na instância 30_1

6.5.2 APRESENTAÇÃO DA INSTÂNCIA 30_2 E RESULTADOS ALCANÇADOS

Tabela 4 - Instância 30_2

Atividade(i)	Sucessores			Recursos	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	0	0	0	0
2	21			120	3	8	10	23
3	5	6		115	14	31	39	66
4	8			85	3	25	54	56
5	7	10	13	135	3	5	9	47
6	15	24		115	3	5	25	71
7	12	23	24	95	6	11	35	68
8	9			140	3	36	37	64
9	11	22		145	3	7	23	68
10	20			155	3	3	13	42
11	17	27		120	4	8	17	89
12	14	16		50	12	22	42	53
13	23	31		100	3	18	36	47
14	26	28		140	3	8	46	60
15	19			60	7	9	14	25
16	18			90	8	14	17	23
17	26	31		130	7	26	80	82
18	27			150	3	3	14	23
19	27	30		50	3	5	29	49
20	28			115	3	34	36	70
21	25	29		110	3	7	18	34
22	24			105	5	24	41	50
23	26	29		105	5	17	20	62
24	29			65	3	6	28	57
25	28			70	3	20	24	36
26	30			55	3	8	13	26
27	31			125	15	21	21	23
28	30			45	6	6	28	38
29	32			105	3	15	45	61
30	32			90	28	28	28	31
31	32			140	16	24	33	71
32				0	0	0	0	0

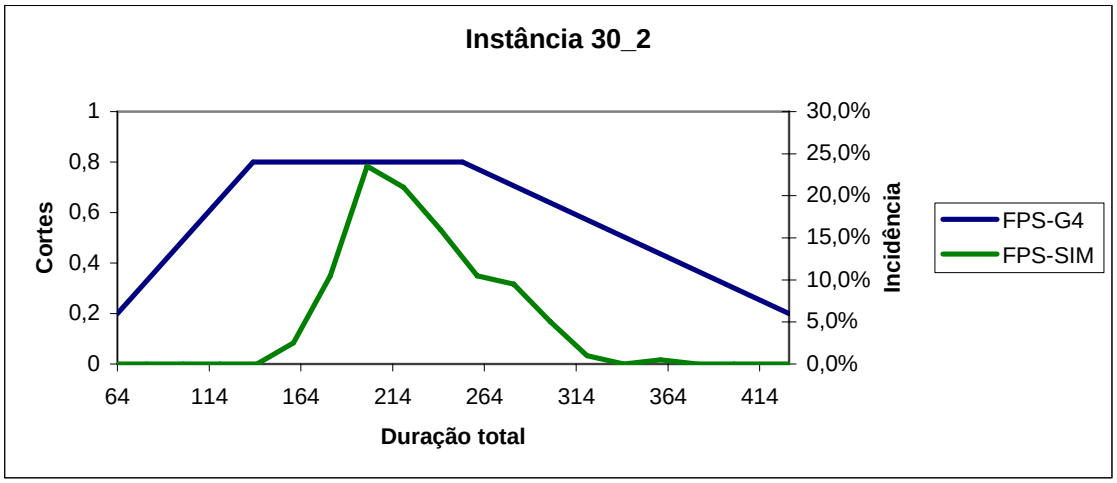


Figura 16 - Soluções de FPS-G4 e FPS-SIM na instância 30_2

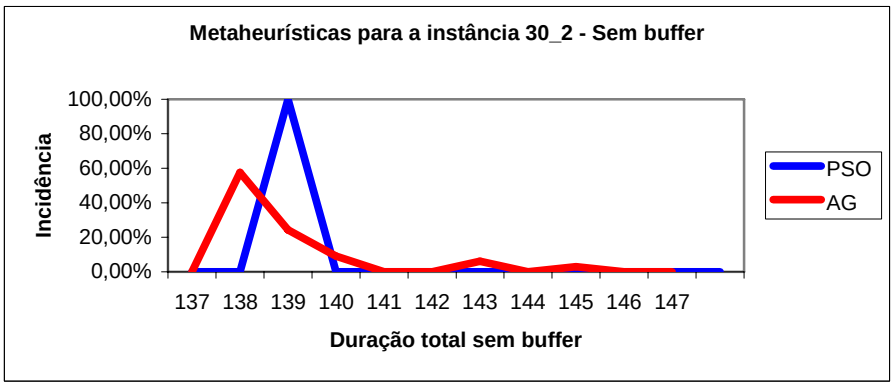


Figura 17 - Soluções de PSO e AG sem buffer para a instância 30_2

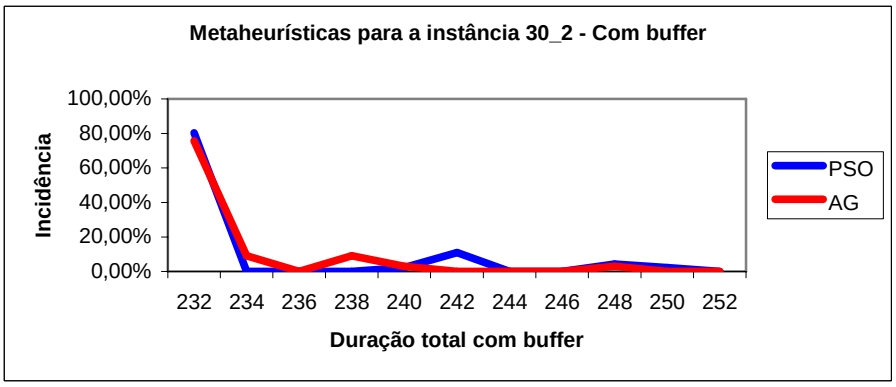


Figura 18 - Soluções de PSO e AG com buffer para a instância 30_2

6.5.3 APRESENTAÇÃO DA INSTÂNCIA 30_3 E RESULTADOS ALCANÇADOS

Tabela 5 - Instância 30_3

Atividade (i)	Sucessores			Recursos	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	4	28	37	74
2	5	12	15	70	3	8	10	23
3	13	14		55	14	31	39	66
4	6	8	20	40	3	25	54	56
5	10	23		90	3	5	9	47
6	7	9	16	75	2	5	25	71
7	10	18	27	95	6	11	35	68
8	18			70	2	36	37	64
9	11	17	19	55	3	7	23	68
10	22	24	26	100	2	2	13	42
11	18			35	4	8	17	89
12	14	19	29	55	12	22	42	53
13	21	25	29	60	3	18	36	47
14	23			35	2	8	46	60
15	16	22	27	95	7	9	14	25
16	19	26		85	8	14	17	23
17	24	26	31	70	7	26	80	82
18	24	29		75	2	3	14	23
19	25			90	3	5	29	49
20	25	31		90	3	34	36	70
21	23			80	3	7	18	34
22	31			25	5	24	41	50
23	30			75	5	17	20	62
24	28			45	2	6	28	57
25	30			60	2	20	24	36
26	28			55	2	8	13	26
27	28			55	15	21	21	23
28	30			80	6	6	28	38
29	32			25	3	15	45	61
30	32			40	28	28	28	31
31	32			80	16	24	33	71
32				0	4	31	36	62

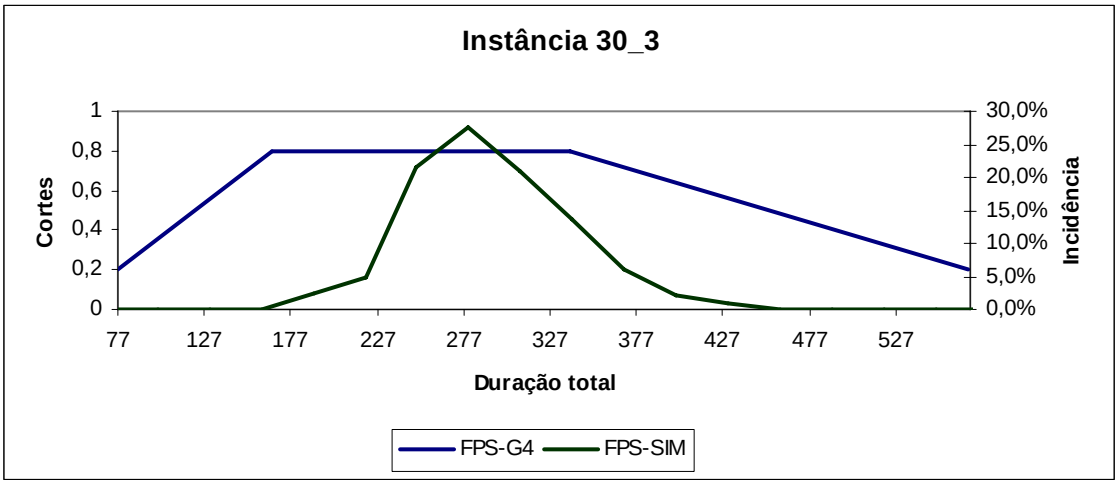


Figura 19 - Soluções de FPS-G4 e FPS-SIM para a instância 30_3

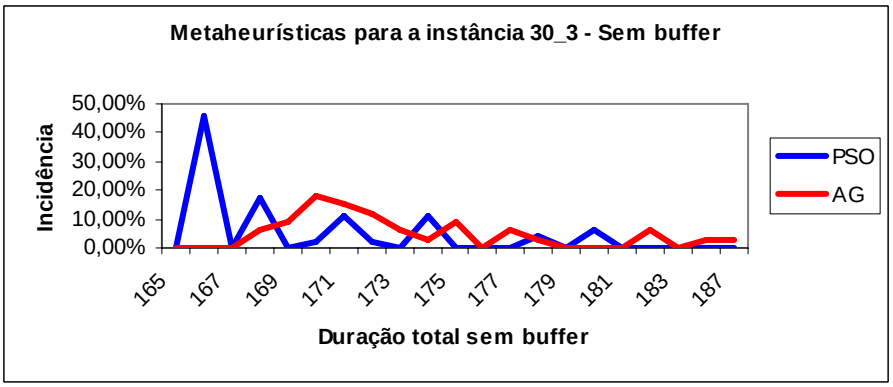


Figura 20 - Soluções de PSO e AG sem buffer para a instância 30_3

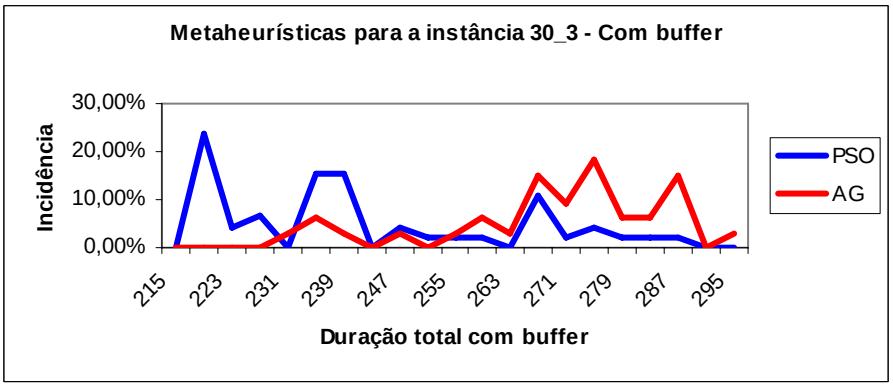


Figura 21 - Soluções de PSO e AG com buffer para a instância 30_3

6.5.4 ANÁLISE DOS RESULTADOS PARA PROJETOS DE 30 ATIVIDADES

Tabela 6 - Comparação de resultados para projetos de 30 atividades

Instâncias	**30_1**	**30_2**	**30_3**
FPS – G4			
i. Duração total *fuzzy*	(71, 120, 225, 392)	(64, 138, 252, 430)	(77, 166, 139, 569)
ii. $\mathbf{t_H}$	308	338	446
iii. Tempo de execução	1s	1s	1s
FPS – SIM			
iv. Média	200,40	219,05	276,31
v. Desvio padrão	33,12	37,07	45,12
vi. % $\mathbf{t_H}$	100%	99,5%	100%
vii. Tempo de execução	3s	3s	3s
PSO			
ix. Média das durações sem buffer	120,24	138,00	169,41
x. Desvio padrão das durações sem buffer	1,14	0	4,39
xi. Média das durações com buffer	197,50	233,43	239,74
xii. Desvio padrão das durações com buffer	8,51	5,24	21,07
xiii. Tempo médio de execução	78,76 s	84,09 s	79,46 s
Algoritmo Genético			
xiv. Média das durações sem buffer	121,85	138,94	173,27
xv. Desvio padrão das durações sem buffer	3,26	1,68	4,75
xvi. Média das durações com buffer	201,45	232,67	267,39
xvii. Desvio padrão das durações com buffer	8,79	3,53	16,42
xviii. Tempo médio de execução	136,79 s	143,09 s	134,00 s

A Tabela 6 permite observar que o método FPS-G4 pode ser útil, para projetos de 30 atividades, quando o gerente precisar saber qual é a duração máxima do projeto, no pior caso. Entretanto, o método FPS-SIM oferece uma projeção mais próxima da realidade, como já era esperado. Cabe lembrar que, se a duração total da solução original (linha i) for transformada em um número exato com 90% de concordância, o

valor encontrado será o $\mathbf{t_H}$. Na linha *vi*, pode-se observar que a quase totalidade das simulações apresentaram duração total do projeto menor que $\mathbf{t_H}$. Isto significa que, mesmo que fosse realizado um número menor de sorteios nas durações, os resultados de programação ainda teriam durações totais menores que a duração *fuzzy* encontrada pela solução original.

Comparando apenas os métodos baseados em metaheurísticas, é possível chegar à conclusão que, para estes projetos de 30 atividades, o método baseado em otimização por enxame de partículas – PSO - fornece melhores resultados que o método baseado em um algoritmo genético. Nas três instâncias de 30 atividades que foram testadas, o método baseado em PSO obteve soluções com menor duração (com e sem *buffer*) e estas soluções estavam mais concentradas. Além disso, as soluções encontradas pelo método baseado em PSO exigiram menor tempo de execução (perto de 50% a menos).

Se a comparação for feita de forma global, entre as quatro metodologias apresentadas, é possível perceber que, em média, o método baseado em PSO tem desempenho um pouco melhor que os outros, apresentando duração mais alta apenas na instância 30_2, onde a solução FPS-SIM tem média 219,05 unidades de tempo (contra 233,43 unidades de tempo do método baseado em PSO).

6.6 PROJETOS COM 60 ATIVIDADES

As instâncias de 60 atividades são apresentadas em detalhes no APÊNDICE B. A seguir, são apresentados os resultados computacionais.

6.6.1 RESULTADOS DA INSTÂNCIA 60_1

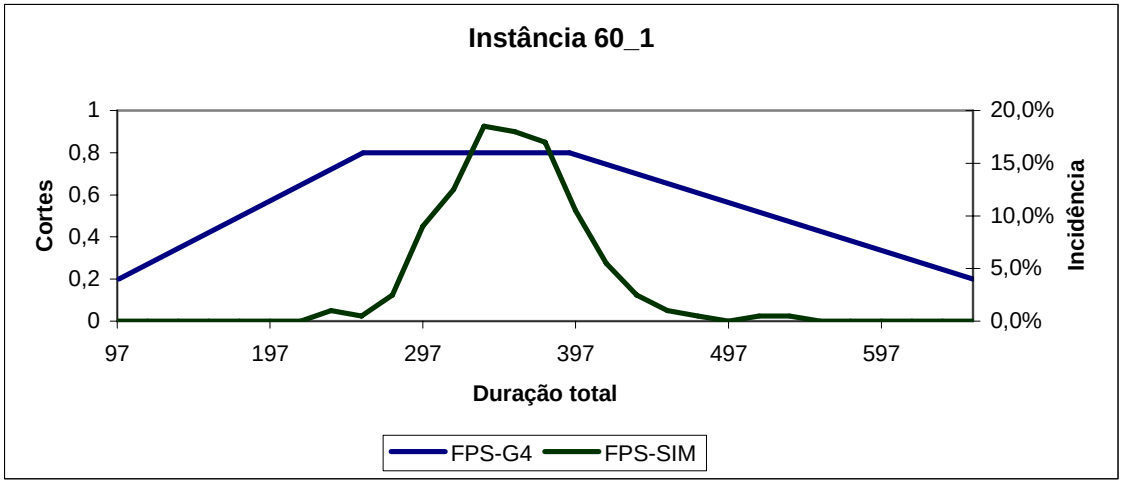


Figura 22 - Soluções dos métodos FPS-G4 e FPS-SIM para a instância 60_1

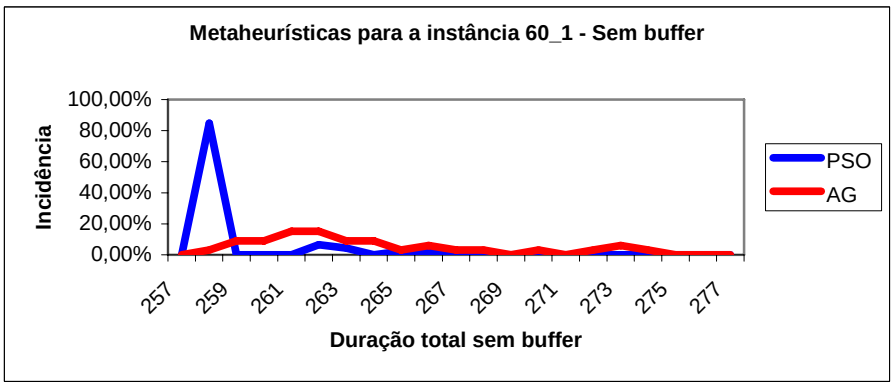


Figura 23 - Soluções de PSO e AG sem buffer para a instância 60_1

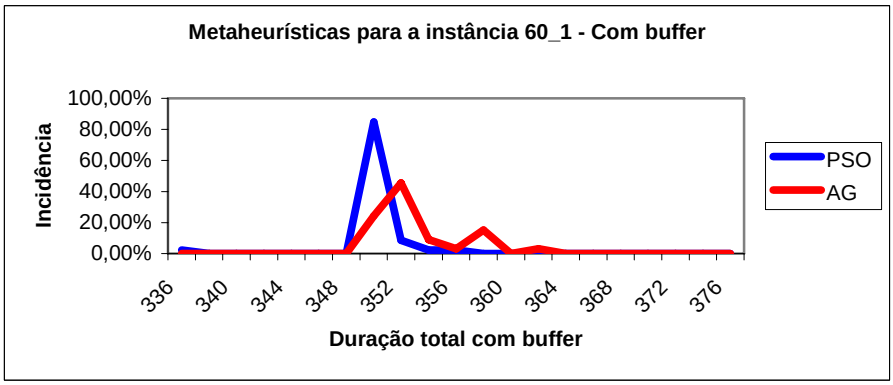


Figura 24 - Soluções de PSO e AG com buffer para a instância 60_1

6.6.2 RESULTADOS DA INSTÂNCIA 60_2

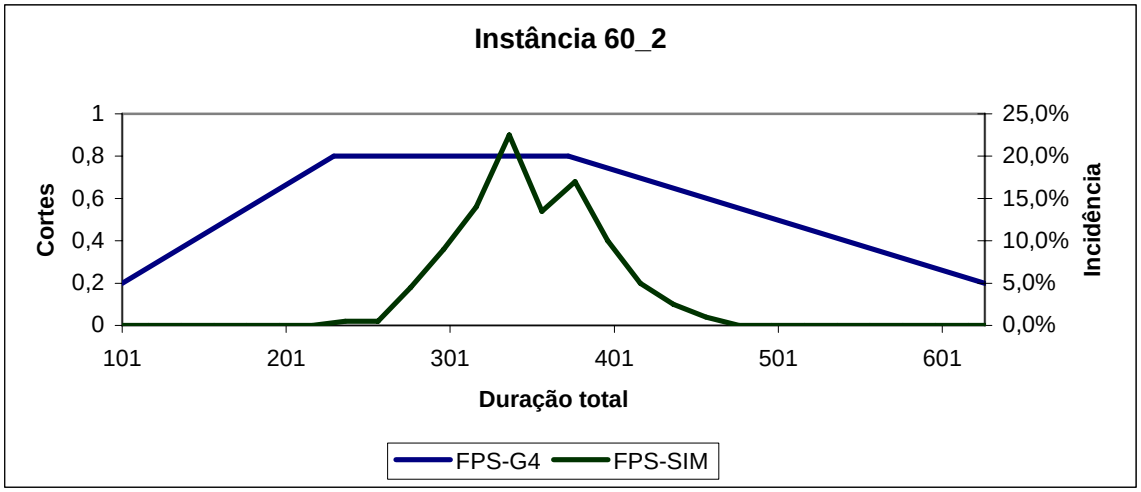


Figura 25 - Soluções de FPS-G4 e FPS-SIM para a instância 60_2

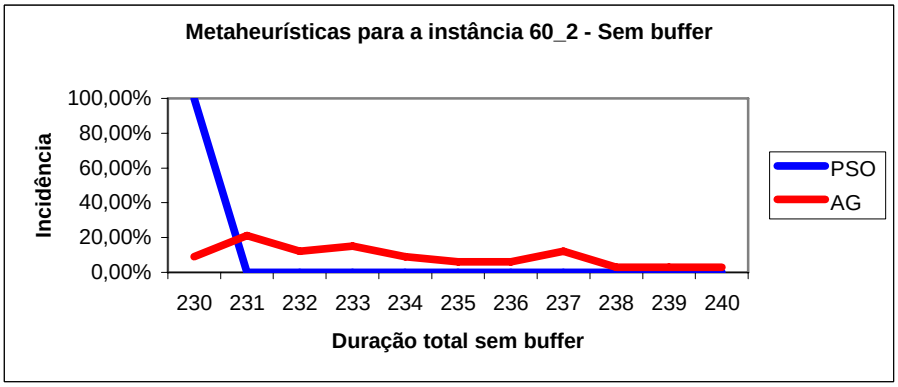


Figura 26 – Soluções de PSO e AG sem buffer na instância 60_2

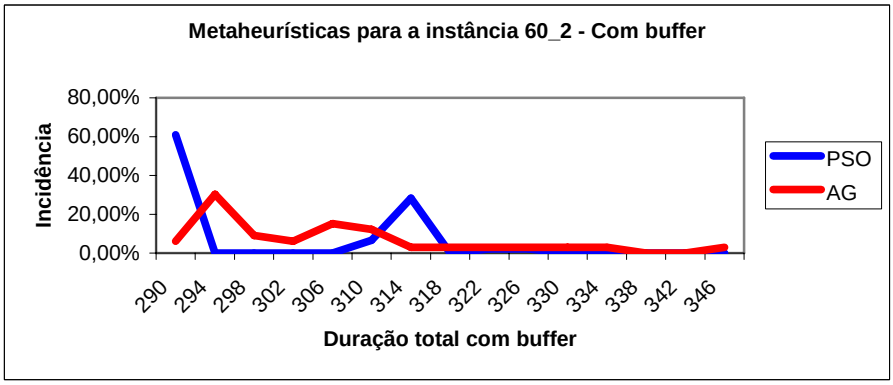


Figura 27 - Soluções de PSO e AG com buffer na instância 60_2

6.6.3 RESULTADOS PARA A INSTÂNCIA 60_3

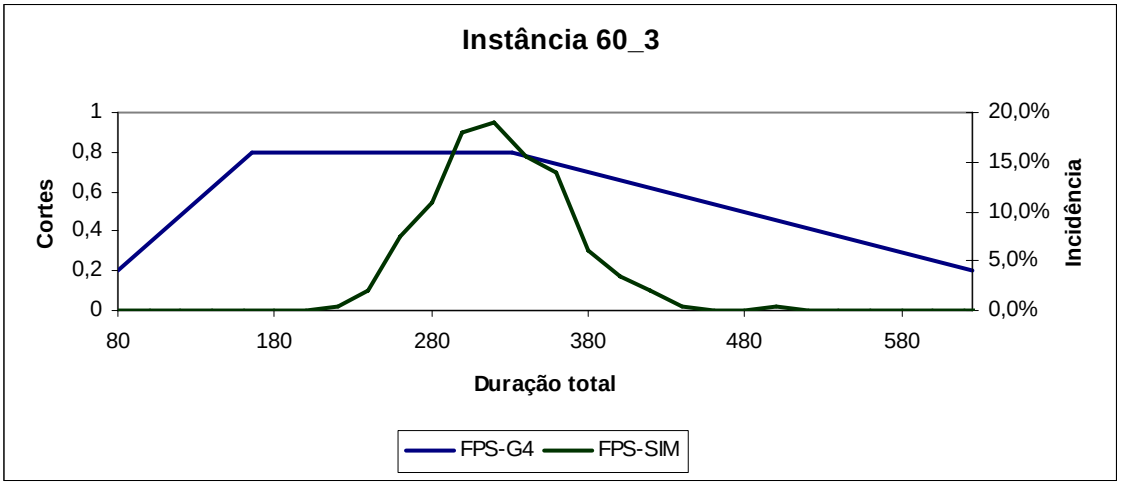


Figura 28 - Soluções de FPS-G4 e FPS-SIM na instância 60_3

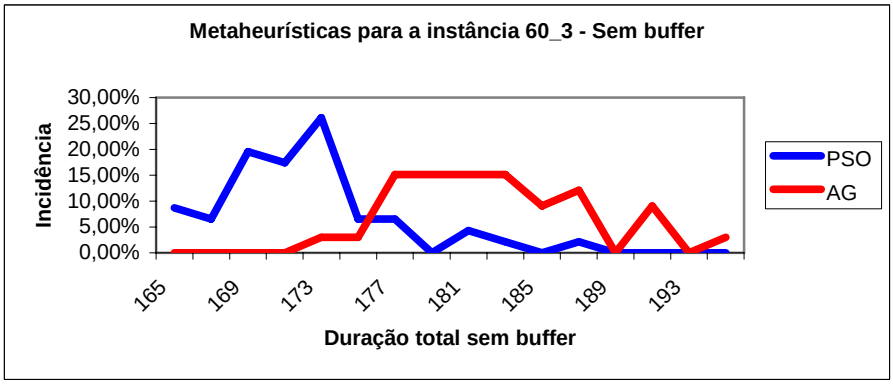


Figura 29 – Soluções de PSO e AG sem buffer na instância 60_3

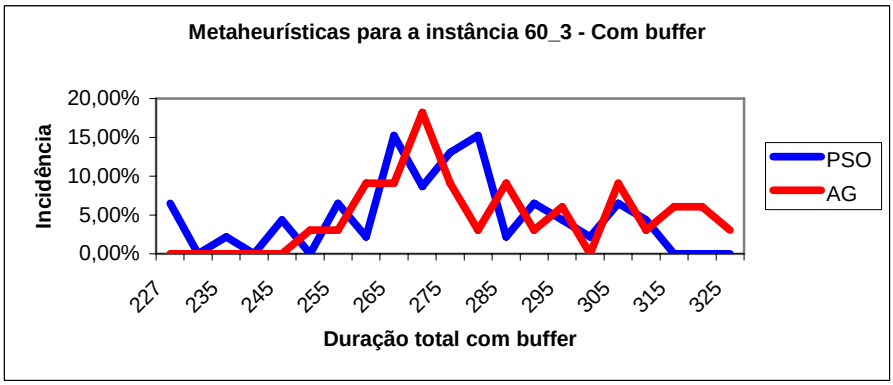


Figura 30 - Soluções de PSO e AG com buffer na instância 60_3

6.6.4 ANÁLISES DOS RESULTADOS PARA PROJETOS DE 60 ATIVIDADES

Tabela 7 - Comparação de resultados para projetos de 60 atividades

Instâncias	**60_1**	**60_2**	**60_3**
FPS – G4			
i. Duração total *fuzzy*	(98, 258, 393, 657)	(101, 230, 373, 627)	(80, 165, 332, 625)
ii. $\mathbf{t_H}$	522	497	481
iii. Tempo de execução	1s	1s	1s
FPS – SIM			
iv. Média	346,15	340,80	314,16
v. Desvio padrão	44,30	40,29	41,83
vi. % $\mathbf{t_H}$	100%	100%	99,5%
vii. Tempo de execução	7s	6s	7s
PSO			
ix. Média das durações sem buffer	258,80	230,00	171,74
x. Desvio padrão das durações sem buffer	2,01	0	4,71
xi. Média das durações com buffer	349,20	298,63	270,93
xii. Desvio padrão das durações com buffer	2,38	11,23	20,72
xiii. Tempo médio de execução	161,15 s	160,87 s	159,28 s
Algoritmo Genético			
xiv. Média das durações sem buffer	263,79	233,61	181,88
xv. Desvio padrão das durações sem buffer	4,41	2,79	5,13
xvi. Média das durações com buffer	352,48	303,82	281,39
xvii. Desvio padrão das durações com buffer	3,17	13,87	21,33
xviii. Tempo médio de execução	276,27 s	280,27 s	278,67 s

As informações contidas na **Erro! Fonte de referência não encontrada.**Tabela 7 e nos gráficos permitem concluir que, para estas instâncias de 60 atividades, o método FPS-G4 novamente é mais relevante se o gerente precisar saber qual é a duração máxima do projeto. Analisando a linha *vi*, observa-se que os valores encontrados no método FPS-SIM são, quase em sua totalidade, mais baixos que os valores $\mathbf{t_H}$ do

método FPS-G4. Este fato confirma, assim como nos projetos de 30 atividades, que o método FPS-SIM fornece soluções com menores durações que o método FPS-G4.

Analisando os métodos baseados em metaheurísticas, o método baseado em PSO teve um desempenho melhor que o método baseado em algoritmo genético. Excetuando-se a instância 60_3, onde os resultados são muito dispersos e pouco conclusivos para os dois métodos, o método baseado em PSO fornece soluções mais concentradas em valores mais baixos de duração total, sem ou com *buffer* de projeto. Porém, a diferença entre as soluções encontradas pelos dois métodos não é muito grande, ficando perto de 2%. Entretanto, observando o tempo de execução dos dois métodos, o baseado em PSO se mostra mais rápido em aproximadamente 40%. Um fato interessante aconteceu na instância 60_2, em que todas as configurações do método baseado em PSO forneceram a mesma solução sem buffer (230 unidades de tempo). Mesmo com este fato, o método baseado no algoritmo genético apresentou solução semelhante, com média de 233,61 unidades de tempo.

Pelos fatos analisados acima, é possível concluir que o método baseado em PSO se mostrou mais indicado para resolver as instâncias testadas de projetos com 60 atividades.

6.7 PROJETOS COM 90 ATIVIDADES

As instâncias de 90 atividades são apresentadas em detalhes no APÊNDICE B. A seguir, são apresentados os resultados computacionais.

6.7.1 RESULTADOS PARA A INSTÂNCIA 90_1

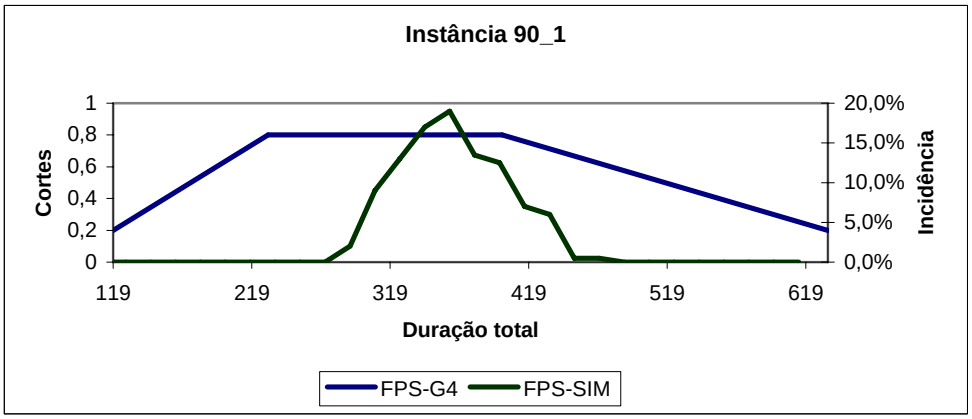


Figura 31 - Soluções de FPS-G4 e FPS-SIM na instância 90_1

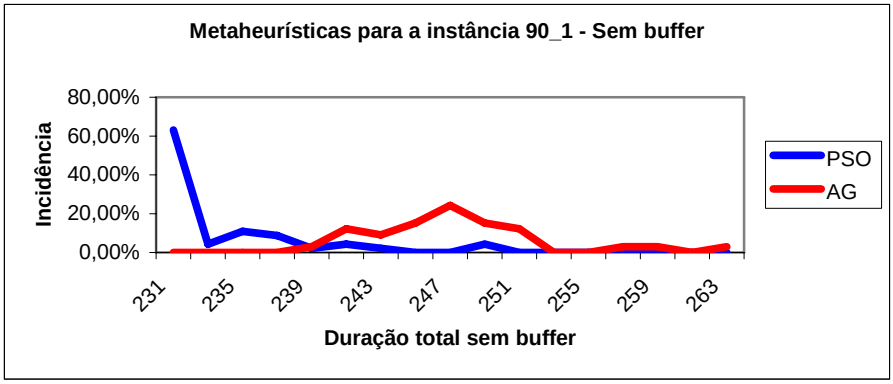


Figura 32 - Soluções de PSO e AG sem buffer na instância 90_1

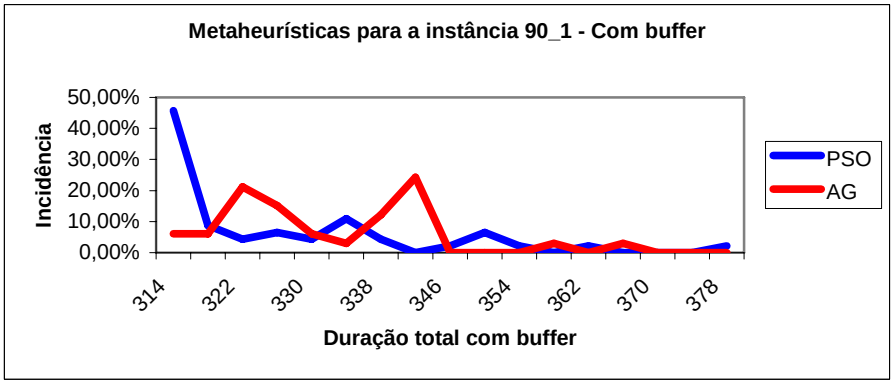


Figura 33 - Soluções de PSO e AG com buffer na instância 90_1

6.7.2 RESULTADOS DA INSTÂNCIA 90_2

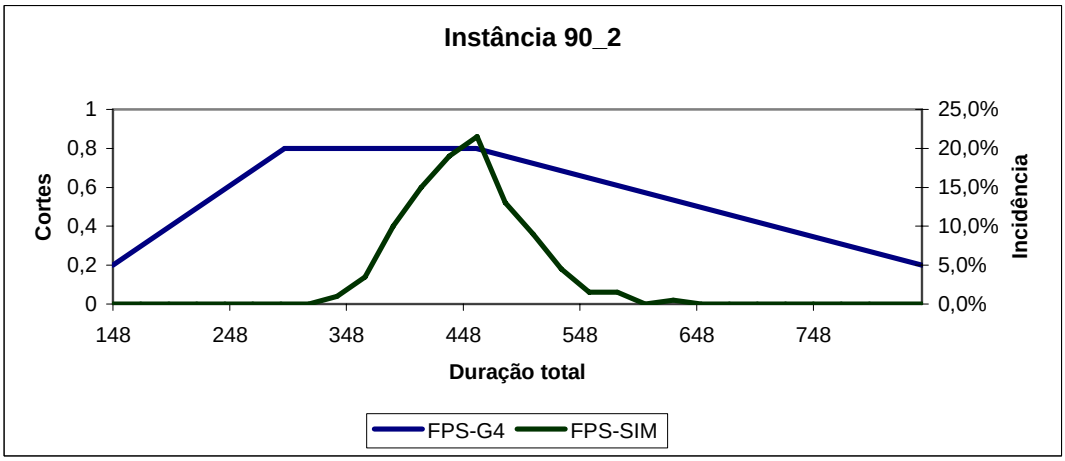


Figura 34 - Soluções de FPS-G4 e FPS-SIM na instância 90_2

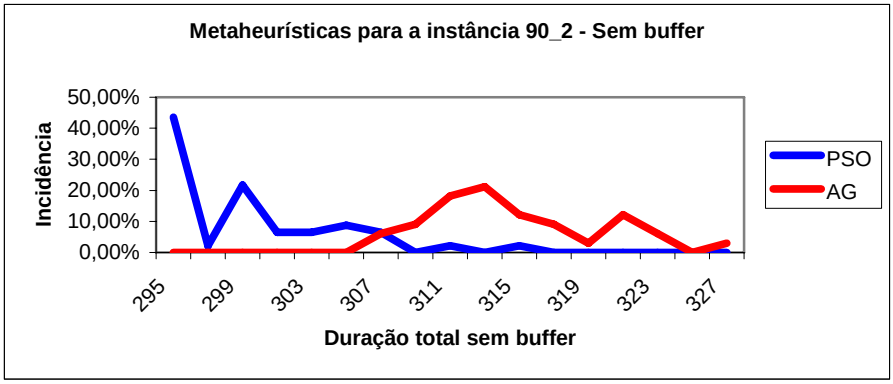


Figura 35 - Soluções de PSO e AG sem buffer na instância 90_2

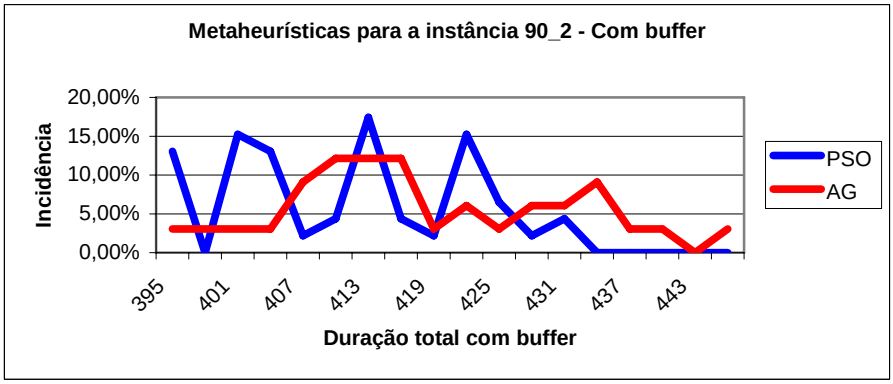


Figura 36 - Soluções de PSO e AG com buffer na instância 90_2

6.7.3 RESULTADOS DA INSTÂNCIA 90_3

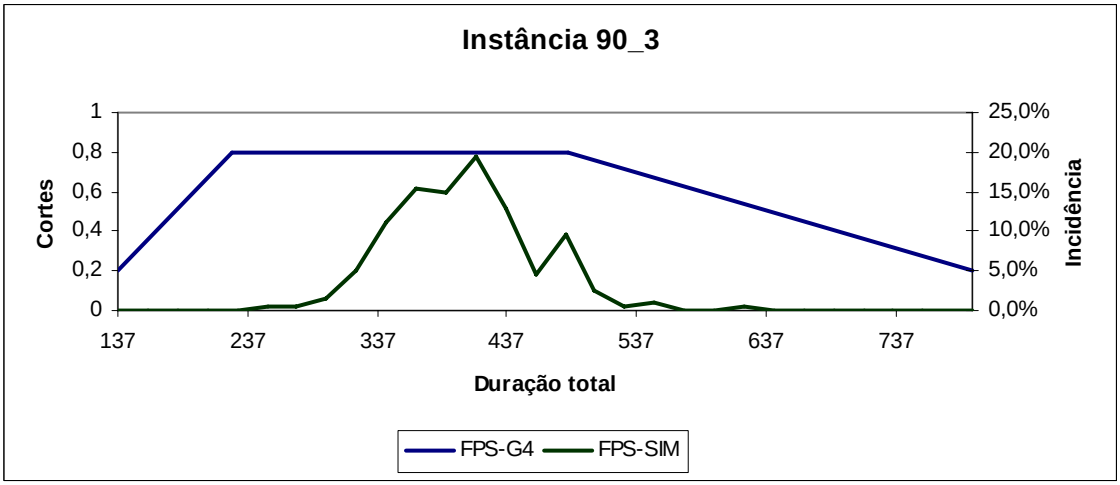


Figura 37 - Soluções de FPS-G4 e FPS-SIM na instância 90_3

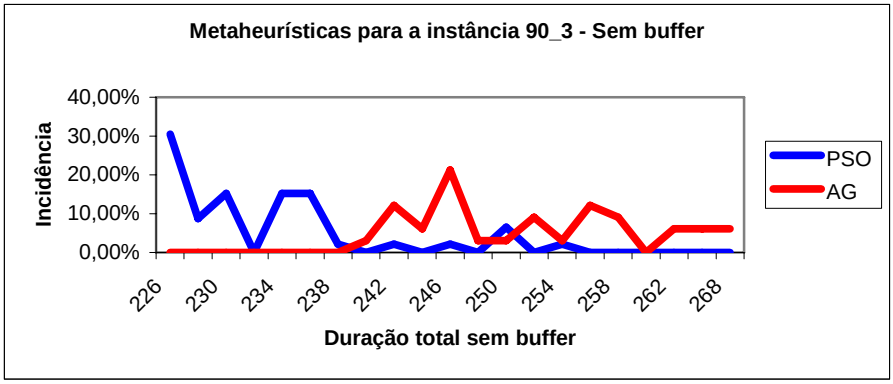


Figura 38 - Soluções de PSO e AG sem buffer na instância 90_3

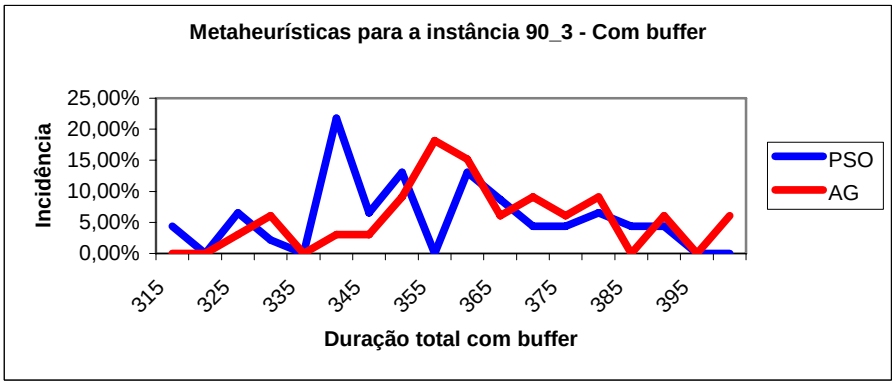


Figura 39 - Soluções de PSO e AG com buffer na instância 90_3

6.7.4 ANÁLISE DOS RESULTADOS PARA PROJETOS DE 90 ATIVIDADES

Tabela 8 - Comparação entre os resultados para projetos de 90 atividades

Instâncias	**90_1**	**90_2**	**90_3**
FPS – G4			
i. Duração total *fuzzy*	(119, 231, 400, 635)	(148, 295, 459, 841)	(137, 225, 484, 796)
ii. $\mathbf{t_H}$	509	660	627
iii. Tempo de execução	1s	1s	1s
FPS – SIM			
iv. Média	355,41	439,44	392,83
v. Desvio padrão	37,14	48,19	54,57
vi. % $\mathbf{t_H}$	100%	100%	100%
vii. Tempo de execução	11s	12s	11s

PSO			
ix. Média das durações sem buffer	233,46	299,00	232,04
x. Desvio padrão das durações sem buffer	4,48	4,96	7,55
xi. Média das durações com buffer	324,98	410,52	351,57
xii. Desvio padrão das durações com buffer	15,20	10,69	19,26
xiii. Tempo médio de execução	260,22 s	259,83 s	260,07 s
Algoritmo Genético			
xiv. Média das durações sem buffer	246,76	314,12	251,45
xv. Desvio padrão das durações sem buffer	4,96	5,05	8,12
xvi. Média das durações com buffer	329,52	417,61	360,27
xvii. Desvio padrão das durações com buffer	13,59	12,68	18,46
xviii. Tempo médio de execução	446,91 s	454,00 s	454,45

As informações da Tabela 8 e os gráficos permitem concluir, mais uma vez, que o método FPS-G4 é indicado para se descobrir o pior caso possível. Entretanto, o método FPS-SIM fornece soluções com menores durações totais, o que o torna mais atraente. Analisando os resultados das abordagens metaheurísticas para projetos com 90 atividades, percebe-se que os métodos forneceram durações totais com *buffer* de projeto muito dispersas. Para as instâncias testadas de projetos com 90 atividades, o método baseado em PSO forneceu soluções bem melhores, com durações totais quase 10% menores que as fornecidas pelo método baseado em algoritmo genético. Além, disso, novamente o método baseado em PSO foi mais rápido em sua execução, com tempo médio até 40% menor que o método baseado em algoritmo genético. Pelos motivos explicados acima, o método baseado em PSO se mostrou mais indicado para resolver as instâncias de projetos com 90 atividades que foram testadas. Entretanto, é importante perceber que a configuração de parâmetros deve ser feita com cuidado, uma vez que os métodos forneceram soluções pouco concentradas.

6.8 PROJETOS COM 120 ATIVIDADES

As instâncias de 120 atividades são apresentadas em detalhes no APÊNDICE B. A seguir, são apresentados os resultados computacionais.

6.8.1 RESULTADOS DA INSTÂNCIA 120_1

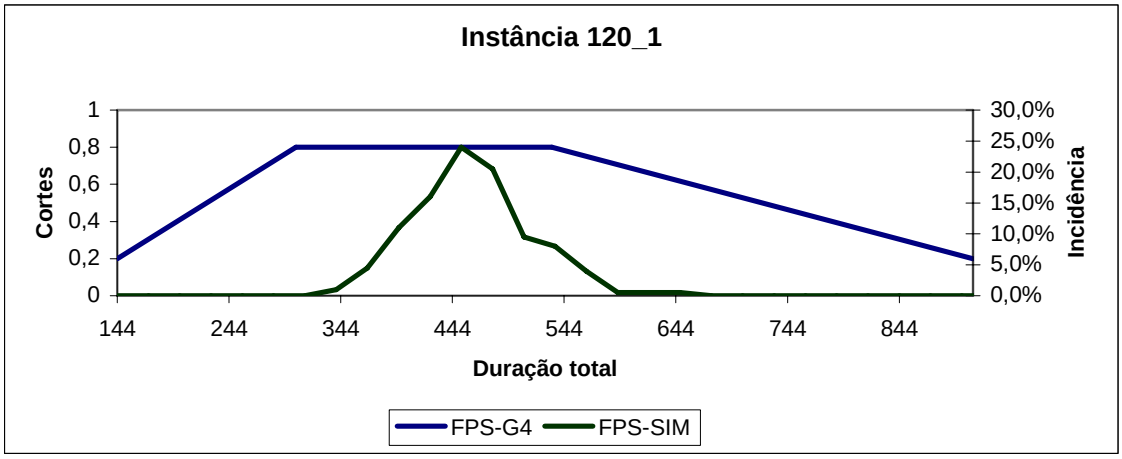


Figura 40 - Soluções de FPS-G4 e FPS-SIM na instância 120_1

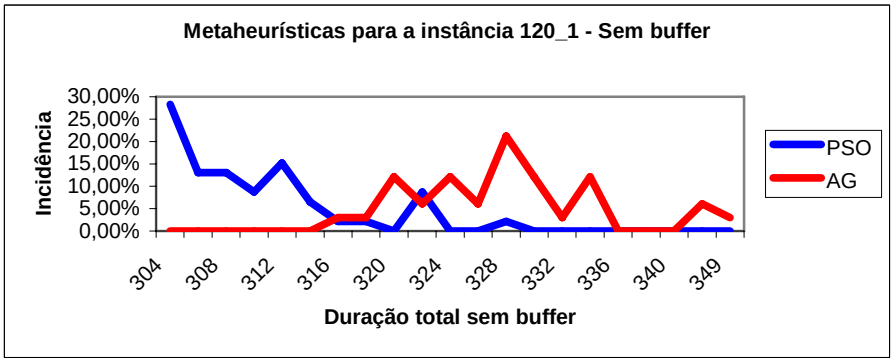


Figura 41 - Soluções de PSO e AG sem buffer na instância 120_1

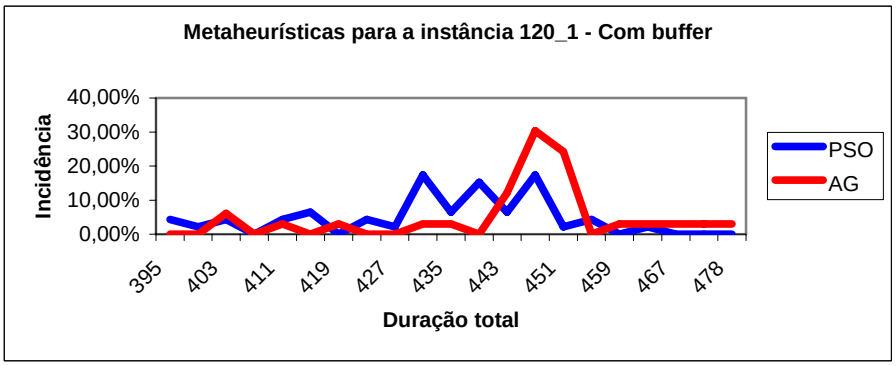


Figura 42 - Soluções de PSO e AG com buffer na instância 120_1

6.8.2 RESULTADOS DA INSTÂNCIA 120_2

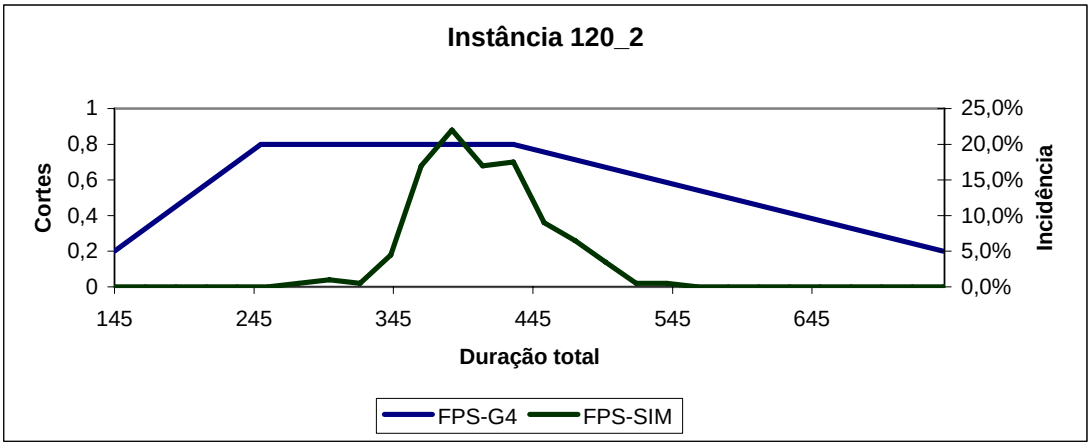


Figura 43 - Soluções de FPS-G4 e FPS-SIM na instância 120_2

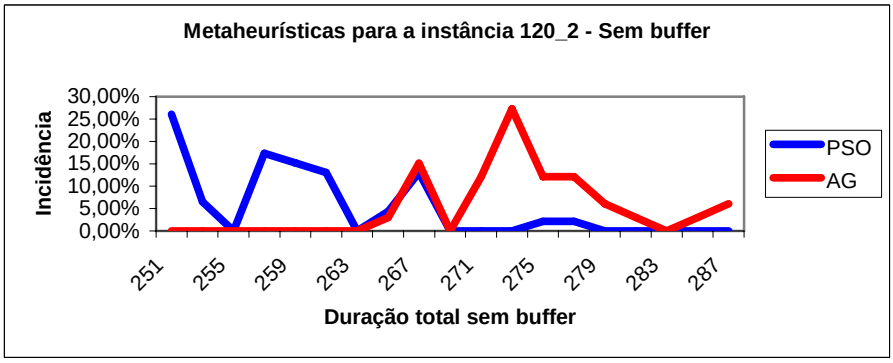


Figura 44 - Soluções de PSO e AG sem buffer na instância 120_2

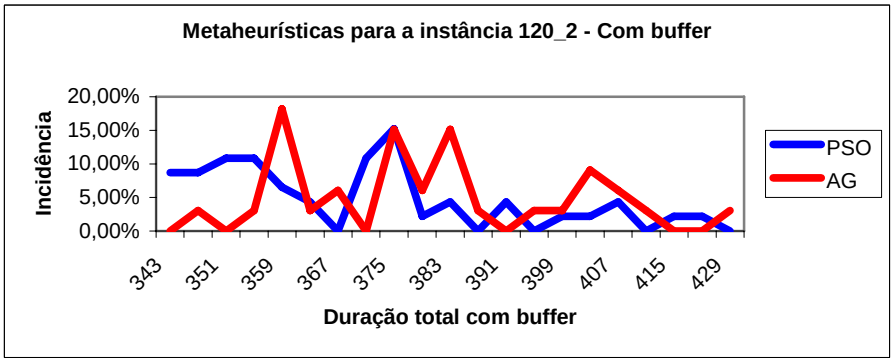


Figura 45 - Soluções de PSO e AG com buffer na instância 120_2

6.8.3 RESULTADOS DA INSTÂNCIA 120_3

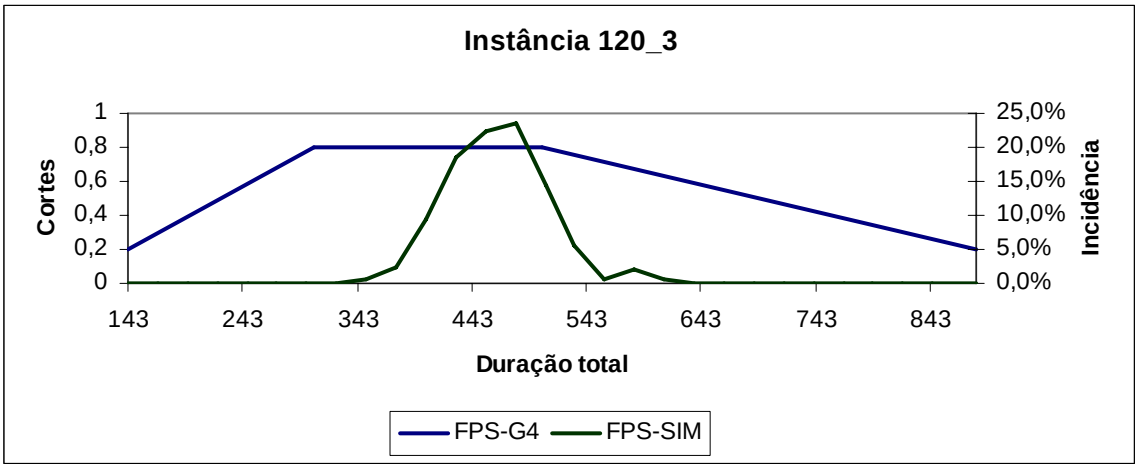


Figura 46 – Soluções de FPS-G4 e FPS-SIM na instância 120_3

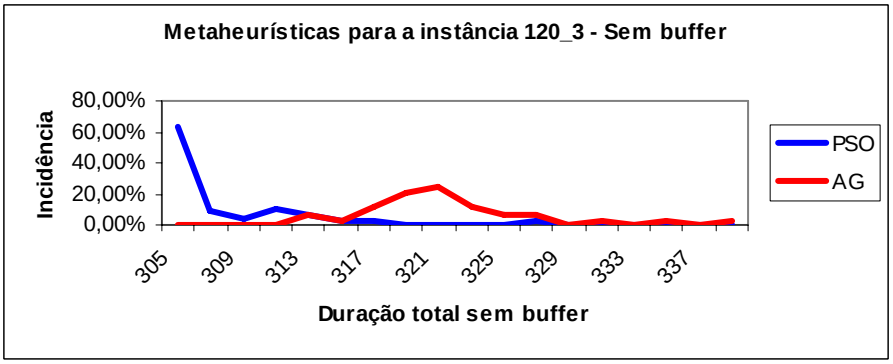


Figura 47 - Soluções de PSO e AG sem buffer na instância 120_3

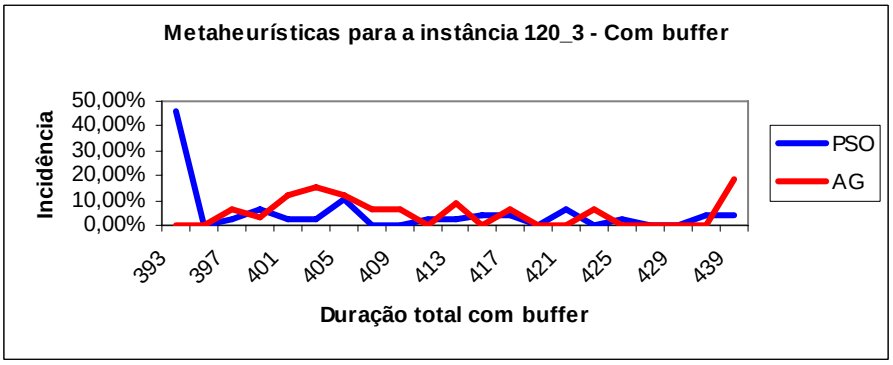


Figura 48 - Soluções de PSO e AG com buffer na instância 120_ 3

6.8.4 ANÁLISES DOS RESULTADOS PARA PROJETOS DE 120 ATIVIDADES

Tabela 9 - Comparação entre os resultados para projetos de 120 atividades

Instâncias	**120_1**	**120_2**	**120_3**
FPS – G4			
i. Duração total *fuzzy*	(144, 304, 533, 910)	(145, 250, 431, 739)	(143, 305, 504, 883)
ii. $\mathbf{t_H}$	717	585	695
iii. Tempo de execução	1s	1s	1s
FPS – SIM			
iv. Média	448,02	397,54	452,39
v. Desvio padrão	52,08	42,32	43,37
vi. % $\mathbf{t_H}$	100%	100%	100%
vii. Tempo de execução	16s	17s	16s
PSO			
ix. Média das durações sem buffer	309,50	257,98	307,26
x. Desvio padrão das durações sem buffer	6,00	6,72	4,06
xi. Média das durações com buffer	430,89	367,39	403,20
xii. Desvio padrão das durações com buffer	16,78	20,58	13,77
xiii. Tempo médio de execução	365,65 s	359,72 s	407,20 s
Algoritmo Genético			
xiv. Média das durações sem buffer	327,48	273,52	321,00
xv. Desvio padrão das durações sem buffer	7,26	5,53	5,57
xvi. Média das durações com buffer	443,97	378,97	411,39
xvii. Desvio padrão das durações com buffer	16,85	19,66	12,70
xviii. Tempo médio de execução	643,30 s	630,85 s	788,55 s

As informações nos gráficos e na Tabela 9 permitem conclusões, para projetos de 120 atividades, muito semelhantes às relativas aos projetos de 30, 60 e 90 atividades. O método FPS-G4 fornece a solução com o pior caso, o que pode ser útil, dependendo do interesse do gerente de projeto.

Analisando os resultados fornecidos pelos métodos baseados em metaheurísticas, as durações totais sem *buffer* calculadas pelo método baseado em PSO foram melhores, em

cerca de 10%, que os calculados pelo método baseado em algoritmo genético. Além disso, as soluções do método baseado em PSO se mostraram mais concentradas, o que permite uma menor dependência da configuração de parâmetros.

Ao analisar as durações totais com *buffer*, a instância 120_1 teve resultados que apontam para uma quase equivalência entre os métodos comparados. A instância 120_2 teve resultados que revelam um comportamento instável dos dois métodos, com uma leve tendência de melhores resultados para o método baseado em PSO. Finalmente, a instância 120_3 apresenta uma clara dominância do método baseado em PSO em relação ao método baseado em algoritmo genético.

Além das soluções, se for analisado o tempo de execução, o método baseado em PSO se mostrou bem mais rápido, para as instâncias de 120 atividades testadas. Seu tempo médio de execução foi até 48% menor que o tempo médio do método baseado em algoritmo genético.

Por conta dos fatos analisados acima, o método baseado em PSO também se mostrou mais indicado para resolver as instâncias de projetos com 120 atividades.

7. CONCLUSÕES E TRABALHOS FUTUROS

No ambiente corporativo competitivo atual, é importante que os projetos sejam bem planejados e gerenciados para evitar desperdícios de tempo e recursos. O objetivo desta dissertação foi apresentar alternativas para a programação de projetos com restrições de recursos e incertezas.

Esta dissertação apresenta quatro métodos diferentes para a resolução do problema: FPS-G4, FPS-SIM, algoritmo genético com buffer e PSO com buffer. Em todos os métodos propostos, existe uma associação do problema com durações de atividades *fuzzy* com problemas determinísticos. Para resolver cada problema determinístico, é realizado o seqüenciamento das atividades e a obtenção da duração total do projeto. No método FPS-G4, a duração *fuzzy* é transformada em um valor determinado (*crisp*) que pode ser associado a ela. No método FPS-SIM, é calculada a média entre todos os valores de duração total que foram encontrados nos cenários simulados. Nos métodos baseados em algoritmo genético com buffer e PSO com buffer, a duração total do projeto é usada como função de performance para avaliação e evolução dos indivíduos.

Para avaliar e validar as metodologias apresentadas, foram realizados testes com 12 instâncias de projetos, de tamanhos variáveis (30, 60, 90 e 120 atividades). Com as análises feitas com base nos resultados mostrados no capítulo 6, pode-se concluir que, caso seja escolhido um método metaheurístico, o baseado em PSO é mais indicado para resolver os problemas de programação de projetos com restrição de recursos e tratamento de incertezas com lógica fuzzy. Entretanto, caso seja escolhido um método heurístico, que tem tempo de execução ínfimo se comparado aos tempos de execução dos métodos metaheurísticos, o mais indicado é o de solução FPS-SIM, que apresenta resultados mais relevantes e próximos da realidade.

Algumas sugestões de trabalhos futuros são a extensão do estudo para múltiplos recursos e também para diferentes tipos de recursos (renováveis e não renováveis). Além disso, sugere-se a implementação do problema em um software de programação matemática para que possam ser comparados os resultados encontrados por métodos metaheurísticos com os resultados exatos fornecidos pelo software.

8. REFERÊNCIAS BIBLIOGRÁFICAS

Abbasi, B., S. Shadrokh, et al. (2006). "Bi-objective resource-constrained project scheduling with robustness and makespan criteria." Applied Mathmatics and Computation.

Alba, E. e J. F. Chicano (2007). "Software project management with GAs." Informations Sciences.

Aloulou, M. A., M. C. Portmann, et al. (2002). Predictive-reactive scheduling for the single machine problem. 8th Workshop on Project Management and Scheduling. Valencia, Spain.

Baldwin, F. F. e N. C. F. Guild (1979). "Comparison of fuzzy sets on the same decision space." Fuzzy Sets and Systems **2**: 213 - 231.

Carlier, J. e E. Néron (2007). "Computing redundant resources for the resource-constrained project scheduling problem." European Journal of Operational Research.

Chang, C. K., M. Christensen, et al. (2001). "Genetic algorithms for project management." Annals of Software Engineering **11**: 107-139.

Chang, C. K., H. Jiang, et al. (2008). "Time-line based model for software project scheduling with genetic algorithms." Information and Software Technology.

Charnes, A. e W. W. Cooper (1959). "Chance-constrained programming." Management Science **6**(1): 73-79.

Chtourou, H. e M. Haouari (2008). "A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling." Computers and industrial engineering.

Conway, R. W., W. L. Maxwell, et al. (1967). Theory of Scheduling. Nova Iorque, John Wiley.

Czyzak, P. e R. Slowinski (1991). "FLIP - multiobjective fuzzy linear programming package." IIASA.

Demeulemeester, E., B. De Reick, et al. (2000). "The discrete time/resource trade-off problem in project networks: a branch-and-bound approach." IIE Transactions **32**: 1059-1069.

Demeulemeester, E. e W. Herroelen (2002). Project Scheduling – A Research Handbook. Boston, Kluwer Academic Publishers.

Dubois, D. e H. Prade (1987). Theories des Possibilities: Application a la Representation des Connaissances en Informatique, Masson.

Eberhart, R. e J. Kennedy (1995). Particle swarm optimization. IEEE International Conference on Neural Networks**:** 1942–1948.

Fernandez, A. A. (1995). The optimal solution to the resource constrained project scheduling problem with stochastic task durations., University of Central Florida. **Unpublished Doctoral Dissertation.**

Fernandez, A. A., R. L. Armacost, et al. (1996). "The role of the non-anticipativity constraint in commercial software for stochastic project scheduling." Computers and Industrial Engineering **31**: 233-236.

Fernandez, A. A., R. L. Armacost, et al. (1998). "Understanding simulation solutions to resource-constrained project scheduling problems with stochastic task durations." Engineering Management Journal **10**: 5-13.

Gao, H. (1995). Building robust schedules using temporal protection—an empirical study of constraint based scheduling under machine failure uncertainty. Department of Industrial Engineering. Toronto, Canada., University of Toronto. **Master's**.

Goldratt, E.M. (1997). Critical Chain. North River Press, Massachusetts.

Hall, N. e M. Posner (2000a). Sensitivity analysis for intractable scheduling problems, The Ohio State University.

Hall, N. e M. Posner (2000b). Sensitivity analysis for efficiently solvable scheduling problems, The Ohio State University.

Hapke, M., A. Jaskievicz, et al. (1999). "Fuzzy multimode resource-constrained project scheduling with multiple objectives." Project Scheduling – Recent Models, Algorithms and Applications.

Hapke, M., R. Slowinski, et al. (1994). "Fuzzy project scheduling system for software development." Fuzzy Sets and Systems **67**: 101-117.

Herroelen, W. e E. Demeulemeester (1994). "Resource-Constrained Project Scheduling. A View on Recent Developments." Tijdschrift voor Economie em Management **XXXIX, 4**.

Herroelen, W. e R. Leus (2004). "The construction of stable project baseline schedules." European Journal of Operational Research **156**: 550-565.

Herroelen, W. e R. Leus (2005). "Project scheduling under uncertainty: Survey and research potentials." European Journal of Operational Research.

Holland, H. J. (1975). Adaptation in Natural and Artificial Systems, The University of Michigan Press.

Kaufman, A. e M. M. Gupta (1985). Introduction to fuzzy, theory and application, Van Nostrand Reinhold.

Ke, H. e B. Liu (2005). "Project scheduling problem with stochastic activity duration times." Applied Mathmatics and Computation **168**: 342-353.

Ke, H. e B. Liu (2007). "Project scheduling problem with mixed uncertainty of randomness and fuzziness." European Journal of Operational Research: 135-147.

Kelley, J. E. e M. R. Walker (1959). Critical Path Planning and Scheduling. Proceedings of the Eastern Joint Computer Conference.

Kim, K. W., M. Gen, et al. (2003). "Hybrid genetic algorithm with fuzzy logic for resource-constrained project scheduling." Applied Soft Computing.

Lambrechts, O., E. Demeulemeester, et al. (2008). "A tabu search procedure for developing robust predictive project schedules." International Journal of Production Economics **111**.

Long, L. D. e A. Ohsato (2007). "Fuzzy critical chain method for project scheduling under resource constraints and uncertainty." International Journal of Project Management.

Loterapong, P. e O. Moselhi (1996). "Project network analysis using fuzzy sets theory." J Constr Manage **122**(4): 114-123.

Malcom, D. G., J. H. Roseboom, et al. (1959). "Application of a Technique for Research and Development Program Evaluation." Operations Research **7**.

Mauguière, P., J. C. Billaut, et al. (2002). Grouping jobs on a single machine with heads and tails to represent a family of dominant schedules. 8th Workshop on Project Management and Scheduling. . Valencia, Spain.

Mika, M., G. Waligóra, et al. (2005). "Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models." European Journal of Operational Research **164**.

Ozdamar, L. e E. Alanya (2001). "Uncertainty Modelling in Software Development Projects (With Case Study)." Annals of Operations Research.

Pacheco, M. A. C. (2005). Algoritmos Genéticos: Princípios e Aplicações. Curso de Introdução aos Algoritmos Genéticos. Rio de Janeiro, ICA – Laboratório de Computação Aplicada – PUC-Rio.

Pan, N., P. Hsaio, et al. (2007). "A study of project scheduling optimization using Tabu Search algorithm." Engineering Applications of Artificial Intelligence.

Patrick, S.F. (1999). Critical chain scheduling and buffer management, getting out from between parkinson_s rock and murphy_s hardplace. PM Network, April.

PMI (2000). Project Management Body of Knowledge - PMBOK, Project Institute Management.

Rommelfanger, H. (1990). FULPAL: An Interactive Method for Solving (Multiobjective) Fuzzy Linear Programming Problems. Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming Under Uncertainty., Dordrecht, Kluwer Academic Publishers.

Roubens, M. (1990). Inequality constraints between fuzzy numbers and their use in mathematical programming. Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty. Dordrecht, Kluwer Academic Publishers**:** 321 - 330.

Sabuncuoglu, I. e M. Bayiz (2000). "Analysis of reactive scheduling problems in a job shop environment." European Journal of Operational Research **126**: 567-586.

Sadeh, N., S. Otsuka, et al. (1993). Predictive and reactive scheduling with the microboss production scheduling and control system. Proceedings of the IJCAI-93 Workshop on Knowledge-Based Production Planning, Scheduling and Control.

Simon French, B. A. e M. A. D.Phil (1982). SEQUENCING AND SCHEDULING: An Introduction ot he Mathematics of the Job-Shop, Ellis Horwood Limited.

Smith, S. S. (1994). Reactive scheduling systems. Intelligent Scheduling Systems. D. E. Brown e W. T. Scherer, Kluwer.

Szelke, E. e R. M. Kerr (1994). "Knowledge-based reactive scheduling." Production Planning and Control **5 (2)**: 124 - 145.

Talbot, F. B. (1982). "Resource-constrained project scheduling with time-resource tradeoffs: the nonpreemptive case." Management Science **28**(10): 1197-1210.

Tukel, O. I., Rom, W. O. e Eksioglu, S. D. (2006). "An investigation of buffer sizing techniques in critical chain scheduling." European Journal of Operational Research 172: 401–416.

Valls, V., F. Ballestín, et al. (2008). "A hybrid genetic algorithm for the resource-constrained project scheduling problem." European Journal of Operational Research **185**.

Van de Vonder, S., E. Demeulemeester, et al. (2008). "Proactive heuristic procedures for robust project scheduling: An experimental analysis." European Journal of Operational Research **189**.

Vasconcellos, R. V. J. C. (2007). Um algoritmo genético para o problema de scheduling de projetos com restrição de recursos - Uma solução com gerenciamento de risco. Programa de Engenharia de Produção. Rio de Janeiro, Universidade Federal do Rio de Janeiro. **D. Sc.:** 251.

Vieira, G. E., J. W. Herrmann, et al. (2003). "Rescheduling manufacturing systems: A framework of strategies, policies and methods." Journal of Scheduling **6**: 39-62.

Wang, J. (2004). "A fuzzy robust scheduling approach for product development projects." European Journal of Operational Research **152**: 180-194.

Zhang, H., H. Li, et al. (2006). "Particle swarm optimization for resource-constrained project scheduling." International Journal of Project Management **24**.

Zhang, H., X. Li, et al. (2005). "Particle swarm optimization-based schemes for resource-constrained project scheduling." Automation in Construction **14**.

Zimmermann, H. J. (1991). Fuzzy Set Theory and its Applications, Kluwer Academic.

APÊNDICE A - Programação de Projetos *Fuzzy*

A Programação de Projetos *Fuzzy* (*Fuzzy Project Scheduling* - *FPS*) é apresentada por Hapke, Slowinski et al. (1994), com a idéia geral de tratar as incertezas utilizando números fuzzy para as variáveis de tempo, apresentando a possibilidade de vários tipos de recursos e dois critérios possíveis para a avaliação da programação: o tempo total do projeto (C_{max}) e a máxima latência (L_{max}).

1. Formulação matemática

Os recursos são renováveis e podem ser de *K* tipos, representados no conjunto *R*, por $R_1, ..., R_K$ com o limite de uso de $N_1, ..., N_K$ unidades em cada unidade de tempo.

O conjunto *Z* é composto de *n* atividades, com necessidades discretas de recursos, representadas pelo vetor $\boldsymbol{r} = [r_1,...,r_k]$. Para cada atividade Z_j, são conhecidos os seguintes parâmetros de tempo: a duração $\tilde{p}$ e a data devida (em que precisa estar pronta) $\tilde{d}$. Os parâmetros citados são modelados como números *fuzzy* (utilizando ~).

Como as variáveis de tempo são números *fuzzy*, o tempo total do projeto também é representado assim ($\tilde{C}_{\max}$).

A formulação matemática do problema é baseada na seguinte definição das variáveis binárias de decisão:

$$x_{jt} = \begin{cases} 1 & \text{se a atividade } Z_j \text{ está terminada no instante } t \\ 0 & \text{caso contrário} \end{cases}$$

Os parâmetros $\tilde{e}_j$, instante mais cedo em que a atividade pode ser concluída e $\tilde{l}_j$, instante mais tarde em que a atividade pode ser concluída, são calculadas utilizando o Método de Caminho Crítico (CPM), utilizando a modelagem *fuzzy* devido às incertezas.

O conjunto B_j é composto das atividades predecessoras da atividade Z_j.

Então, o problema pode ser modelado da seguinte forma:

$$\min \tilde{C}_{\max} = \sum_{t=\tilde{e}_n}^{\tilde{l}_n} t x_{nt}$$

ou

$$\min \tilde{L}_{\max} = \max_j \left\{ 0, \sum_{t=\tilde{e}_j}^{\tilde{l}_j} \left(t x_{jt} - \tilde{d}_j \right) \right\}$$

Sujeito a:

- Restrições de precedência

$$\sum_{t=\widetilde{e}_j}^{\widetilde{l}_j}\left(t-\widetilde{p}_j\right)x_{jt}-\sum_{t=\widetilde{e}_f}^{\widetilde{l}_f}tx_{ft}\geq 0,\ \ j=1,...,n+1,\forall f\in B_j$$

- Restrições de recursos

$$\sum_{j=1}^{n}\sum_{q=t}^{t+\widetilde{p}_j-1}r_{jk}x_{jq}\leq N_{kt},\ \ k=1,...,K,t=1,...,T$$

A solução do problema pode ser feita pelos métodos conhecidos de programação *fuzzy* binária. Para isso, recomenda-se a leitura de Czyzak e Slowinski (1991). Devido à grande escala dos problemas reais, a aplicação dos métodos exatos de solução fica inviável. Hapke, Slowinski et al. (1994) propuseram, então, a abordagem heurística.

2. Tratamento das incertezas

Como já apresentado na seção 3.3, os números *fuzzy* podem ser representados por seis pontos, com três cortes, baseados na função de pertinência. Os três passos propostos para tratar as incertezas são:

iv. modelar as incertezas;
v. transformar o problema com incertezas em um problema determinístico associado;
vi. solucionar o problema determinístico associado, verificar os resultados e o possível retorno a (ii) para revisão de parâmetros.

2.1 Modelagem das incertezas

Um método adequado de se modelar as incertezas é, segundo Rommelfanger (1990), definir níveis de corte da função de pertinência. Para isso, ele propôs a seguinte divisão:

- α = 1: $\mu(x)=1$ significa que o valor de x certamente pertence ao conjunto de valores possíveis;
- α = λ: $\mu(x)>\lambda$ significa que os valores de x com $\mu(x)\geq\lambda$ apresentam boas chances de pertencer ao conjunto de valores possíveis;
- α = ε: $\mu(x)<\varepsilon$ significa que valores estes valores de x têm poucas chances de pertencer ao conjunto de valores possíveis.

2.2 Transformação em problema determinístico

Como cada número *fuzzy* é composto de três α-cortes, e cada α-corte determina dois valores de durações de atividades, pode-se concluir que cada número *fuzzy* representa seis possíveis valores de duração.

Com estes seis valores de duração para cada atividade, o problema de programação de projetos com incerteza está associado a seis problemas determinísticos.

2.3 Solução do problema determinístico

Na fase de solução do problema, existem, então, seis programações diferentes, que devem ser ordenadas para a apresentação de um único resultado de programação *fuzzy*.

No artigo original, é exibida uma primeira solução viável, que não leva em conta as restrições de recursos. Assim, o gerente de projeto pode perceber possíveis casos onde são usados muitos recursos por um curto período. Nestes casos, o gerente poderia reduzir a utilização dos recursos, implicando em *makespan* maior. Porém, quando o gerente restringe os recursos, é apresentada uma programação final *fuzzy*, relativo àquela oferta de recursos. O gerente poderia, então, fazer algumas configurações diferentes de recursos, obtendo com isso diferentes programações *fuzzy*. De posse dessas programações diferentes, ele escolhe qual é a configuração de recursos que lhe parecer mais favorável.

Nesta comparação entre as diferentes configurações de recursos, é necessária a utilização de técnicas para comparação de números *fuzzy*.

3. Comparação entre números *fuzzy*

Uma explicação básica sobre a comparação entre números fuzzy pode ser feita de forma intuitiva, considerando as áreas determinadas pelas funções de pertinência. (Baldwin e Guild (1979))

Sejam *a* e *b* números *fuzzy*, com S_L e S_R sendo as áreas determinadas pelas respectivas funções de pertinência (de acordo com a Figura 49).

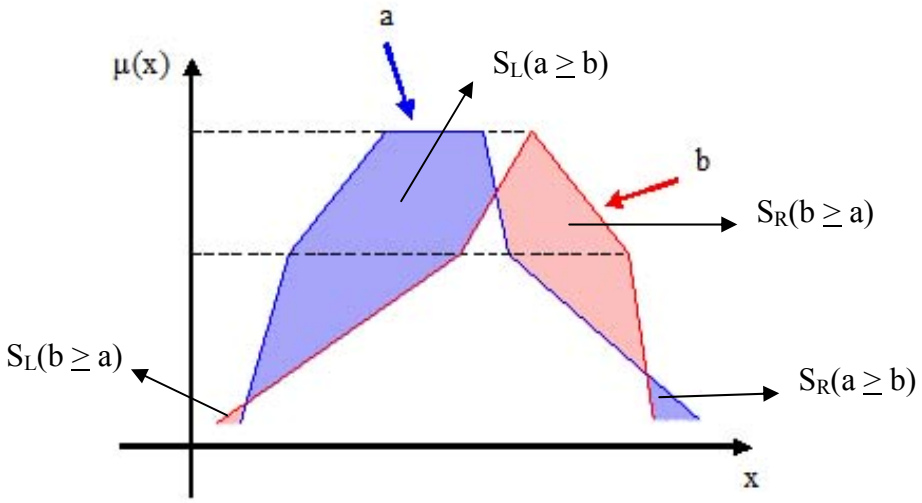


Figura 49 - Comparação entre números fuzzy

As áreas são definidas por:

$$S_L(a \geq b) = \int_{U_1(a,b)} [\inf_{x \in \Re} I(a,\alpha) - \inf_{x \in \Re} I(b,\alpha)] d\alpha ,$$

onde $U_1(a,b)$ é um subconjunto de $[\varepsilon,1]$: $\left\{\alpha \middle| \inf_{x \in \Re} I(a,\alpha) \geq \inf_{x \in \Re} I(b,\alpha)\right\}$ e

$$S_R(a \geq b) = \int_{V_1(a,b)} \left[\sup_{x \in \Re} I(a,\alpha) - \sup_{x \in \Re} I(b,\alpha)\right] d\alpha ,$$

onde $V_1(a,b)$ é um subconjunto de $[\varepsilon,1]$: $\left\{\alpha \middle| \sup_{x \in \Re} I(a,\alpha) \geq \sup_{x \in \Re} I(b,\alpha)\right\}$.

As áreas $S_L(a \leq b)$ e $S_R(a \leq b)$ podem ser definidas analogamente.

Roubens (1990) formalizou o grau de $a \geq b$, de acordo com:

$$C(a \geq b) = \max\{S_L(a \geq b) + S_R(a \geq b) - S_L(b \geq a) - S_R(b \geq a), 0\}.$$

APÊNDICE B – Instâncias utilizadas nos testes computacionais

As instâncias utilizadas têm tamanhos diferentes – 30, 60, 90 e 120 atividades. A seguir, são apresentadas as relações de precedência, as durações das atividades e os recursos necessários para a realização de cada atividade. As instâncias de projetos com 30 atividades já foram apresentados no corpo da dissertação. A seguir, são apresentadas as instâncias restantes.

1. Projetos com 60 atividades

1.1 Instância 60_1

Atividade (i)	Sucessores			Recursos (i)	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	0	0	0	0
2	11	14	16	34	13	16	47	93
3	5			32	3	39	77	80
4	7	9	12	66	3	12	30	52
5	6	22	30	70	2	4	9	23
6	8	33	38	28	10	18	29	62
7	10	15	45	42	6	12	19	54
8	13	48		40	14	20	34	36
9	29	46		36	4	8	22	86
10	17	20		8	6	21	30	33
11	34	44		32	27	30	33	58
12	32	43	54	24	2	13	22	42
13	21	31		60	2	13	17	27
14	60			32	5	11	18	52
15	24			34	4	13	22	54
16	18	37	39	44	2	4	11	72
17	27	28	42	50	3	5	22	40
18	19			56	9	19	51	54
19	23	49		50	3	4	4	5
20	24			44	5	25	31	86
21	23			52	2	10	31	91
22	26	58		26	2	2	4	9
23	25			36	15	17	30	49
24	40			76	8	14	24	73
25	35			60	14	22	27	66
26	41			60	25	33	44	50
27	29			74	7	12	17	37
28	36			44	5	9	11	29
29	52	61		60	6	11	17	89
30	35	50		30	3	10	22	44
31	52			38	3	3	29	69
32	60			26	11	13	31	53

33	47			28	4	28	37	74
34	35			52	3	8	10	23
35	40	55	56	60	14	31	39	66
36	41			38	3	25	54	56
37	49			22	3	5	9	47
38	40			32	2	5	25	71
39	57			44	6	11	35	68
40	53			46	2	36	37	64
41	53			16	3	7	23	68
42	56			32	2	2	13	42
43	55			58	4	8	17	89
44	57			40	12	22	42	53
45	53			30	3	18	36	47
46	47	56		66	2	8	46	60
47	58			46	7	9	14	25
48	52			38	8	14	17	23
49	59			38	7	26	80	82
50	51	61		38	2	3	14	23
51	54			66	3	5	29	49
52	54			38	3	34	36	70
53	57			34	3	7	18	34
54	55			54	5	24	41	50
55	58	60		38	5	17	20	62
56	61			36	2	6	28	57
57	59			44	2	20	24	36
58	59			64	2	8	13	26
59	62			72	15	21	21	23
60	62			36	6	6	28	38
61	62			64	3	15	45	61
62				0	0	0	0	0

1.2 Instância 60_2

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	5	6	9	125	13	16	47	93
3	10	11	12	90	3	39	77	80
4	7	13	50	140	3	12	30	52
5	17	36	47	110	3	4	9	23
6	27	34	38	95	10	18	29	62
7	8	14	52	145	6	12	19	54
8	20	24	48	185	14	20	34	36
9	15	17	36	65	4	8	22	86
10	17	18	28	125	6	21	30	33
11	23	26	30	175	27	30	33	58
12	25	33	44	140	3	13	22	42
13	18	32	37	110	3	13	17	27
14	22	41		130	5	11	18	52
15	16	22	56	65	4	13	22	54

16	23	43	51	75	3	4	11	72
17	19	21	31	70	3	5	22	40
18	19	41		105	9	19	51	54
19	29	35	39	75	3	4	4	5
20	41	54	56	105	5	25	31	86
21	22	27	51	115	3	10	31	91
22	39	46		145	3	3	4	9
23	28	31		135	15	17	30	49
24	35	43	61	160	8	14	24	73
25	29	36	37	125	14	22	27	66
26	27	32	47	60	25	33	44	50
27	33	43		130	7	12	17	37
28	45	58		165	5	9	11	29
29	34			115	6	11	17	89
30	44	47		100	3	10	22	44
31	44	45		165	3	3	29	69
32	34	35	40	150	11	13	31	53
33	50	53	59	110	4	28	37	74
34	52	55		145	3	8	10	23
35	49	53		150	14	31	39	66
36	39	40		40	3	25	54	56
37	38	51		75	3	5	9	47
38	40	48		145	3	5	25	71
39	42	49		75	6	11	35	68
40	42	45	52	110	3	36	37	64
41	42	46	60	105	3	7	23	68
42	59			120	3	3	13	42
43	60			95	4	8	17	89
44	50	57		120	12	22	42	53
45	46			120	3	18	36	47
46	57			125	3	8	46	60
47	48	57		160	7	9	14	25
48	49			145	8	14	17	23
49	55			80	7	26	80	82
50	55			70	3	3	14	23
51	54	61		120	3	5	29	49
52	53	54		70	3	34	36	70
53	56			140	3	7	18	34
54	59			110	5	24	41	50
55	58			105	5	17	20	62
56	58			140	3	6	28	57
57	61			65	3	20	24	36
58	60			115	3	8	13	26
59	62			165	15	21	21	23
60	62			125	6	6	28	38
61	62			20	3	15	45	61
62				0	0	0	0	0

1.3 Instância 60_3

Atividade (i)	Sucessores(i)			Recursos (i)	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	0	0	0	0
2	5	10	11	100	13	16	47	93
3	35	42	43	170	3	39	77	80
4	7	9	36	60	3	12	30	52
5	6	23	53	110	3	4	9	23
6	13	17	26	120	10	18	29	62
7	8	14	32	170	6	12	19	54
8	12	21	22	155	14	20	34	36
9	17	19	22	85	4	8	22	86
10	15	20	39	185	6	21	30	33
11	16	39	51	165	27	30	33	58
12	49			120	3	13	22	42
13	18	28	39	145	3	13	17	27
14	21	25	55	160	5	11	18	52
15	27	34	38	145	4	13	22	54
16	30	41	54	105	3	4	11	72
17	18	21	35	160	3	5	22	40
18	20	32	33	115	9	19	51	54
19	24	25	28	110	3	4	4	5
20	49			100	5	25	31	86
21	29	40		120	3	10	31	91
22	31	43	44	145	3	3	4	9
23	26	30	37	135	15	17	30	49
24	29	40	48	165	8	14	24	73
25	27	30	31	130	14	22	27	66
26	32	58		155	25	33	44	50
27	35	43	44	120	7	12	17	37
28	29	33	50	145	5	9	11	29
29	31			110	6	11	17	89
30	33	34	57	115	3	10	22	44
31	34	45		155	3	3	29	69
32	49			150	11	13	31	53
33	61			135	4	28	37	74
34	52			145	3	8	10	23
35	37	41		160	14	31	39	66
36	37	38	57	170	3	25	54	56
37	52			120	3	5	9	47
38	45	53	55	100	3	5	25	71
39	40			120	6	11	35	68
40	41	54		100	3	36	37	64
41	50	60		180	3	7	23	68
42	47	48	55	155	3	3	13	42
43	47	50		145	4	8	17	89
44	45	46	56	120	12	22	42	53
45	54			170	3	18	36	47
46	47	48		130	3	8	46	60

47	51			95	7	9	14	25
48	51	59		115	8	14	17	23
49	60			85	7	26	80	82
50	52	56		115	3	3	14	23
51	53	57		125	3	5	29	49
52	59			185	3	34	36	70
53	60			170	3	7	18	34
54	61			135	5	24	41	50
55	56	59		105	5	17	20	62
56	58			110	3	6	28	57
57	58			85	3	20	24	36
58	61			120	3	8	13	26
59	62			190	15	21	21	23
60	62			125	6	6	28	38
61	62			150	3	15	45	61
62				0	0	0	0	0

2. Projetos com 90 atividades

2.1 Instância 90_1

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	10	13	26	100	3	5	15	64
3	14			80	16	37	71	75
4	5	6	8	90	5	21	27	33
5	7	21	75	80	3	10	18	63
6	17	20		90	29	29	54	73
7	9	19	22	185	3	35	43	52
8	24	25	66	105	23	35	71	83
9	16			130	6	15	28	77
10	11	12	42	90	18	37	41	86
11	23	36	77	135	14	20	34	82
12	28	37	52	175	10	26	46	70
13	15	21		145	6	20	26	34
14	44			105	19	33	35	51
15	40	43	51	95	3	10	20	65
16	30	39		100	9	14	16	32
17	18	65		120	3	34	35	70
18	35	45	81	105	3	3	3	11
19	29	33	47	35	18	55	56	57
20	67			85	3	7	22	34
21	54	86		130	5	51	54	84
22	27			80	10	32	42	45
23	53			95	9	26	29	44
24	31	80		140	17	30	31	62
25	30	37		115	5	33	56	66
26	71	90		80	3	3	7	28
27	32			100	4	9	13	35

28	63			155	3	3	46	52
29	34	62		135	3	6	9	9
30	89			120	3	30	51	79
31	41			85	7	17	27	35
32	84			105	26	30	38	63
33	57			115	6	13	14	36
34	70			65	4	4	15	35
35	49	69		135	6	8	20	36
36	38	58		100	3	3	3	4
37	62	68	85	125	3	19	22	55
38	87			95	3	4	15	43
39	59	73		45	3	13	46	75
40	46	50		130	18	20	34	36
41	58			160	4	21	39	83
42	91			120	7	7	23	39
43	88			130	3	4	7	90
44	48	77		120	3	7	23	60
45	85			75	13	35	39	45
46	47			155	3	4	7	49
47	60			60	3	19	27	30
48	70			120	3	3	37	83
49	56			135	7	28	41	48
50	78			50	6	7	18	19
51	55			135	5	6	16	38
52	80			80	3	6	19	26
53	60			130	4	15	44	83
54	64			145	3	21	23	38
55	83			165	17	22	48	64
56	77			130	3	4	40	41
57	72			120	9	15	44	68
58	65			145	3	23	26	52
59	79			140	3	7	50	59
60	61			110	17	17	18	57
61	76			155	7	8	10	10
62	69	70		115	3	31	38	51
63	64			175	8	20	22	63
64	81			125	9	15	25	76
65	76	83		85	20	30	35	37
66	86			120	4	5	19	33
67	73	86		165	10	43	52	83
68	78			125	10	28	40	45
69	79			130	3	12	52	76
70	72			95	3	24	32	50
71	78			150	5	13	13	25
72	87			80	4	18	20	30
73	74	76		65	24	35	39	70
74	81			70	21	28	39	65
75	82	84	85	115	3	5	18	53
76	79	84		120	3	3	3	6

77	80			95	3	3	20	28
78	87			40	6	12	21	84
79	82			65	13	16	27	30
80	82			90	8	39	40	42
81	91			185	3	4	27	99
82	88			140	5	8	43	49
83	91			125	4	5	6	68
84	90			100	15	15	55	63
85	89			105	6	13	47	48
86	88			95	14	29	57	79
87	89			100	3	4	51	53
88	90			165	9	9	12	17
89	92			135	4	4	19	24
90	92			105	10	18	52	69
91	92			140	16	17	18	69
92				0	0	0	0	0

2.2 Instância 90_2

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	9	11	12	145	4	31	36	62
3	8	33	67	175	3	3	40	53
4	5	15	21	115	19	29	49	92
5	6	7	13	110	3	7	29	43
6	10	16	25	145	6	16	27	40
7	8	19	24	130	4	11	25	30
8	55	74		130	3	6	15	49
9	10	18	27	75	31	36	36	62
10	14	31		115	9	23	37	76
11	16	23		125	10	14	46	86
12	38	52		75	5	9	26	63
13	49	57	65	90	3	3	10	34
14	28	43		105	48	49	66	88
15	41	57	89	95	3	19	26	61
16	17	22	27	125	17	44	46	49
17	20	47	54	130	6	46	49	89
18	28	48		40	3	18	43	55
19	23	53	68	175	3	3	10	35
20	29	59	77	85	3	3	3	16
21	34	40	55	70	7	25	35	76
22	31			75	6	9	9	37
23	41	46	70	175	3	4	11	68
24	26	35	36	105	6	41	81	85
25	26	30	34	75	3	8	11	30
26	28	32	88	105	3	16	31	49
27	50	61	79	135	3	11	23	54
28	57	63	91	115	10	13	36	71

29	38	45	49	115	4	7	7	20
30	33			105	11	11	32	56
31	34	42	66	100	3	3	8	18
32	72	77	82	110	3	3	6	18
33	36	37	39	145	3	3	34	39
34	39	48	71	115	6	7	10	49
35	44	52	54	200	3	4	11	32
36	44	59	85	130	3	23	39	52
37	38	49	51	45	3	27	27	46
38	41	43	80	125	3	5	72	76
39	46	86		180	8	9	10	22
40	47	48	83	115	8	18	45	73
41	75	78		135	3	14	30	42
42	60	72	86	115	8	15	17	22
43	44	46	61	60	5	6	8	9
44	75	87		115	9	18	43	53
45	53	72	80	105	6	11	29	40
46	64			100	3	28	29	31
47	56			170	3	3	3	55
48	62	65	77	140	3	7	28	40
49	71	83		110	3	4	6	11
50	51	55	69	95	3	11	64	65
51	76			140	11	11	18	18
52	53	58	70	135	6	7	26	40
53	66	73	84	110	13	21	32	45
54	63	64	66	35	14	16	21	42
55	73	85		60	3	10	19	63
56	62	70	74	115	9	19	44	57
57	67			45	3	11	15	33
58	59	61		65	3	12	47	53
59	68	76		160	7	17	18	61
60	62	73		65	3	5	16	62
61	64	68	69	130	5	12	18	47
62	63			110	27	27	33	49
63	67	81		125	6	33	43	68
64	65	81		115	6	23	23	40
65	85			105	3	14	32	43
66	69			170	6	9	17	25
67	76	79		115	3	3	26	26
68	74	88		185	24	32	62	74
69	75	78		95	3	7	11	80
70	71	78		150	3	7	8	55
71	79			105	3	4	48	57
72	83			155	7	34	41	65
73	88			120	8	23	37	77
74	86			55	5	26	33	45
75	81			80	20	31	35	72
76	80	82		70	3	3	23	55
77	87			100	3	7	52	72

78	87			105	15	15	33	73
79	84			100	3	3	5	18
80	90			85	6	8	44	47
81	82			230	3	7	23	82
82	90			105	6	44	45	86
83	84			105	10	30	33	55
84	90			95	19	23	28	55
85	89			135	32	33	35	62
86	89			60	4	7	24	45
87	91			60	3	12	40	62
88	91			55	9	12	21	30
89	92			130	5	25	32	44
90	92			160	3	13	23	40
91	92			100	6	7	9	16
92				0	0	0	0	0

2.3 Instância 90_3

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	8	60	64	90	3	5	15	64
3	5	20	23	50	16	37	71	75
4	13	15	39	70	5	21	27	33
5	6	7	26	130	3	10	18	63
6	10	11	34	150	29	29	54	73
7	12			80	3	35	43	52
8	9	19	29	160	23	35	71	83
9	16	53	78	140	6	15	28	77
10	17	25	36	160	18	37	41	86
11	18	21	24	120	14	20	34	82
12	14	49	70	90	10	26	46	70
13	30			90	6	20	26	34
14	89			110	19	33	35	51
15	40			80	3	10	20	65
16	37			90	9	14	16	32
17	32			140	3	34	35	70
18	22	58	63	80	3	3	3	11
19	27	45	47	120	18	55	56	57
20	33	34	76	110	3	7	22	34
21	47	52		80	5	51	54	84
22	55	66		150	10	32	42	45
23	85			110	9	26	29	44
24	31	61		80	17	30	31	62
25	51			190	5	33	56	66
26	28	32		160	3	3	7	28
27	43	59		60	4	9	13	35
28	35	50		70	3	3	46	52
29	74	75		110	3	6	9	9

30	41			120	3	30	51	79
31	37			80	7	17	27	35
32	48			130	26	30	38	63
33	72			110	6	13	14	36
34	74			50	4	4	15	35
35	43			100	6	8	20	36
36	38	42	54	150	3	3	3	4
37	63			60	3	19	22	55
38	73			90	3	4	15	43
39	44	46	68	100	3	13	46	75
40	58			20	18	20	34	36
41	47			120	4	21	39	83
42	67			90	7	7	23	39
43	70			110	3	4	7	90
44	73			120	3	7	23	60
45	62	72		60	13	35	39	45
46	54	64		70	3	4	7	49
47	56			120	3	19	27	30
48	56			110	3	3	37	83
49	59			90	7	28	41	48
50	65	86		150	6	7	18	19
51	57			160	5	6	16	38
52	57	59	80	130	3	6	19	26
53	73			140	4	15	44	83
54	69			80	3	21	23	38
55	89			70	17	22	48	64
56	76			70	3	4	40	41
57	91			120	9	15	44	68
58	88			140	3	23	26	52
59	84			190	3	7	50	59
60	82			180	17	17	18	57
61	64			170	7	8	10	10
62	65	77		50	3	31	38	51
63	71			140	8	20	22	63
64	84			90	9	15	25	76
65	87			90	20	30	35	37
66	81			70	4	5	19	33
67	81			40	10	43	52	83
68	79			100	10	28	40	45
69	85			130	3	12	52	76
70	85			110	3	24	32	50
71	90			190	5	13	13	25
72	78			110	4	18	20	30
73	74			30	24	35	39	70
74	87			130	21	28	39	65
75	77	79		70	3	5	18	53
76	79	81		100	3	3	3	6
77	90			140	3	3	20	28
78	88			180	6	12	21	84

79	83			40	13	16	27	30
80	91			130	8	39	40	42
81	83			160	3	4	27	99
82	86			160	5	8	43	49
83	91			110	4	5	6	68
84	86			140	15	15	55	63
85	88			110	6	13	47	48
86	87			70	14	29	57	79
87	90			110	3	4	51	53
88	89			150	9	9	12	17
89	92			110	4	4	19	24
90	92			130	10	18	52	69
91	92			120	16	17	18	69
92				0	0	0	0	0

3. Projetos com 120 atividades

3.1 Instância 120_1

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	7	15	30	32	19	29	49	92
3	9	10	37	28	3	7	29	43
4	5	27	32	16	6	16	27	40
5	6	8	11	37	4	11	25	30
6	16	79		17	2	6	15	49
7	12	19	25	23	31	36	36	62
8	17	60	66	29	9	23	37	76
9	11	18	80	16	10	14	46	86
10	23	29		27	5	9	26	63
11	13	54	91	25	2	2	10	34
12	14	53	78	33	48	49	66	88
13	93			36	2	19	26	61
14	67			35	17	44	46	49
15	20	26		34	6	46	49	89
16	50	92	110	38	2	18	43	55
17	117			35	2	2	10	35
18	114			24	3	3	3	16
19	21			24	7	25	35	76
20	22	38	48	44	6	9	9	37
21	31	49	103	41	2	4	11	68
22	36	58		36	6	41	81	85
23	24	35	101	32	2	8	11	30
24	56			36	3	16	31	49

25	65			35	2	11	23	54
26	43	63	105	37	10	13	36	71
27	28	34	46	33	4	7	7	20
28	47			36	11	11	32	56
29	70			29	2	3	8	18
30	33	39	42	28	2	3	6	18
31	69			33	2	3	34	39
32	52	106	116	32	6	7	10	49
33	40			31	2	4	11	32
34	59	121		27	3	23	39	52
35	97			27	2	27	27	46
36	55			26	2	5	72	76
37	47			30	8	9	10	22
38	112			33	8	18	45	73
39	57	103		29	2	14	30	42
40	41	74		20	8	15	17	22
41	44	55		29	5	6	8	9
42	81	84		34	9	18	43	53
43	45			35	6	11	29	40
44	51	86		19	2	28	29	31
45	72	107		30	2	2	2	55
46	58			27	2	7	28	40
47	57	85		36	2	4	6	11
48	65			36	2	11	64	65
49	111			29	11	11	18	18
50	85			29	6	7	26	40
51	64	76		31	13	21	32	45
52	68	73		29	14	16	21	42
53	56			35	2	10	19	63
54	67			26	9	19	44	57
55	77			26	3	11	15	33
56	102			27	3	12	47	53
57	62	63	98	32	7	17	18	61
58	70	82		25	2	5	16	62
59	61			22	5	12	18	47
60	110			31	27	27	33	49
61	90			29	6	33	43	68
62	97			28	6	23	23	40
63	77	94		27	2	14	32	43
64	71			39	6	9	17	25
65	80			22	2	2	26	26
66	104			28	24	32	62	74
67	98			36	2	7	11	80
68	81			32	2	7	8	55
69	114			26	2	4	48	57
70	102			22	7	34	41	65
71	100			30	8	23	37	77
72	75			28	5	26	33	45
73	118			28	20	31	35	72

74	111			30	2	3	23	55
75	79	89		32	2	7	52	72
76	106			36	15	15	33	73
77	83	88	115	24	2	3	5	18
78	109			25	6	8	44	47
79	84			35	3	7	23	82
80	87	95		37	6	44	45	86
81	93	101		32	10	30	33	55
82	109			31	19	23	28	55
83	108			17	32	33	35	62
84	100			32	4	7	24	45
85	112			27	2	12	40	62
86	99			31	9	12	21	30
87	96	104	113	30	5	25	32	44
88	89			46	2	13	23	40
89	120			26	6	7	9	16
90	113			24	5	6	9	38
91	102			23	2	6	38	59
92	103			31	2	5	15	64
93	95	105		34	16	37	71	75
94	100			30	5	21	27	33
95	111			42	2	10	18	63
96	108			38	29	29	54	73
97	99			30	2	35	43	52
98	116			21	23	35	71	83
99	116			36	6	15	28	77
100	109			26	18	37	41	86
101	110			35	14	20	34	82
102	114			15	10	26	46	70
103	115			35	6	20	26	34
104	105			21	19	33	35	51
105	115			36	2	10	20	65
106	119			19	9	14	16	32
107	121			40	2	34	35	70
108	121			34	2	2	2	11
109	112			32	18	55	56	57
110	113			27	3	7	22	34
111	119			34	5	51	54	84
112	118			29	10	32	42	45
113	120			22	9	26	29	44
114	117			20	17	30	31	62
115	117			30	5	33	56	66
116	119			28	2	2	7	28
117	118			19	4	9	13	35
118	120			31	2	2	46	52
119	122			33	2	6	9	9
120	122			32	3	30	51	79
121	122			26	7	17	27	35
122				0	0	0	0	0

3.2 Instância 120_2

Atividade (i)	Sucessores(i)			Recursos (i)	a(i)	b(i)	c(i)	d(i)
1	2	3	4	0	0	0	0	0
2	9			100	13	16	47	93
3	5	7	8	110	3	39	77	80
4	93			120	3	12	30	52
5	6	10	13	110	3	4	9	23
6	17	28		130	10	18	29	62
7	101			70	6	12	19	54
8	12	19	48	40	14	20	34	36
9	11	23		160	4	8	22	86
10	21	29	40	140	6	21	30	33
11	15	30	32	130	27	30	33	58
12	14			130	3	13	22	42
13	43	66		60	3	13	17	27
14	16	18	35	90	5	11	18	52
15	20	33	50	110	4	13	22	54
16	53	107		120	3	4	11	72
17	22	64		130	3	5	22	40
18	28			160	9	19	51	54
19	106			200	3	4	4	5
20	27	41	61	90	5	25	31	86
21	56	85		110	3	10	31	91
22	53	54		60	3	3	4	9
23	24	38	57	90	15	17	30	49
24	25	26	44	160	8	14	24	73
25	100	116		140	14	22	27	66
26	34	36	67	150	25	33	44	50
27	55			140	7	12	17	37
28	31	80		140	5	9	11	29
29	49			70	6	11	17	89
30	39	43	79	170	3	10	22	44
31	37			130	3	3	29	69
32	46			170	11	13	31	53
33	63	106		130	4	28	37	74
34	72			80	3	8	10	23
35	42			140	14	31	39	66
36	53			70	3	25	54	56
37	56			60	3	5	9	47
38	62	69	87	190	3	5	25	71
39	52			180	6	11	35	68
40	66	77	107	110	3	36	37	64
41	109	110		60	3	7	23	68
42	70			110	3	3	13	42
43	57			40	4	8	17	89

44	45	59	71	110	12	22	42	53
45	46			50	3	18	36	47
46	47	51	92	60	3	8	46	60
47	75	81	82	110	7	9	14	25
48	58			80	8	14	17	23
49	68			100	7	26	80	82
50	100			120	3	3	14	23
51	98			70	3	5	29	49
52	115			200	3	34	36	70
53	65	102	112	50	3	7	18	34
54	106			160	5	24	41	50
55	60			80	5	17	20	62
56	72			90	3	6	28	57
57	84	107		80	3	20	24	36
58	77	88		100	3	8	13	26
59	83	96		80	15	21	21	23
60	84			40	6	6	28	38
61	110			30	3	15	45	61
62	76			30	28	28	28	31
63	66	103		110	16	24	33	71
64	74			120	4	31	36	62
65	74	85		80	3	3	40	53
66	91	119		120	19	29	49	92
67	114			190	3	7	29	43
68	78			40	6	16	27	40
69	89			150	4	11	25	30
70	73			50	3	6	15	49
71	73			160	31	36	36	62
72	90			100	9	23	37	76
73	93	94		110	10	14	46	86
74	95			160	5	9	26	63
75	118			100	3	3	10	34
76	99			90	48	49	66	88
77	97			110	3	19	26	61
78	98			50	17	44	46	49
79	95			60	6	46	49	89
80	88	90		130	3	18	43	55
81	99	105		130	3	3	10	35
82	89			130	3	3	3	16
83	86			70	7	25	35	76
84	111			120	6	9	9	37
85	101			110	3	4	11	68
86	94			150	6	41	81	85
87	91			100	3	8	11	30
88	95			60	3	16	31	49
89	117			130	3	11	23	54
90	109			190	10	13	36	71
91	120			80	4	7	7	20
92	116			140	11	11	32	56

93	99			110	3	3	8	18
94	97			110	3	3	6	18
95	105			150	3	3	34	39
96	104			80	6	7	10	49
97	100	108		110	3	4	11	32
98	108			100	3	23	39	52
99	101			60	3	27	27	46
100	121			170	3	5	72	76
101	108			140	8	9	10	22
102	114			120	8	18	45	73
103	105			100	3	14	30	42
104	112			100	8	15	17	22
105	113			100	5	6	8	9
106	109			100	9	18	43	53
107	113			60	6	11	29	40
108	115	116		60	3	28	29	31
109	121			140	3	3	3	55
110	114			110	3	7	28	40
111	118			130	3	4	6	11
112	117	118		100	3	11	64	65
113	117			120	11	11	18	18
114	115			90	6	7	26	40
115	121			140	13	21	32	45
116	119			150	14	16	21	42
117	119	120		140	3	10	19	63
118	120			60	9	19	44	57
119	122			70	3	11	15	33
120	122			170	3	12	47	53
121	122			130	7	17	18	61
122				0	0	0	0	0

3.3 Instância 120_3

Atividade (i)	**Sucessores(i)**			**Recursos (i)**	**a(i)**	**b(i)**	**c(i)**	**d(i)**
1	2	3	4	0	0	0	0	0
2	6	7	28	140	13	16	47	93
3	5	10	22	150	3	39	77	80
4	9	12	20	150	3	12	30	52
5	8	19	21	150	3	4	9	23
6	47	49	111	100	10	18	29	62
7	26	52	65	160	6	12	19	54
8	11	44	118	130	14	20	34	36
9	31	36	38	130	4	8	22	86
10	17	25	71	170	6	21	30	33
11	15	33	35	180	27	30	33	58
12	13	14	18	80	3	13	22	42
13	47			150	3	13	17	27

14	16	30	48	170	5	11	18	52
15	25	45	63	130	4	13	22	54
16	21	24		110	3	4	11	72
17	55	73	75	140	3	5	22	40
18	27	85	108	130	9	19	51	54
19	40	41	62	110	3	4	4	5
20	53	83		120	5	25	31	86
21	71	74		140	3	10	31	91
22	23	32	43	200	3	3	4	9
23	29	46	56	140	15	17	30	49
24	37			210	8	14	24	73
25	42	54		160	14	22	27	66
26	53	70	88	170	25	33	44	50
27	38	67	98	190	7	12	17	37
28	34	72	73	130	5	9	11	29
29	57	75	77	180	6	11	17	89
30	44	96		190	3	10	22	44
31	37	57	115	110	3	3	29	69
32	77			210	11	13	31	53
33	39	62	68	120	4	28	37	74
34	51	81	84	140	3	8	10	23
35	42	80	87	170	14	31	39	66
36	54	94	111	50	3	25	54	56
37	58	61	64	160	3	5	9	47
38	43	48	81	120	3	5	25	71
39	67	74		90	6	11	35	68
40	52	95	108	200	3	36	37	64
41	79	114		210	3	7	23	68
42	60	93	106	60	3	3	13	42
43	44	91		130	4	8	17	89
44	112			200	12	22	42	53
45	47	51	54	200	3	18	36	47
46	50	51	64	60	3	8	46	60
47	67	95		210	7	9	14	25
48	59	73		210	8	14	17	23
49	64	70		200	7	26	80	82
50	78	86		120	3	3	14	23
51	62	85	110	140	3	5	29	49
52	66	87	101	210	3	34	36	70
53	60	61	93	150	3	7	18	34
54	59	69	81	160	5	24	41	50
55	59	89		200	5	17	20	62
56	77	82	94	130	3	6	28	57
57	63	68	84	210	3	20	24	36
58	75			210	3	8	13	26
59	99			200	15	21	21	23
60	79			200	6	6	28	38
61	68	94	102	120	3	15	45	61
62	76	97		210	28	28	28	31

63	69	102	120	160	16	24	33	71
64	66	72	74	200	4	31	36	62
65	83	96	108	130	3	3	40	53
66	71	76	84	90	19	29	49	92
67	79	89		200	3	7	29	43
68	80	110		60	6	16	27	40
69	80			210	4	11	25	30
70	90	96		180	3	6	15	49
71	82	109		200	31	36	36	62
72	98			200	9	23	37	76
73	76	93	102	210	10	14	46	86
74	82	92	100	150	5	9	26	63
75	90	121		170	3	3	10	34
76	100			50	48	49	66	88
77	78	83	105	100	3	19	26	61
78	97	104	111	180	17	44	46	49
79	99			160	6	46	49	89
80	85	107		210	3	18	43	55
81	87	91	92	140	3	3	10	35
82	86	105		140	3	3	3	16
83	88	91	97	90	7	25	35	76
84	86	90		80	6	9	9	37
85	116			90	3	4	11	68
86	88			140	6	41	81	85
87	89	113		140	3	8	11	30
88	103	106		90	3	16	31	49
89	106	115		210	3	11	23	54
90	92	98	104	170	10	13	36	71
91	95			180	4	7	7	20
92	114			150	11	11	32	56
93	103	109		70	3	3	8	18
94	117			150	3	3	6	18
95	101	109		70	3	3	34	39
96	99	105		140	6	7	10	49
97	100	101		200	3	4	11	32
98	118			210	3	23	39	52
99	103	107	115	70	3	27	27	46
100	121			110	3	5	72	76
101	107	120		60	8	9	10	22
102	112			200	8	18	45	73
103	104			200	3	14	30	42
104	110			190	8	15	17	22
105	112			210	5	6	8	9
106	120			200	9	18	43	53
107	113			160	6	11	29	40
108	116			90	3	28	29	31
109	114			200	3	3	3	55
110	116			180	3	7	28	40
111	118			160	3	4	6	11

112	113			190	3	11	64	65
113	121			180	11	11	18	18
114	117			150	6	7	26	40
115	117			200	13	21	32	45
116	119			140	14	16	21	42
117	119			150	3	10	19	63
118	119			150	9	19	44	57
119	122			190	3	11	15	33
120	122			140	3	12	47	53
121	122			150	7	17	18	61
122				0	0	0	0	0