Pontifícia Universidade Católica
do Rio de Janeiro

Nicole Suclla Fernández

Gestão de múltiplos projetos por meio da metodologia da cadeia crítica: Efeitos do buffer de capacidade e dos critérios para priorizar atividades

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção de grau de mestre pelo Programa de Pós-Graduação em Engenharia de Produção do Departamento de Engenharia Industrial da PUC-Rio.

Orientador: Prof. Leonardo Junqueira Lustosa

Rio de Janeiro
Maio 2008

Pontifícia Universidade Católica
do Rio de Janeiro

Nicole Suclla Fernández

Gestão de múltiplos projetos por meio da metodologia da cadeia crítica: Efeitos do buffer de capacidade e dos critérios para priorizar atividades

Dissertação apresentada como requisito parcial para obtenção de grau de mestre pelo Programa de Pós-Graduação em Engenharia de Produção do Departamento de Engenharia Industrial da PUC - Rio

Prof. Leonardo Junqueira Lustosa
Orientador
Departamento de Engenharia Industrial – PUC - Rio

Prof. Nélio Domingues Pizzolato
Departamento de Engenharia Industrial – PUC - Rio

Prof. Silvio Hamacher
Departamento de Engenharia Industrial – PUC - Rio

Prof. José Eugênio Leal
Coordenador Setorial do Centro Técnico Cientifico – PUC - Rio

Rio de Janeiro, 16 de maio de 2008

Nicole Suclla Fernández

Graduou-se em Engenharia Industrial na "Universidad Nacional de San Agustín", Arequipa – Perú em fev., 2004. Obteve o Premio de Excelência Universitária outorgado anualmente pela associação "AFP Unión Vida". Realizou cursos de extensão em Gestão de serviços, Qualidade total, Gestão de recursos humanos e Programação de PLC I de Allen Bradley. Estagiou como assistente técnico-gerencial de microempresários na ONG Cecycap e na área de controle da produção na mineradora "Sociedade Minera Cerro Verde SAA". Trabalhou na área de Logística e controle de custos durante o 2004 – 2005.

Ficha Catalográfica

Fernández, Nicole Suclla

Gestão de múltiplos projetos por meio da metodologia da cadeia crítica: efeitos do buffer de capacidade e dos critérios para priorizar atividades / Nicole Suclla Fernández; orientador: Leonardo Junqueira Lustosa. – 2008.

137 f. : il.(col.) ; 30 cm

Dissertação (Mestrado em Engenharia Industrial)–Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2008.

Inclui bibliografia

1. Engenharia Industrial – Teses. 2. Múltiplos projetos. 3. Cadeia crítica. 4. Buffer de capacidade. 5. Gestão de projetos. 6. Critérios para priorizar atividades. I. Lustosa, Leonardo Junqueira. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Industrial. V. Título.

CDD 658.5

Para meus pais, que me apoiaram incondicionalmente, confiando sempre na minha capacidade e me mostrando a luz no final do caminho.
Para VOCÊ que sempre me guias pelo caminho certo, mesmo quando eu esqueço que estas comigo.

Agradecimentos

Ao meu orientador Prof. Leonardo Lustosa, por compartilhar comigo seus conhecimentos e sua experiência, fazendo da orientação um relacionamento de amigos e colaboradores.

Aos integrantes da banca examinadora pela disponibilidade e observações importantes que realçaram este trabalho

Ao Luiz Cesar Nanci pelas observações relevantes que me ajudaram a esclarecer os objetivos finais deste trabalho

Aos meus amigos pelas dicas de pesquisa e estudo que facilitaram o desenvolvimento deste trabalho

Aos meus pais, pela transmissão de experiências e conselhos oportunos que não só me ajudaram na pesquisa, senão também para o dia-a-dia.

A Paula e ao Leandro por estar sempre comigo e "limpar minha cabeça" quando as preocupações me invadiam.

A minha família pela motivação para ser sempre melhor

A Capes pela bolsa de estudos que me permitiu iniciar o caminho da pesquisa e culminar este trabalho.

A Prochain Solutions Inc. pela disponibilidade de licenças.

Sem a participação de vocês não estaria aqui e este trabalho não existiria.

Resumo

Fernández, Nicole Suclla; Lustosa, Leonardo Junqueira; **Gestão de múltiplos projetos por meio da metodologia da cadeia crítica: Efeitos do buffer de capacidade e dos critérios para priorizar atividades**. Rio de Janeiro, 2008. 137p. Dissertação de Mestrado – Departamento de Engenharia Industrial, Pontifícia Universidade Católica do Rio de Janeiro.

Este trabalho trata da gestão de múltiplos projetos mediante a metodologia da Cadeia Crítica e Gestão de Buffers (*Critical Chain and Buffer Management* – CC/BM). Concentra-se na análise e entendimento dos efeitos que as abordagens mais usuais para dimensionamento dos *buffers* de capacidade (*Capacity Buffers* – BC) e definição dos critérios de priorização de atividades têm sobre os objetivos da empresa e dos próprios projetos. O estudo baseia-se num problema-exemplo apresentado por Pritsker B. *et al.* (1969.). Para facilitar a análise, esse exemplo se caracteriza por uma estrutura simples, mas que apresenta interdependências relevantes para o estudo dos conflitos de compartilhamento de recursos entre projetos. Analisam-se as abordagens de "todos juntos", "projetos sucessivos" e do "recurso gargalo", propostas por Newbold (1998) para planejamento conjunto de projetos simultâneos. Tais abordagens são implementadas através das ferramentas de programação de projetos *Prochain & Pipeline*. A partir dos programas (*schedules*) resultantes, analisam-se as características e deficiências de cada abordagem. Ademais, tais resultados permitem identificar certas condições e efeitos conflitantes no dimensionamento dos BC e no critério de priorização de atividades utilizado nas implementações típicas da metodologia. Tais observações criaram a necessidade de complementar a análise mediante a simulação probabilística da etapa de execução dos projetos. A pesquisa explica a diferença entre os efeitos dos critérios de priorização dinâmica e estática, esclarece a interação entre os *buffers* de capacidade e *buffers* de projeto. Ela ainda analisa os efeitos de dimensionar os BCs, para todos os projetos, como uma mesma porcentagem da carga do recurso e ilustra os *trade-offs* entre estabilidade do sistema e o *makespan* dos projetos.

Palavras-chaves

Múltiplos projetos; cadeia crítica; buffer de capacidade; gestão de projetos; critérios para priorizar atividades

Abstract

Fernández, Nicole Suclla; Lustosa, Leonardo Junqueira; **Multi-project management through the critical chain methodology: Effects of capacity buffer and of the criteria to prioritize activities**. Rio de Janeiro, 2008. 137p. Master Dissertation – Industrial Engineering Department, PUC-Rio

This research addresses the management of multiple simultaneous projects through the application of the Critical Chain and Buffers Management (CC/BM) methodology. The focus is on the analysis and interpretation of the effects that the more usual approaches for Capacity Buffers (CB) sizing and criteria to prioritize activities have on the company's and on the projects' own objectives. The study is based on the problem-example presented by Pritsker B. *et al.* (1969.) For simplifying the analysis, this example is characterized by a simple structure that, nevertheless, presents interdependences significant for the study of conflicts created by sharing resources between simultaneous projects. The approaches for simultaneous projects management, presented by Newbold (1998) are analyzed, namely: "the all together", "the successive projects" and "the strategic resource". Their implementations are made using the Prochain & Pipeline multi-project management tools. From the resulting schedules, the characteristics and deficiencies of each approach are analyzed. Furthermore, these results allow the identification of certain conditions and conflicting effects in the BC sizing and in the criteria used to prioritize activities in the typical methodology implementations. Such observations led to the necessity of complementing the analysis with the aid of probabilistic simulation of the projects' execution stage. The research explains the differences between the effects of dynamic and static prioritization, elucidates the interaction between the capacity and the project buffers. It also examines the effects of using the same percentage of the resource load for sizing the BCs for all the projects, and illustrates the trade-offs between the system stability and the projects makespan.

Keywords

Multi-projects; critical chain; capacity buffer; project management; criteria to prioritize activities.

Sumario

1 Introdução 17
1.1. Motivação do estudo 17
1.2. Objetivo da pesquisa 22
1.2.1. As abordagens de aplicação da metodologia: 22
1.2.2. Os critérios de priorização de projetos: 23
1.2.3. O comportamento dos principais *buffers* de tempo 24
1.3. Delimitação do estudo 25
1.4. Relevância do estudo 26
1.5. Organização do texto 27

2 Contexto e literatura diretamente relacionada 28
2.1. A motivação da pesquisa dentro da literatura de gestão de projetos. 28
2.2. A aplicação da metodologia CC/BM em múltiplos projetos e o estudo da arte 30
2.2.1. Estudos da gestão clássica de projetos individuais 30
2.2.2. A gestão de projetos individuais por meio da CC/BM 31
2.2.3. A gestão de múltiplos projetos por meio da CC/BM 35

3 Conceitos e base teórica 38
3.1. A metodologia da cadeia crítica e gestão de buffers de tempo (CC/BM) 38
3.1.1. Teoria das restrições - TOC (Theory of Constraints) 38
3.1.2. CC/BM em projetos individuais 40
3.1.3. CC/BM em múltiplos projetos 51

4 Metodologia da pesquisa - Efeito das abordagens 57
4.1. O problema-exemplo 58
4.1.1. O Problema original 58
4.1.2. Casos de estudos teóricos: Variação do problema original 60

4.2. As abordagens de aplicação 61
4.2.1. Abordagem de “Todos juntos” 61
4.2.2. Abordagem de “Projetos sucessivos” 61
4.2.3. Abordagem do “Recurso estratégico ou gargalo” 62
4.3. Implementação dos casos de estudo 62
4.3.1. Abordagem de “Todos juntos” 63
4.3.2. Abordagem de “Projetos sucessivos” 68
4.3.3. Abordagem do “Recurso estratégico ou gargalo” 79

5 Efeitos do buffer de capacidade e da priorização de atividades 89
5.1. Efeitos dos *buffers* de capacidade - BC 89
5.1.1. Objetivos 89
5.1.2. Estrutura da Simulação 90
5.2. Efeitos da priorização de atividades. 98
5.2.1. Objetivos 98
5.2.2. Estrutura da simulação 98
5.3. Análise dos resultados 101
5.3.1. Efeitos dos BC 101
5.3.2. Efeitos dos critérios de priorização 113

6 Conclusões e sugestões 118

7 Bibliografia 123

8 Apêndice A: Prochain & Pipeline 125
8.1. Parâmetros de programação 125
8.2. Principais Indicadores 127

9 Apêndice B: Resultados da simulação 128
9.1. Análise das causas de atrasos 128
9.2. *Trade-off* entre *Makespan* e diminuição de atrasos por falta de recurso 131
9.3. Efeitos dos critérios de priorização 133

Lista de Figuras

Figura 1.1: Condições de sucesso dos projetos 19

Figura 2.1: Classificação de projetos. 36

Figura 3.1: Distribuição de probabilidade das durações das atividades 42

Figura 3.2: Execução de atividades com multi-tarefas (*multitasking*) 43

Figura 3.3: Execução de atividades sem multi-tarefas 43

Figura 3.4: Exemplo clássico de gerenciamento de *buffers.* 48

Figura 3.5: O *Programa base* e o *programa projetado* 49

Figura 3.6: Principais elementos da metodologia CC/BM 51

Figura 3.7: Principais elementos da metodologia CC/BM em múltiplos projetos. 56

Figura 4.1: Programa ótimo original 58

Figura 4.2: Programa ótimo modificado 59

Figura 4.3: Representação gráfica das redes dos projetos. 60

Figura 4.4: Incongruência entre importância de atividades e definição da cadeia crítica. Status ao 9/10 65

Figura 4.5: Atrasos nos projetos dois e três. Status no dia 12/10 66

Figura 4.6:Atrasos nos projetos um e dois. Status ao dia 12/10 67

Figura 4.7 : Programa máster nivelando todos os recursos.
Status do dia 09/10 71

Figura 4.8: Reprogramando o máster. Status do 12/10 72

Figura 4.9: Sem reprogramar o máster. Status do 12/10 73

Figura 4.10: Programa projetado sugerido em base a
prioridades fixas. Status ao dia 15/10 74

Figura 4.11: Programa resultante da aplicação
de prioridades dinâmicas. Status do dia 16/10 75

Figura 4.12: Programa máster segundo prioridades 1,3,2.
Status ao dia 9/10 78

Figura 4.13: Programa master baseado no recurso gargalo "A".
Status do dia 8/10 82

Figura 4.14: Diagrama de carga de trabalho, Recurso "A"
(Programa sem BC) 83

Figura 4.15: Diagrama de carga de trabalho, Recurso "B"
(Programa sem BC) 83

Figura 4.16: Diagrama de carga de trabalho, Recurso "C"
(Programa sem BC) 84

Figura 4.17: Programa máster com BC ao 25% .
Status do dia 8/10 85

Figura 4.18: Diagrama de carga de trabalho, Recurso "B"
(Programa com BC 25%) 86

Figura 4.19: Diagrama de carga de trabalho, Recurso "C" (Programa com BC 25%) 86

Figura 4.20: Programa máster com BC 25%. Status do dia 17/10 87

Figura 4.21: Programa máster híbrido. Status do dia 09/10 (Programa projetado com BC 25%) 88

Figura 5.1: Programa base do modelo de simulação 91

Figura 5.2: Distribuição Binomial negativa 94

Figura 5.3: Modelo de simulação dos efeitos do BC 96

Figura 5.4: Modelo de simulação dos efeitos do critério de priorização de atividades 101

Figura 5.5: Consumo de BP - 25% prob. Atraso 103

Figura 5.6: Consumo de BP - 20% prob. Atraso 103

Figura 5.7: Consumo de BP - 15% prob. Atraso 103

Figura 5.8: Consumo de BP - 10% prob. Atraso 103

Figura 5.9: Mitigação dos atrasos por falta de recurso (Projeto dois - 25% de probabilidade de atraso) 105

Figura 5.10: Mitigação dos atrasos por falta de recurso (Projeto três - 25% de probabilidade de atraso) 106

Figura 5.11: Mitigação dos atrasos por falta de recurso (Projeto dois - 20% de probabilidade de atraso) 106

Figura 5.12: Mitigação dos atrasos por falta de recurso
(Projeto três - 15% de probabilidade de atraso) 106

Figura 5.13: Mitigação dos atrasos por falta de recurso
(Projeto dois - 15% de probabilidade de atraso) 107

Figura 5.14: Mitigação dos atrasos por falta de recurso
(Projeto três - 15% de probabilidade de atraso) 107

Figura 5.15: Mitigação dos atrasos por falta de recurso
(Projeto dois - 10% de probabilidade de atraso) 107

Figura 5.16: Mitigação dos atrasos por falta de recurso
(Projeto três - 10% de probabilidade de atraso) 108

Figura 5.17: T*rade-off* entre *makespan* e redução de
atrasos por falta de recurso (25% probabilidade de atraso) 110

Figura 5.18: T*rade-off* entre *makespan* e redução de
atrasos por falta de recurso (20% probabilidade de atraso) 110

Figura 5.19: T*rade-off* entre *makespan* e redução de
atrasos por falta de recurso (15% probabilidade de atraso) 110

Figura 5.20: T*rade-off* entre *makespan* e redução de
atrasos por falta de recurso (10% probabilidade de atraso) 111

Figura 5.21: Caso especial dos efeitos do BC 113

Figura 5.22: Efeitos dos critérios da priorização no sistema 115

Figura 5.23 : Efeitos dos critérios da priorização no sistema 115

Figura 5.24 : Efeitos dos critérios da priorização no projeto 1 116

Figura 5.25 : Efeitos dos critérios da priorização no projeto dois 116

Figura 5.26: Efeitos dos critérios da priorização no projeto três. 116

Figura 9.1: Atraso do projeto um – segunda parte 135

Figura 9.2: Atraso do projeto dois – segunda parte 136

Figura 9.3: Atraso do projeto dois – segunda parte 137

Lista de Tabelas

Tabela 4.1: Dados das redes dos múltiplos projetos 59

Tabela 5.1: Descrição das variáveis que definem o programa de linha de base 91

Tabela 5.2: Descrição das variáveis que definem o programa projetado 92

Tabela 8.1: Parâmetros de programação de projetos individuais - *Prochain* 125

Tabela 8.2: Parâmetros de programação de múltiplos projetos - *Pipeline* 126

Tabela 8.3: Indicadores 127

Tabela 9.1: Componentes de consumo do BP - 25% prob. Atraso 128

Tabela 9.2: Atraso efetivo dos projetos - 25:% prob. de atraso 128

Tabela 9.3: Componentes de consumo do BP - 20% prob. Atraso 129

Tabela 9.4: Atraso efetivo dos projetos - 20:% prob. de atraso 129

Tabela 9.5: Componentes de consumo do BP - 15% prob. Atraso 129

Tabela 9.6: Atraso efetivo dos projetos - 15:% prob. de atraso 130

Tabela 9.7: Componentes de consumo do BP - 10% prob. Atraso 130

Tabela 9.8: Atraso efetivo dos projetos - 10:% prob. de atraso 130

Tabela 9.9: *Trade-off* (25% prob. Atraso) 131

Tabela 9.10: *Trade-off* (20% prob. Atraso) 131

Tabela 9.11: *Trade-off* (15% prob. Atraso) 132

Tabela 9.12: *Trade-off* (10% prob. Atraso) 132

Tabela 9.13: Atraso efetivo do projeto um 133

Tabela 9.14: Atraso efetivo do projeto dois 133

Tabela 9.15: Atraso efetivo do projeto três 134

Tabela 9.16: Atraso efetivo do sistema 134

1
Introdução

1.1.
Motivação do estudo

Os fenômenos de progressiva desintegração vertical das organizações industriais e da customização dos produtos/serviços têm favorecido o crescimento do número de empresas terceirizadas (*outsourced*) que trabalham em desenvolvimento e implementação de projetos. É nesse tipo de empresas que, como explicam Bernardi & Walter (1998), os projetos chegam a atingir tal grau de particularização que são chamados de "sob medida" (*one-of-a-kind*). Essas empresas se enfrentam a um mercado globalizado que exige que cada projeto seja técnica e economicamente competitivo, mantendo uma flexibilidade que lhe permita responder de forma imediata às continuas mudanças dos requisitos do cliente.

Dentro desse panorama, Cohen *et al.* (2004) e Chen (2006) observam que as empresas orientadas à elaboração de projetos necessitam gerir um conjunto de projetos que se desenvolvem simultaneamente num ambiente de recursos limitados. O fato de os vários projetos compartilharem os diversos recursos da empresa especializada constitui uma das vantagens básicas do processo de terceirização ao permitir melhor utilização desses recursos e, conseqüentemente, substanciais economias de escala. Por outro lado, a competição dos diversos projetos pelos recursos comuns e limitados cria complexas interações entre eles. Além disso, tais empresas devem atender um fluxo elevado de projetos que devem ser entregues dentro dos prazos estabelecidos. Tais aspectos se acrescentam aos mais gerais de atingir níveis de qualidade e custos compatíveis com as prioridades estratégicas estabelecidas e para poder sobreviver às exigências financeiras inerentes à execução de cada projeto.

Pela particularidade de cada projeto (e do cliente) é difícil determinar critérios de avaliação sustentáveis na gerência de projetos simultâneos. Teoricamente se estabelece que o sucesso dos projetos está sujeito à satisfação dos

objetivos técnicos dentro do orçamento e prazo estabelecidos. Quer dizer; a avaliação do desempenho dos projetos é expressa em medidas de tempo, escopo definido, havendo uma forte interação entre essas dimensões (LEACH, 2005).

Essas dimensões são representadas na Figura 1.1, onde as três unidades de medida fazem parte de um triângulo eqüilátero no qual o aumento de algum dos lados ocasiona necessariamente mudanças que os outros lados sejam adequados a fim de manter a geometria original. Similarmente o aumento da duração do projeto pode acarretar um aumento do custo e das oportunidades de estender a definição do escopo, tudo em função da necessidade do uso dos recursos disponíveis.

O adequado equilíbrio entre os aspectos de desempenho dos projetos é muito dependente da conjuntura empresarial, das possibilidades abertas e de muitas outras circunstâncias. Além disso, esses *trade-offs* geralmente são muito dinâmicos e dificilmente se mantêm sem mudanças significativas do planejamento à entrega do projeto. É assim que, pela necessidade de definir claramente seu objeto de estudo, a teoria limita a avaliação da eficiência dos planos do projeto e a programação de suas atividades a poucos aspectos. Concentrando-se, geralmente, no tempo total de execução, na pontualidade das entregas, ou no custo.

Mesmo que essa abstração simplificada raramente corresponda ao que se deseja na prática, ela proporciona uma referência para definir a eficiência dos projetos com base no nível de discrepâncias existentes entre o programa planejado e a execução dos projetos. Níveis elevados de discrepância freqüentemente significam perda da racionalidade global do plano e originam deficiências no desempenho na execução do projeto, pelo menos no que concernem aos critérios “tempo” e “custo”.

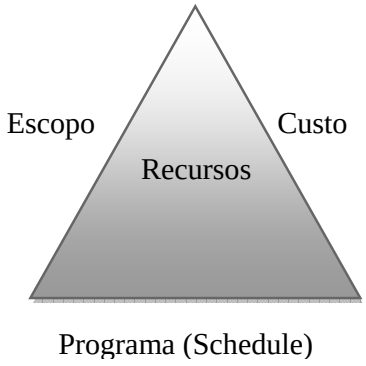

Figura 1.1: Condições de sucesso dos projetos
Fonte: Leach (2005)

Num sistema de produção cujo produto é o desenvolvimento e implementação de projetos, freqüentemente, não estão disponíveis de forma oportuna as informações necessárias para determinar a duração de cada atividade a programar. Tais condições exacerbam a presença de variações na etapa de execução e acompanhamento dos projetos, assim como incertezas não consideradas no cálculo do risco do projeto durante a modelagem do programa, fazem que este seja altamente instável durante a execução. O gerenciamento do risco fica ainda mais complexo quando se trata de um conjunto de projetos em desenvolvimento paralelo, onde as atividades têm que concorrer pela atenção dos mesmos recursos limitados. É assim que a complexidade da gestão de projetos é definida pela necessidade de lidar com altos níveis de incerteza e de priorizar a programação de atividades e recursos (STEYN, 2002). Conseguir administrar essa complexidade atendendo os objetivos da empresa e dos distintos projetos simultâneos, evitando-se sobre-custos, atrasos ou trabalhos deficientes é uma tarefa diária da gerência de projetos. Além desses efeitos indesejados, Leach (2005) menciona outros que fazem parte da problemática da gestão de projetos. Entre eles estão: as disputas entre projetos pelos recursos compartilhados, cancelamento de projetos ainda incompletos, grandes fluxos de caixa negativos por retornos de investimento unicamente ao término simultâneo de vários projetos e o permanente estado de estresse no ambiente de trabalho. Infelizmente, as técnicas mais difundidas no mercado de gestão de projetos (PERT/CPM) não conseguem evitar satisfatoriamente a presença desses efeitos deletérios, motivo

pelo qual, muitas vezes, são utilizadas unicamente para fins de comercialização e não como instrumento de programação e controle.

Diversos autores ressaltam que as deficiências dos métodos clássicos de PERT/CPM se tornam ainda mais críticas no contexto de múltiplos projetos simultâneos.

Alguns desses pontos fracos dizem respeito aos aspectos psicológicos e sociais quase sempre presentes em todas as empresas. Entre os mais conhecidos está a tendência de nunca terminar uma atividade antes da data devida, ainda que isso seja perfeitamente possível. Outro é o hábito de retardar o início de uma tarefa até que tenha decorrido o tempo de segurança adicionado à duração mais provável, degradando, assim, a confiabilidade desejada.

Outra deficiência importante, comumente apontada, é o fato de as técnicas de PERT/CPM focarem nas restrições de precedência, tratando as restrições de recurso apenas de forma acessória. Essa restrição de recursos cria dependências entre tarefas sem relação de precedência que podem ser dominantes e, assim, invalidar a essência da programação das atividades. Mais ainda, se diversos projetos compartilham recursos, então, o que ocorre em um projeto pode, através da indisponibilidade de um recurso, ter um grande impacto em outros projetos. Isso pode ter conseqüências desastrosas porque introduz incertezas externas em cada projeto, e leva a freqüentes remanejamentos de recursos entre projetos. De fato, a tendência é deslocar recursos de projetos com bom andamento para remediar problemas de projetos em situação crítica, azedando o bom relacionamento entre gerentes de projetos.

A metodologia da “Cadeia Crítica e Gestão de Buffers” *(Critical Chain and Buffer Management - CC/BM*) foi proposta por Goldratt em 1997 com base na sua proposta de gestão de gargalos de produção conhecida como Teoria das Restrições (*Theory of Constraints - TOC*). Essa metodologia apresenta alguns elementos adicionais aos geralmente considerados nas técnicas tradicionais de programação de atividades em redes.

Primeiro, ela tem objetivos mais amplos. Ao invés de considerar apenas as questões de custo e de atendimento a prazos (ou minimização do *makespan*), ela procura, também, tratar o problema de estabilidade da programação e fornecer novos elementos para controle gerencial. Isso, é feito através de “*buffers* de tempo” que podem ser vistos como colchões de tempo, que, ao serem agregados

no planejamento do projeto, funcionam como amortecedores que absorvem a diferença existente entre as durações planejadas e executadas das atividades, evitando que a data de entrega prometida seja afetada pelas variações do projeto. Além disso, ela considera explicitamente e de forma orgânica, as restrições de recursos, particularmente, do "recurso gargalo" (conforme definido na TOC) e não apenas as de precedência. Para isso, define "cadeia crítica" (*critical chain*) como sendo o caminho mais longo (em tempo).

Os *buffers* de tempo diferenciam-se das folgas normalmente presentes em PERT/CPM, pois não são conseqüência natural das relações de precedência e das durações das atividades. Eles são propositalmente inseridos com o objetivo de evitar que atrasos em caminhos não críticos venham a atrasar atividades da cadeia crítica e, assim, atrasar a data de término de projetos. Um tipo especial de *buffer* de tempo é o "*buffer* de projeto", colocado ao final da última atividade da cadeia crítica. Esse *buffer* simplesmente atrasa a data de término prevista para o projeto, de forma que o atraso é visto como um fato da realidade.

Além da sua função "isoladora" para evitar que incertezas tornem o programa instável durante sua execução, os *buffers* também têm a função de facilitar o acompanhamento da execução e alertar o gerente para atrasos que estão se tornando críticos. Isso é feito examinando, a cada instante, quais os *buffers* que já foram consideravelmente "consumidos", ou "penetrados" por atrasos nas atividades. Essa penetração de atividade num *buffer* indica quão grave é o atraso em cada ponto crítico do projeto; por exemplo, uma atividade de um caminho não-crítico ameaçando se tornar parte da cadeia crítica, ou seja, se tornar um elemento de atraso da data de término do projeto.

Poderia se argumentar que a abordagem, tradicionalmente utilizada em PERT/CPM, de estimar a duração de cada atividade já com uma margem de segurança para atraso, tem o mesmo efeito isolador. Se, por um lado isso é verdade, por outro tal prática protege cada atividade, não importando se pertence, ou não, à cadeia crítica.

Os aspectos da metodologia relevantes para este estudo são detalhados ao longo deste texto.

1.2.
Objetivo da pesquisa

É objetivo de esta pesquisa estudar a aplicação da metodologia da cadeia critica e gestão de *buffers* de tempo *(Critical Chain and Buffer Management - CC/BM*) na gestão do conjunto de múltiplos projetos simultâneos.

Especificamente, o interesse do estudo é analisar e entender os efeitos que a determinação dos *buffers* de capacidade e dos critérios de priorização de atividades têm na aplicação e no desempenho da metodologia CC/BM em múltiplos projetos. Para melhor entendimento dos objetivos alguns esclarecimentos são necessários.

1.2.1.
As abordagens de aplicação da metodologia:

Além das restrições de precedência, a gestão de múltiplos projetos simultâneos lida constantemente com as diferentes interdependências geradas através das restrições criadas pelo compartilhamento de recursos entre os projetos individuais e pelas prioridades no seqüenciamento das atividades em tais recursos. Frente a essa situação, a metodologia *CC/BM* estabelece um novo conceito de programação que através da inclusão de elementos de mitigação de risco facilite a consecução dos objetivos do sistema e dos próprios projetos. O objetivo de cada projeto é geralmente determinado pelo cumprimento da data de entrega prometida e os objetivos da empresa são ligados à melhor utilização dos recursos, ao fluxo contínuo de projetos no sistema e à minimização de projetos atrasados. Contudo, existe uma nebulosidade na definição dos processos e detalhes de implementação que originam uma brecha entre os conceitos teóricos e sua correta aplicação, abrindo-se espaço à geração de mais de uma alternativa de aplicação prática.

Newbold (1998) definiu os programas "**todos juntos", "projetos sucessivos"** e **"recurso estratégico ou gargalo"** como abordagens de tipos de programação para a aplicação da metodologia CC/BM na gestão de múltiplos projetos. Cada uma dessas abordagens ao definir diferentes critérios de programação, faz uso de um ou mais elementos da metodologia, influenciando seu desempenho em relação aos objetivos de cada projeto e da empresa.

O programa de “todos juntos” inclui as atividades de todos os projetos num único projeto criado artificialmente. Esse programa utiliza uma única cadeia crítica e um único sistema de *buffers* de proteção, que não diferenciam explicitamente os projetos individuais. Os programas de “projetos sucessivos” e de “recurso estratégico ou gargalo” utilizam um programa especial que inclui todos os projetos, corretamente estruturados com suas respectivas cadeias críticas e *buffers* de tempo. O programa de “projetos sucessivos” nivela a carga de trabalho através de todos os recursos do sistema, enquanto o programa do “recurso estratégico” nivela a carga somente através do recurso gargalo, permitindo a existência de sobrecargas no sistema.

Nesse contexto, a pesquisa se concentra em identificar: Quais são as principais diferencias entre as abordagens utilizadas na implementação prática da metodologia CC/BM em múltiplos projetos? E, Que efeito gera sua aplicação, sobre os objetivos dos projetos e do sistema?

1.2.2. Os critérios de priorização de projetos:

A gestão de múltiplos projetos num sistema produtivo é efetuada por meio de um programa global ou “*master*” que integra a gerência dos diferentes projetos individuais que o formam. Nesse programa se costuma utilizar um critério que pretende priorizar os projetos segundo a relevância que eles possuem para a consecução dos objetivos da empresa. Como a programação das atividades é subordinada ao cumprimento das prioridades estabelecidas, a priorização dos projetos é traduzida na instituição de critérios de decisão que facilitem a programação inicial básica do máster. Contudo, na etapa da execução, dependendo das circunstâncias enfrentadas, o efeito dessas decisões pode ser prejudicial para os objetivos de cada projeto.

Ao atribuir a cada projeto uma prioridade fixa como regra de decisão estrita, provoca-se a concentração da atenção nos poucos projetos prioritários, muitas vezes, sacrificando injustificadamente o desempenho dos demais. Por outro lado, a metodologia tem a finalidade de manter a estabilidade dos programas; ou seja, de torná-los robustos e imunes às incertezas e variações que naturalmente surgem durante a execução. Certamente, ao manter fixas as prioridades definidas “a

priori" haverá menos discrepâncias entre a seqüência de atividades executadas e as planejadas, mantendo o sistema estável, mesmo quando estiver fora do prazo. Autores como Cohen *et al.* (2004) assumem, num experimento comparativo entre diferentes metodologias de programação, que o critério próprio da CC/BM, na escolha das atividades a executar, é baseado em priorizações dinâmicas. Esse tipo de priorização classifica as atividades em base à característica crítica das atividades e no nível de consumo dos *buffers.*

Nesse contexto, a pesquisa se concentra em responder se realmente vale a pena executar as atividades com base nas prioridades estáticas ou se a definição de prioridades dinâmicas poderia atingir melhores resultados tanto para o sistema como para cada projeto.

1.2.3. O comportamento dos principais *buffers* de tempo

A incerteza e variabilidade são características inerentes na determinação das durações das atividades programadas em cada projeto. A metodologia CC/BM estabelece os *buffers* ou pulmões de tempo como dispositivos que evitam que a presença de distúrbios na execução dos projetos afete o cumprimento dos objetivos individuais dos projetos e globais do sistema. Dois principais tipos de *buffers* são essenciais para definir a segurança adicionada a cada projeto, o *buffer* do projeto – *BP* e o *buffer* de capacidade – *BC.* O primeiro tem a finalidade de proteger a data de término do projeto de todos os possíveis distúrbios que possam ocorrer na suas próprias atividades, ou como conseqüência de distúrbios em outros projetos. O segundo tipo de *buffer* tem a finalidade de proteger o projeto individual dos distúrbios que possam ocorrer em outros projetos direta ou indiretamente interdependentes; isto é, através dos compartilhamentos de recursos entre os projetos.

O BC, ao ser baseado no recurso gargalo do sistema, é dimensionado como uma porcentagem da sua carga de trabalho, entendendo-se como recurso gargalo o recurso que determina a capacidade produtiva do sistema de produção. A inserção de ambos os *buffers* implica o retardo da data de término do projeto, razão pela qual seus dimensionamentos precisam ser cuidadosamente avaliados, assegurando

o cumprimento das funções de segurança e controle dos projetos, sem comprometer a competitividade da empresa no que diz respeito a prazos.

As técnicas utilizadas para dimensionar o BP baseiam-se numa série de pressupostos estatísticos (incluindo a independência estatística entre atrasos nas atividades) que, mesmo sendo uma aproximação simplificada do problema real, permitem a obtenção de *buffers* funcionais que normalmente, na prática, produzem resultados aceitáveis (NEWBOLD, 1998; LEACH, 2005). Já o dimensionamento dos BC é determinado empiricamente, sendo que uma interpretação errada das interdependências entre projetos ocasionaria seu super-dimensionamento e conseqüentemente datas prometidas de término de projetos excessivamente tardias.

Nesse contexto, a pesquisa se concentra em esclarecer: Qual é a interação prática que existe entre os BP e o BC? Quais são os efeitos que diferentes tamanhos de BC têm sobre a estabilidade do sistema e sobre a duração total ou *makespan* dos projetos? Que efeitos ocasionam a generalização do critério de dimensionamento do BC como uma mesma porcentagem da carga de trabalho do gargalo para todos os projetos?

1.3. Delimitação do estudo

Conforme discutido acima, a implementação da metodologia CC/BM na gestão de múltiplos projetos apresenta diversos aspectos ainda pouco esclarecidos. Examinar, mesmo superficialmente, todos esses aspectos seria tarefa impossível numa pesquisa como esta e, por isso foi necessário reduzir o escopo do estudo. Para tal fim, limitou-se a análise ao estudo do comportamento de três elementos chave na aplicação da CC/BM, a saber: a abordagem (“**todos juntos”, “projetos sucessivos”** e **“recurso estratégico ou gargalo”)**, os *buffers* de capacidade (BC) e os critérios de priorização (como atribuir prioridades às tarefas que competem para execução num mesmo recurso). A análise buscará identificar os efeitos que cada um desses elementos tem na consecução dos objetivos globais do sistema e dos objetivos individuais de cada projeto. Outros elementos da metodologia (como os *buffers* tambor, os *buffers* de alimentação e os critérios para dimensionamento deles) não foram abordados na pesquisa.

A existência limitada de *software* para implementação da metodologia CC/BM adequados para pesquisa e a dificuldade na obtenção de licenças especiais para pesquisas, fez com que o estudo se limitasse à utilização de uma única ferramenta de software, o Prochain & Pipeline (Prochain user's guide). Entretanto, a ferramenta utilizada tem boa aceitação por profissionais e é largamente referenciada na literatura, facilitando a comunicação com outros pesquisadores interessados.

A pesquisa limitou-se à avaliação de problemas desenvolvidos a partir de um único exemplo teórico adaptado de um conhecido estudo de Pritsker B. *et al.* (1969.). Isso com a premissa de obter problemas relevantes e concisos que permitam o fácil reconhecimento e análises dos comportamentos básicos da metodologia e dos seus elementos estudados. A utilização de problemas de maior complexidade de dimensão real não permitiria atingir os objetivos de análise e entendimento da pesquisa, obliterando os resultados de interesse. Não obstante, uma vez realizada esta pesquisa seria interessante sua continuação na aplicação em problemas reais, a fim de verificar a freqüência e a gravidade de ocorrências dos fenômenos aqui estudados.

1.4. Relevância do estudo

Atualmente, as empresas definem que a verdadeira gestão de projetos é, necessariamente, aquela que considera a existência simultânea de múltiplos projetos. Outro tipo de abordagem para o planejamento e controle de projeto é dificilmente aplicável no mundo real. Frente a essa situação, a literatura tem experimentado diferentes abordagens de gestão, entretanto, geralmente baseadas nas metodologias clássicas. (PERT/CPM) e muito pouco no que toca a abordagem da CC/BM. Outros poucos estudos teóricos têm se encontrado sobre o comportamento da CC/BM em múltiplos projetos. Em ambos os casos faltam pesquisas que se orientem à aplicabilidade das teorias estudadas.

É verdade que não existem soluções de gestão e programação que sejam eficientes para todas as empresas, ainda mais, é muito improvável que a mesma solução tenha os mesmos resultados em projetos distintos. Contudo, é necessário o estabelecimento de pautas teóricas e práticas que permitam entender o

comportamento dos elementos da metodologia CC/BM e reconhecer os possíveis efeitos que as diferentes decisões de aplicação possam causar no sistema. É assim que esta pesquisa se faz relevante para estabelecer um ponto de partida no estreitamento da brecha entre a teoria e prática da gestão de múltiplos projetos, além disso, aporta novos aspectos para o entendimento dos principais elementos da metodologia pouco explorados na literatura.

1.5. Organização do texto

Este trabalho é estruturado em seis capítulos, sendo que o primeiro è a introdução recém apresentada. No segundo capítulo é realizada uma revisão bibliográfica relativa às metodologias de gestão de projetos tanto em projetos individuais como em múltiplos projetos. A finalidade desse capítulo é enquadrar a motivação e a problemática que deram origem a esta pesquisa dentro do estado da arte. No Capítulo três se apresentam os conceitos da metodologia da cadeia crítica (CC/BM) aplicada em projetos individuais e múltiplos projetos. O objetivo do capítulo é estabelecer a base teórica que permita o acompanhamento da pesquisa.

Nos Capítulos quatro e cinco se apresenta a análise central da pesquisa; o capítulo quatro se concentra no estudo das abordagens de aplicação da metodologia CC/BM e o Capítulo cinco aborda o estudo dos *buffer* de capacidade – BC e dos critérios utilizados na priorização de atividades a executar. Nesse último capitulo se apresenta tanto os elementos e definições que estruturam o experimento de simulação, como a análise dos resultados obtidos. No Capítulo seis são apresentadas as conclusões obtidas e algumas sugestões para futuras pesquisas

2
Contexto e literatura diretamente relacionada

Este capítulo apresenta os principais trabalhos, diretamente relacionados à pesquisa, encontrados na literatura de gestão de projetos. Em seguida, apresenta-se a relevância que o estudo das abordagens de aplicação, priorização de atividades e do comportamento dos principais *buffers* de tempo tem dentro da literatura que engloba a metodologia da CC/BM em múltiplos projetos.

2.1.
A motivação da pesquisa dentro da literatura de gestão de projetos.

Para melhor compreender a complexidade do sistema de gerenciamento e descrever o comportamento do tipo de projeto que motivam o estudo, utiliza-se a classificação sugerida por Hans *et al.* (2007) Como mostra a Figura 2.1, essa classificação é baseada nos graus de variabilidade e de dependência característicos dos projetos, esquematizando-se quatro quadrantes diferenciados. A motivação desta pesquisa surge nos projetos localizados no quadrante extremo **AA**. Esses projetos se caracterizam por ter, de um lado, uma complexidade elevada devido às dependências das influencias internas (alto nível de compartilhamento de recursos entre projetos) e externas (necessidade de comprar equipamentos, tecnologias, subcontratar pessoal, etc.), e de outro lado, uma alta variabilidade decorrente da incerteza presente na informação que se tem das atividades no momento da programação.

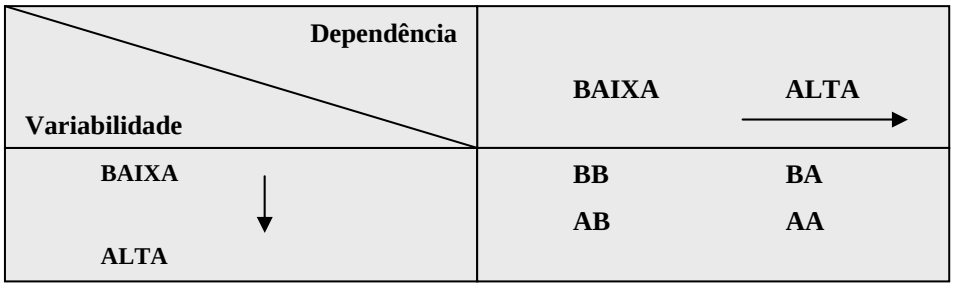

Figura 2.1: Classificação de projetos.
Fonte: Hans *et al.* (2007).

Metodologias clássicas como PERT/CPM, ao não considerarem explicitamente as restrições de recursos limitados e focalizarem o controle do risco isoladamente em cada atividade do programa, fazem que para o tipo de problema AA, elas tendam a ser utilizadas apenas para apresentar programas de trabalho aos clientes, mas não para efetivamente controlar os projetos. Como alternativa para este tipo de projetos de alta variabilidade e complexidade, recentemente surgiu a metodologia CC/BM (Programação por Cadeia Critica e Gestão de *Buffers*,) criada por Goldratt em 1997. Baseada na teoria das restrições (TOC, *Theory of Constraints*), ela utiliza o conceito de **cadeia crítica** e administração de ***buffers* de tempo** para a programação e controle de projetos. Segundo Rand (2000) e Steyn (2002), esta metodologia pretende cobrir as deficiências crônicas dos projetos (exceder prazos limites, exceder orçamentos e reduzir alcances do projeto) que as metodologias e abordagens anteriores não têm conseguido minorar, mesmo utilizando os sistemas de *software* mais sofisticados e caros. O surgimento desta metodologia vem de uma visão divergente que segundo Rand (2000), leva em conta aspectos psicológicos e técnicos, imersos no desenvolvimento de projetos. Entretanto, sua adoção tem ocorrido com cautela, sendo ainda poucas as avaliações de sua eficácia prática, assim como os estudos sistemáticos e analíticos a seu respeito. Faz-se, assim, necessária a realização de pesquisas que esclareçam suas vantagens e limitações práticas nos diferentes ramos que a gestão de projetos alcança, assim como indicações úteis para orientar os gerentes em diversos aspectos práticos de sua aplicação (ROZENES et al, 2006; STEYN, 2002),

2.2.
A aplicação da metodologia CC/BM em múltiplos projetos e o estudo da arte

O estudo da programação conjunta de múltiplos projetos é abordado como uma complicação do problema original em projetos individuais, mas ainda há pouco conhecimento no que se refere às suas particularidades.

2.2.1.
Estudos da gestão clássica de projetos individuais

Na programação de projetos individuais existem metodologias bem conhecidas e difundidas, como PERT (*Program Evaluation and Review Technique*) e CPM (*Critical Path Method*), ambas desenvolvidas quase ao mesmo tempo (*circa* 1957), mas com finalidades distintas. O PERT foi criado como técnica de controle dos tempos de execução de atividades de projetos especiais, e o CPM como ferramenta para controlar e otimizar os custos de operação do projeto. Uma diferença entre elas está na forma de estimar as durações das atividades, o CPM usa tempos determinísticos e o PERT tempos probabilísticos com estimativas de tempos otimistas, pessimistas e mais prováveis. A solução em ambos os métodos é determinada com base nas relações de precedência das atividades, supondo que os recursos estarão sempre disponíveis quando forem necessários, premissa, esta dificilmente observada durante a execução do programa. (PATTERSON, 1984).

Mesmo assim, segundo Jyh-Bin (2007), o método foi considerado o mais bem sucedido durante muito tempo e, ainda agora, depois de ter passado por importantes evoluções e adequações é base da maioria de sistemas de *software* dedicados à gestão de projetos. Observa-se, entretanto, que as condições vigentes quando a técnica foi concebida também evoluíram, sem que fossem, necessariamente, bem acompanhadas pelo método. O novo conceito de gestão de projetos responde as exigências competitivas do mercado atual. Num ambiente competitivo se estabelecem prazos de entrega mais curtos associados a níveis de custo e qualidade ótimos, exigências que, na gestão de projeto, são traduzidos como níveis de risco mais altos.

Ante esta situação, Rand (2000), Steyn (2002) e Raz *et al.* (2003), explicam como metodologias como PERT/CPM, que não consideram explicitamente e

adequadamente a restrição de recursos na sua solução, fazem com que o resultado seja ineficiente para controlar o risco peculiar de cada projeto, assim como para manter, ao nível organizacional, um fluxo equilibrado de projetos. Desta maneira, a literatura define como problema principal da gestão de projetos a tomada de decisão sobre que atividade, dentro de um conjunto de opções, atrasar caso exista um conflito de recursos. Problema conhecido formalmente como "o problema de programação de projetos com recurso limitado" ("*the resource-constrained project scheduling problem*").

Desde então, numerosos estudos têm se dedicado ao problema, utilizando-se desde programação linear até os mais criativos procedimentos heurísticos, mas sempre com base nos princípios da metodologia PERT/CPM. Essa tentativa de abordar o problema de programação de projetos com recurso limitado se estendeu também na prática, quando os sistemas de *software* para gestão de projetos começaram adicionar às aplicações da metodologia PERT/CPCM funções de programação com nivelamento de recursos. Segundo Herroelen & Leus (2001) os sistemas comerciais priorizam a seqüência de atividades mediante regras simples que nos melhores casos podem atingir projetos 5% mais longos do ótimo, criando seqüências deficientes que produzem caminhos críticos e tempos de folga irreais. Isso, acrescido ao erro na estimação das durações de cada atividade, origina um programa baseado num somatório de incertezas. Como a metodologia PERT/CPM centraliza o controle do projeto localmente em cada folga de atividade, torna-se difícil identificar oportunamente as fontes de problemas e riscos de atraso. (RAND, 2000; JYH-BIN, 2007).

2.2.2. A gestão de projetos individuais por meio da CC/BM

Como tentativa de cobrir as ambigüidades e deficiências presentes na gestão de projetos surge a metodologia da cadeia crítica e gestão de *buffers* (CC/BM). Essa nova metodologia, criada por Eli Goldratt (1997), estabelece um algoritmo de programação aplicado tanto a projetos individuais como a sistemas de múltiplos projetos. Como Rand (2000) explica, a CC/BM é baseada nos princípios básicos da **teoria das restrições**, a qual tem como finalidade aumentar o fluxo de produção (*output*) do sistema. O processo da teoria se inicia com a identificação

da restrição ativa dominante, ou seja, o recurso gargalo que limita o *output* do sistema. Em seguida, programa-se o recurso gargalo de forma que sua produção seja máxima, para depois programar as demais atividades e recursos em função da programação ótima do gargalo. Se, na programação completa resultante, for identificado outro gargalo, este levará a segunda prioridade de programação, e assim por diante.

A premissa da aplicação deste processo no cenário de gestão de projetos é a definição de metas globais e, não, locais, de forma que a cada decisão se considere os efeitos que se terá no sistema de projetos. É assim, que ao aplicar a metodologia CC/BM na programação de projetos individuais, se concentra a atenção nos objetivos do projeto total e não no cumprimento das datas de início e término das atividades que o formam. Ao estender a aplicação da metodologia para múltiplos projetos, priorizam-se os objetivos do sistema como conjunto, supondo que os objetivos dos projetos individuais são garantidos no passo anterior. Será visto nos Capítulos IV e V como esse suposto nem é sempre adequado durante a etapa de execução dos projetos.

Para cumprir os objetivos do projeto e do sistema, a CC/BM implementa no seu método elementos de proteção que mitigam o efeito dos atrasos das atividades evitando o atraso da entrega dos projetos. Segundo, como explicam Herroelen & Leus (2001) a proteção dos projetos responde a atrasos geralmente originados em aspectos comportamentais psicológicos dos trabalhadores. Estes aspectos psicológicos seriam explicados pelos fenômenos das “Síndrome do Estudante” e pela “Lei de Parkinson**”,** a partir dos quais se infere que o conhecimento prévio da existência de um tempo de segurança embutido nas datas estabelecidas para cada atividade gera a tendência de postergar até o ultimo momento a execução das atividades “menos prioritárias”.

Ao considerar tais conceitos na determinação da segurança dos projetos, a metodologia se torna adequada para gestão de projetos de alta complexidade (quadrantes **BA** e **AA** na Figura 2.1), especialmente quando o recurso crítico, ou gargalo, é humano. Uma vez que a metodologia se concentra na proteção do cumprimento das datas de entrega prometidas nos projetos, faz-se necessário o entendimento da origem das incertezas presentes neles. Para isso, a metodologia

se baseia na famosa “Lei de Murphy”, que explica como a execução do projeto é afetada pelo desconhecimento exato de quando começar o preparo das atividades dependentes da duração variável da atividade anterior. Embora ainda exista muito por pesquisar sobre a modelagem da variabilidade e incertezas na execução de projetos, autores como Leach (2005) afirmam que os efeitos delas são ampliados quando a segurança do projeto é distribuída e alocada a cada atividade (método PERT/CPM), provocando-se tempos ociosos e atrasos na execução dos projetos, dificultando a continuidade de trabalho nos projetos.

Baseando-se na concepção de gestão de projetos através da proteção das datas de entrega, acrescentaram-se novos objetivos na avaliação do desempenho da programação (*schedule*),. tais como: a minimização da quantidade de trabalho em processo (WIP- *Work in Process*) no sistema, a minimização do tempo de produção dos projetos, e a continuidade na carga e no fluxo de trabalho. (HERROELEN & LEUS 2001).

Para atingir um programa individual que cumpra com as exigências de proteção da metodologia, definiram-se dentro da sua estrutura elementos fundamentais como a **cadeia crítica, *buffer* do projeto –BP, *buffers* de alimentação – BA e *buffers* de recursos - BR.** Com base da cadeia crítica, definida como a seqüência de atividades que determina a duração do projeto (incluindo-se as restrições de precedência e de recursos), se determina a duração e localização do BP. A finalidade desses *buffers* é assegurar que a data de entrega do projeto seja cumprida. Para tal fim, este é constituído pelos **tempos de segurança** que foram desconsiderados inicialmente em cada atividade e localizado no final da cadeia. Os BA têm a finalidade de proteger a cadeia crítica dos atrasos nas atividades não críticas, e por isso, inseridos nas uniões entre as cadeias não críticas e a cadeia crítica, ou seja, sempre que uma atividade não crítica desemboque na cadeia crítica. Os BR têm a finalidade de informar ao recurso estratégico quando começar a se preparar para trabalhar na seguinte atividade. (HERROELEN & LEUS 2001; HERROELEN *et. al*., 2002).

Além dos elementos estruturais, a metodologia estabelece a utilização de um “programa alternativo” que reflete o estado do projeto na etapa de execução, mesmo que ele não seja considerado por todos os autores, Herroelen & Leus

(2001), Herroelen *et al.* (2002) o incluem como elemento importante para a consecução dos objetivos anteriormente definidos. Esse programa alternativo, chamado de **programa projetado** é implementado nos principais sistemas de *software* que aplicam a metodologia. Herroelen *et al.* (2002) numa análise dos diferentes elementos que formam a CC/BM, lamentam a falta de interesse da metodologia pelos algoritmos usados na solução do problema do *Resource-Constrained Project Scheduling Problem* (RCPSP), afirmando que a determinação da seqüência de atividades (no recurso gargalo) e conseqüentemente, o desempenho do projeto, dependem diretamente do algoritmo utilizado. Da mesma forma, os projetos são afetados pelo desempenho dos sistemas de *software* atuais, que aplicam algoritmos de RCPSP como uma caixa preta, gerando soluções, que segundo Herroelen & Leus (2001) e Herroelen (2005) pioram no desempenho na medida em que aumenta o número de atividades e o número de tipos de recursos.

Contudo, a metodologia alega que seu interesse está nas técnicas de proteção do programa que assegurem o cumprimento do prazo de entrega estabelecido, sem importar se o método de programação é ótimo ou aceitável. Com essa lógica, o estudo das técnicas do tratamento da incerteza e variabilidade se faz relevante. Em princípio, a proposta original de Goldratt utiliza *buffers* que sejam 50% da duração da cadeia à qual pertencem. Isso pode resultar em projetos muito longos de provável rejeição pelos clientes. Um método alternativo como o da “raiz quadrada do erro” calcula a raiz quadrada da soma dos quadrados dos erros em atividades individuais (erro é a diferença entre durações das atividades que consideram seguranças e durações enxutas) determina o tamanho do *buffer* com base ao risco acumulado ao longo da cadeia que o alimenta (HERROELEN & LEUS, 2001). Isso, é claro, tem como premissa a independência probabilística entre as durações.

Ambas as técnicas são incluídas em sistemas de *software* que implementam CC/BM. Não obstante, na procura de *buffers* que reflitam melhor as incertezas do projeto, Kjersti & Taylor (1999) propõem um método de simulação simples que determina *buffers* mais enxutos baseados unicamente no comportamento do projeto, outras propostas como uso de técnicas alternativas como teoria de filas foram apresentadas por Hans & Herroelen *et. al*. (2007).

2.2.3.
A gestão de múltiplos projetos por meio da CC/BM

Essa deficiência de centralizar o controle do projeto localmente em cada folga de atividade se vê acentuada em sistemas de múltiplos projetos, principalmente, quando o nível de compartilhamento de recursos é elevado e complexo. (RAND, 2000; JYH-BIN, 2007)

Ao ampliar o escopo das pesquisas para um marco de múltiplos projetos, o problema objeto de estudo interessou-se pela definição de uma metodologia de gestão que permita uma melhor coordenação entre os vários projetos planejados e executados simultaneamente, definindo-se a definição organizacional, ou de planejamento, da gestão de múltiplos projetos.

Kurtulus & Davis (1982) definem duas abordagens na programação de múltiplos projetos. Numa os projetos estão ligados unicamente pela dependência de um *pool* de recursos disponíveis e a visualização do programa é através de múltiplos projetos com vários caminhos críticos. Na segunda abordagem os projetos são combinados artificialmente num único projeto, adicionam-se atividades fictícias de inicio e fim, e o controle é feito como se fosse um único projeto com um único caminho crítico.

Hans *et al.* (2007) apresentam uma evolução das abordagens de planejamento mais relevantes. O uso de níveis hierárquicos de planejamento vão desde um único nível, onde se utiliza métodos de programação dinâmica na obtenção de uma seqüência ótima que minimize o custo total, até o uso de três níveis hierárquicos diferenciados. Nesse último, utilizam-se relacionamentos entre macro atividades que representam os projetos individuais já programados em base a CC/BM (incluindo a inserção de *time buffers* de proteção e subordinando a programação à seqüência do recurso gargalo). Essa abordagem tem como objetivo geral maximizar o valor presente líquido do conjunto de projetos, mediante objetivos específicos de atingir projetos de menor duração.

Newbold (1998) se concentra nas abordagens de aplicação da CC/BM, resumindo-as a três mais importantes: A abordagem de “todos juntos”, dos “projetos sucessivos” e do “recurso estratégico ou gargalo”. A primeira se diferencia por usar um único programa artificial como base de programação. As duas últimas gerenciam os programas (*schedules*) individuais nivelando a carga de trabalho dos recursos através do sistema sem alterar sua estrutura individual. A abordagem de “projetos sucessivos” nivela todos os recursos do sistema, sem permitir qualquer tipo de sobrecarga, enquanto a abordagem do recurso estratégico somente nivela a carga do recurso gargalo, permitindo a existência de outros recursos sobrecarregados.

No atual estado da pesquisa, dependendo da abordagem utilizada e do conjunto de regras de priorização de atividades, pode-se chegar a programas muito diferentes, fazendo com que a confiança na eficiência dos resultados seja reticente devido à ambigüidade imersa tanto na programação como na avaliação de prioridades.

Kurtulus & Davis (1982) explicam a necessidade de estabelecer categorias de projetos com regras de priorização específicas para cada uma. Continuando com essa lógica na aplicação da CC/BM, Cohen *et al.* (2004) realizam experimentos para comparar diferentes metodologias de programação, em que o critério para a escolha das atividades a executar na metodologia CC/BM, é baseado em priorizações dinâmicas. Esse tipo de priorização classifica as atividades com base na característica crítica das atividades e no nível de consumo dos *buffers.*

Contudo, a metodologia CC/BM e as ferramentas de *software* a ela relacionadas, aplicam o critério de priorização fixa, baseado na organização prioritária dos projetos que são programados simultaneamente. (NEWBOLD 1998, LEACH 2005). Tais ferramentas de software, mesmo fornecendo uma série de relatórios sofisticados, ainda são deficientes na identificação oportuna das atividades que poderiam prejudicar os projetos e, mais ainda, na prevenção dos possíveis efeitos que possam desencadear-se nos diferentes projetos do sistema. (KOLISH, 1999; HERROELEN, 2005)

Trabalhos que aprofundem sobre o comportamento dos elementos de programação da metodologia CC/BM em múltiplos projetos (como são os *buffers* de capacidade – BC, *buffers* tambor - BT e da mesma **seqüência crítica**), são escassos na literatura pesquisada. A literatura revisada aborda a programação de múltiplos projetos somente como um complemento teórico do problema de programação individual. Não foram também encontrados estudos que abordem diretamente a aplicabilidade e implementação da CC/BM.

3
Conceitos e base teórica

Este capítulo estabelece os conceitos necessários para entender os elementos, os processos e a finalidade da metodologia da cadeia crítica CC/BM, tanto em projetos individuais como em ambientes de múltiplos projetos. Essa base conceitual é necessária para o acompanhamento dos capítulos seguintes

3.1.
A metodologia da cadeia crítica e gestão de buffers de tempo (CC/BM)

A metodologia da cadeia crítica surge como uma teoria alternativa ante a necessidade de aprimorar os resultados da abordagem clássica na gestão de projetos. Essa metodologia nasce da aplicação da "teoria das restrições" num sistema de planejamento e execução de projetos.

3.1.1.
Teoria das restrições - TOC (Theory of Constraints)

Segundo Leach (2005) a teoria é estruturada no entendimento de qualquer sistema produtivo. Estabelecendo, que todos os sistemas produtivos possuem, no mínimo, uma restrição que limita suas saídas ou *outputs*. Quer dizer, que se o sistema estivesse livre de restrições teria saídas infinitas ou nulas. A finalidade da TOC é aproveitar ao máximo a capacidade do sistema, fazendo que as saídas atinjam a meta estabelecida. A teoria aceita que qualquer otimização isolada das partes pode não gerar uma melhora nos resultados do sistema e que toda solução aplicada ao sistema tende a se deteriorar com o tempo em conseqüência das mudanças do ambiente. Essas premissas básicas fazem com que a teoria estimule uma melhora contínua dos processos e resultados do sistema através do ciclo persistente de procura e eliminação de novas restrições.

Com raiz nesses conceitos sistêmicos Goldratt cria o método de controle de produção chamado *Drum-Buffer-Rope*. Este método é composto pelo "tambor"

(*drum*) que representa a capacidade de processamento da restrição do sistema pela "corda" ou *rope*, que representa a informação do estado do tambor que libera o ingresso de trabalho no sistema com o objetivo de não deixar a restrição ociosa, finalmente, pelos *buffers*. Estes ao serem localizados em lugares estratégicos absorvem as flutuações não previstas no processo. (NEWBOLD, 1998; LEACH, 2005)

Leach (2005) explica como esse método se amolda as condições do mundo dos projetos, onde certamente existem restrições e flutuações estatísticas que evitam o término do projeto em menor tempo. Neste contexto a *restrição*, ou *tambor*, do sistema corresponde à cadeia crítica, os *buffers* são equivalentes a pulmões tempo e a "corda" ou *rope*, é a ordem de liberação de projetos baseada na capacidade disponível dos recursos, na execução de atividades em base ao estado dos *buffers* e na decisão de reprogramar a rede. Repare-se que para aplicar estes conceitos no mundo dos projetos é preciso aceitar que cada projeto é um sistema onde existem dependências entre seus elementos, e que toda mudança de atividades, recursos, políticas, etc. afeta o sistema.

A teoria define cinco passos importantes no processo de melhora contínua do sistema:

1 Identificação da restrição do sistema: Nos projetos a restrição do sistema comumente é formada pelos recursos e interdependências entre atividades que tem o impacto mais negativo em função de tempo, custo e escopo do projeto.

2 Exploração da restrição: O objetivo é desenvolver um programa (*schedule)* que permita obter o maior proveito da restrição mantendo-a o mais ocupada possível.

3 Subordinar tudo à restrição: Focalizar a atenção na restrição fazendo que todos os elementos do sistema trabalhem em função dela. Ou seja, programar todas as atividades e recursos não restritivos em função da cadeia ou seqüência crítica.

4 Elevar a restrição: Aumentar recursos e/ou resolver dependências das atividades.

5 Se a restrição é resolvida ir ao passo um.

3.1.2.
CC/BM em projetos individuais

Para poder entender os elementos técnicos e a estrutura de programação da metodologia é preciso entender primeiramente sua base teórica. Nesse sentido, Leach (2005), Herroelen *et al.* (2002) e outros, explicam a presença constante de atrasos e sobre-custos na execução de projetos por meio da combinação de três fenômenos: “A síndrome do estudante”, “A lei de Parkinson”, e a conhecida “Lei de Murphy”.

A **síndrome do estudante** se refere à tendência que as pessoas, especialmente as que estão constantemente ocupadas, têm para esperar que as atividades se tornem realmente urgentes antes de iniciá-las. Leach (2005) afirma que, geralmente, o trabalho realizado nos primeiros dois terços da duração total da atividade é menor a um terço do total do trabalho. Esta situação faz com que a probabilidade de reconhecer alguma variação positiva ou adiantamento na execução das durações das atividades seja extremamente pequena. Isso explicaria o motivo pelo qual raramente se observaram adiantamentos no termino das tarefas. A **lei de Parkinson** diz que o trabalho tende a se estender até cobrir (e muitas vezes exceder) todo o tempo permitido. Tal síndrome se vê reforçada por políticas comumente implantadas na gestão de projetos, onde o término antecipado de alguma tarefa é atribuído a uma estimativa excessivamente conservadora e significa um posterior aumento da carga de trabalho para o executor ou uma redução nas futuras estimativas de tempo. A **lei de Murphy** se refere à incerteza presente no projeto, esta assume que “*tudo que pode sair errado, certamente acontecerá*”. O verdadeiro sentido dessa lei é reforçar a necessidade de modelar um programa que proteja o cumprimento da meta estabelecida (em tempo, custo e escopo) das possíveis variações que possam acontecer durante a etapa da execução.

Segundo Herroelen *et al.* (2002) e Leach (2005), para evitar os efeitos não desejados resultantes da interação dos fenômenos mencionados anteriormente, é preciso observar três cuidados básicos dentro do programa:

- **Estimar as durações das atividades considerando apenas 50% de probabilidade do projeto não atrasar**.

  Tanto Leach (2005) quanto Herroelen & Leus (2001) e Herroelen *et al.* (2002) reconhecem que a duração das atividades tem um caráter estatístico com uma distribuição de probabilidade similar à da Figura 3.1. É assim que, tanto durações extremamente curtas, quanto durações extremamente longas têm uma chance apreciável de acontecer. Nesse contexto Leach (2005) adverte que as pessoas tendem a definir durações de atividades superestimadas com a finalidade de se proteger de futuras cobranças de desempenho. Isso ocasiona que a estimação da data de término do projeto seja exagerada, ademais, o fato de localizar essa proteção em cada atividade permite a ocorrência dos efeitos não desejados originados pela lei de Parkinson. A metodologia recomenda mitigar tais efeitos usando a média ou "tempo seco" na estimação das durações das atividades. Ou seja, subtrair o tempo de contingência considerado numa estimação protegida, a qual geralmente considera o 90% de confiança admitindo 10% de probabilidade de atraso. Tal contingência será posteriormente realocada num *buffer* específico que permita controlar melhor a variância inerente do projeto.

  Segundo Leach (2005) essa variância é originada por causas comuns ou naturais do sistema, sendo que sua gestão é o elemento essencial para o sucesso dos projetos. Por outro lado, Herroelen *et al.* (2002) questionam a utilidade de tal abordagem, alegando que durante a etapa de execução, certamente as distribuições das atividades não serão as mesmas.

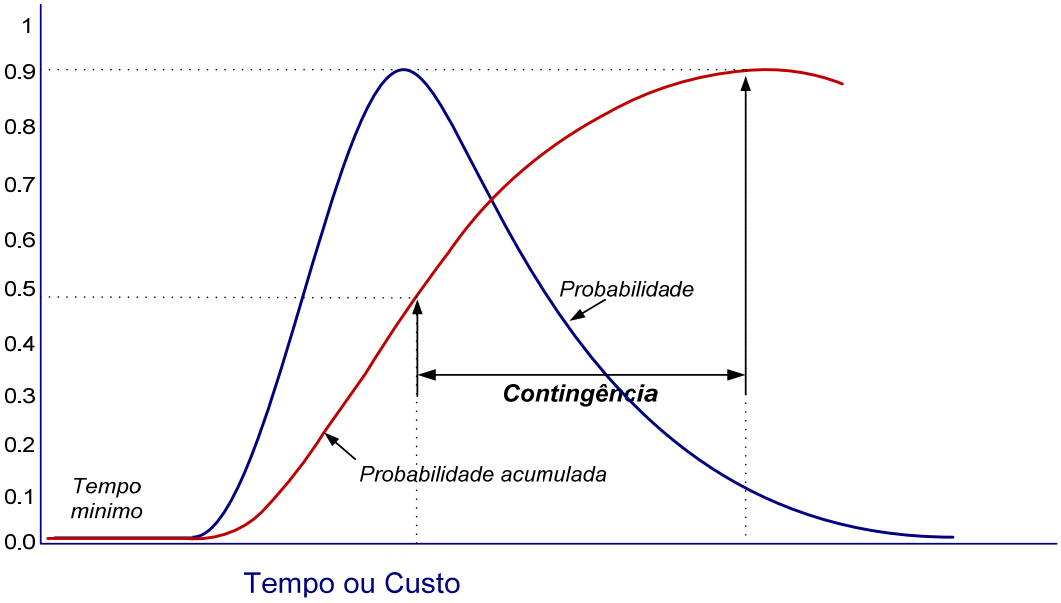

Figura 3.1: Distribuição de probabilidade das durações das atividades
Fonte: Leach (2005)

- **O programa não deve estabelecer datas para prometidas individuais, *due dates* ou *milestones*, que são atividades sem duração que marcam pontos de controle intermediários no projeto**.

  O programa da cadeia crítica estabelece datas apenas para o início das primeiras atividades de cada cadeia e para o fim do projeto (depois do buffer do projeto), para o resto das atividades se utilizam datas aproximadas que indiquem o início e o tempo faltante para completá-las. Isso se baseia na idéia de que o uso de *milestones* e *due dates* origina objetivos locais, fazendo com que os objetivos do sistema sejam relegados a segundo plano. Ademais, os controles intermediários evitam que ganhos de tempo obtidos por atividades terminadas antecipadamente sejam transmitidos ao longo das atividades sucessoras da cadeia, quer dizer, que mesmo tendo a atividade pronta para ser elaborada pelo recurso da seguinte atividade, este não estará preparado para trabalhar nela, pois a programação não contempla tal possibilidade. Conseqüentemente, o uso desses controles intermediários tende a facilitar o consumo do tempo disponível para executar uma atividade, tal como prevê a lei de Parkinson. (ANAVI-ISAKOW & GOLANY, 2003)

- **O programa não deve permitir a execução de múltiplas tarefas, ou seja, um único recurso não deve interromper uma atividade para fazer outra**.

  A origem desse comportamento é a pressão que os clientes e os encarregados dos projetos fazem sobre os recursos para priorizar seu projeto. Geralmente se pensa que trabalhar com múltiplas tarefas incrementa a eficiência e produtividade total. Leach (2005) entre outros mostram como dividir os esforços entre atividades, faz com que as atividades e os projetos sejam acabados mais tarde. Comparando a Figura 3.2 e Figura 3.3 se aprecia como os projetos, ao trabalhar-se simultaneamente em múltiplas tarefas, são estendidos consideravelmente. A duração dos projetos tende a aumentar quando se leva em conta os tempos de *set up* (tempo que demora o recurso para retomar a atividade interrompida).

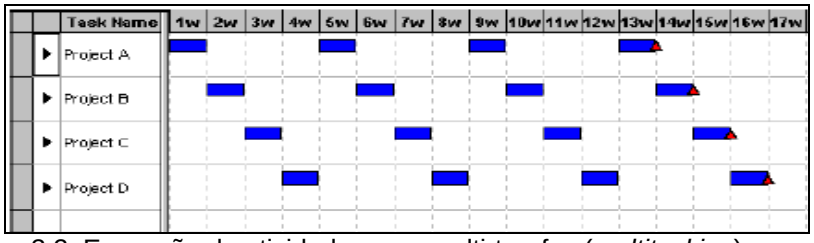

Figura 3.2: Execução de atividades com multi-tarefas (*multitasking*)
Fonte: http://www.sciforma.com

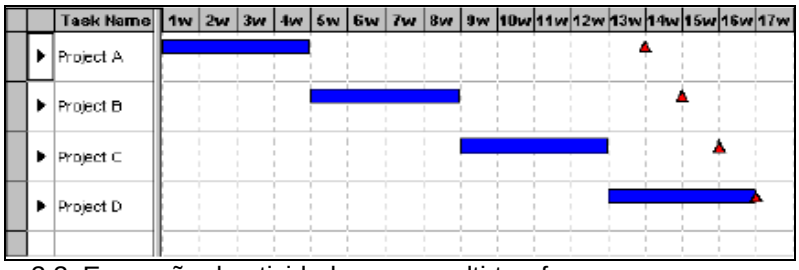

Figura 3.3: Execução de atividades sem multi-tarefas
Fonte: http://www.sciforma.com

3.1.2.1. Principais Elementos

Uma vez estabelecidas as condições básicas da metodologia, pode se definir os principais elementos que, junto ao processo de programação pretendem determinar uma data de termino de projeto que seja mínima e muito

provavelmente atingível. Para tal fim, a metodologia estabelece como premissa a necessidade de outorgar estabilidade ao programa mantendo níveis mínimos da carga de trabalho-em-processo (*work-in-process, WIP)*. (ANAVI-ISAKOW & GOLANY, 2003). A seguir alguns elementos são definidos de forma a tornar mais claro o resto deste texto.

- **Programação inicial de atividades**. - A metodologia recomenda programar as atividades usando as datas de termino mais tarde (*late-finish)* que permitam atingir a data de entrega prometida, levando em conta as restrições de precedência e de compartilhamento de recursos. Com base na programação "de trás para frente" se executam as atividades segundo o "sistema de produção puxado". É assim que se reduzo o nível de carga de trabalho-em-processo concentrando a maior quantidade de trabalhos simultâneos no final do programa, ou quando é estritamente necessário. Para reforçar tal prática, a metodologia estabelece que as atividades iniciais (todas aquelas que não têm predecessor) devem ser executadas exatamente na data estabelecida, e, as atividades intermédias, (que tem predecessores), devem ser executadas o mais cedo possível. Além disso, a maneira de controlar melhor a carga de trabalho, é necessário que a programação de atividades seja acompanhada pela resolução de conflitos entre os recursos, a qual também será feita "de trás para frente". Para lograr este nivelamento de recursos, Herroelen *et al.* (2002) mencionam a utilização de heurísticas, não bem definidas na literatura, que certamente implicam trasladar para a esquerda (mais cedo no tempo) as atividades que provocam o menor aumento da duração do projeto.

- **A cadeia crítica. -** A cadeia crítica é a seqüencia de atividades que, considerando as restrições de precedência tecnológicas e de dependência entre atividades pelo compartilhamento de recursos, determina a duração total do projeto. Essas atividades se caracterizam por não ter espaço suficiente para serem deslocadas no tempo, nem para o futuro (por ser programada com os tempos mais tarde) nem para o passado. Ou seja, são atividades sem folga que fazem parte da cadeia mais longa do programa.

De acordo com a “teoria das restrições”, é preciso subordinar o programa a um único elemento mais restrito, por conseguinte, a restrição do programa corresponde a uma única cadeia crítica. Caso exista mais de uma cadeia sem folga e com a mesma duração, será preciso escolher só uma delas. (NEWBOLD 1998; HERROELEN *et. al.*, 2002; ANAVI-ISAKOW & GOLANY, 2003; LEACH, 2005)

- ***Buffers*. –** Segundo Herroelen & Leus (2001) **s**ão pseudo-atividades que ao serem inseridas no programa fazem as funções de “estoques de tempo”. A inserção desse tempo extra no projeto procura assegurar a satisfação dos objetivos do sistema, evitando, tanto quanto, que as datas de entrega prometida sejam afetadas pelos possíveis distúrbios ou atrasos que possam acontecer durante a etapa de execução. Alem de proteger o programa, os *buffers* desempenham a função de controle durante o acompanhamento do projeto. Conforme se verá à continuação a alocação desta segurança em lugares estratégicos do programa permite obter informações oportunas sobre o estado do projeto e de suas partes críticas, facilitando o planejamento e aplicação de medidas corretivas quando o projeto estiver fora de controle. De acordo com a localização, definem-se três diferentes tipos de *buffers*: *Buffers* do projeto, *Buffers* de alimentação e *Buffers* do recurso.
  - O ***buffer do projeto – BP*** tem como função proteger o projeto de atrasos que possam acontecer na cadeia crítica. Para tal fim, eles são localizados logo depois da última atividade crítica, definindo assim, a data de entrega do projeto.
  - Os ***buffers de alimentação – BA*** têm a função de proteger a cadeia crítica de atrasos que possam acontecer nas atividades não críticas. Eles são localizados sempre onde as atividades não críticas desembocam na cadeia crítica.
  - Os ***buffers do recurso – BR*** se comportam de forma diferente dos anteriores, sua presença no programa não implica em um aumento de tempo de duração do projeto. Eles cumprem uma função meramente informativa. Sua localização é junto aos recursos que trabalham nas

atividades críticas, sempre e quando a atividade crítica anterior seja executada por outro recurso. A função dos BR é a de informar ao recurso o nível de avanço da atividade anterior, dando-lhe um horizonte de tempo para se preparar e dar início a sua atividade o mais rápido possível.

O dimensionamento dos *buffers* é calculado para refletir a incerteza presente na estimação das durações das atividades e do projeto, protegendo-o das variações comuns ou naturais do sistema. Existem dois métodos de dimensionamento formalmente apresentados na metodologia: "O método de cortar e colar" e "O método da raiz quadrada do erro". (TUKEL *et al*., 2006)

- O ***método de cortar e colar*** determina o tamanho dos *buffers* supondo que as estimações das atividades são feitas com uma probabilidade de atraso de 50%. O *buffer* é resultante da metade da soma de todas as contingências cortadas na estimação das durações das atividades que o alimentam. Kjersti & Taylor (1999), Herroelen *et al.* (2002), Raz *et al.* (2003) e Jyh-Bin (2007) advertem a tendência de crescimento linear que o *buffer* tem em relação ao comprimento da cadeia que o alimenta. Os autores indicam também como esse prazo excessivamente dilatado poderia significar a inserção de proteção desnecessária que reduza a competitividade dos projetos em comparação com empresas concorrentes.
- O ***método da raiz quadrada do erro*** usa duas estimativas da duração das atividades para determinar o tamanho dos *buffers*. A primeira é o "tempo seco" ou estimação da duração com 50% de probabilidade de atrasar ($\mathbf{A_i}$). A segunda estimativa inclui uma proteção suficiente para ser considera como duração de baixo risco (*low-risk*), esta é geralmente estimada com 90% de probabilidade de não atrasar ($\mathbf{S_i}$). A incerteza de cada atividade é calculada como a diferença de ambas estimativas ($\mathbf{S_i - A_i}$). Newbold (1998) postula que a duração das atividades tem uma distribuição lognormal e determina que a incerteza calculada equivale a dois desvios padrões. Com essa mesma lógica, o

autor estabelece que para cada atividade, o desvio padrão é aproximadamente igual à metade da incerteza calculada (**$(S_i - A_i)/2$**).

Para poder calcular o tamanho do buffer que possa cobrir a incerteza total da cadeia que o alimenta, o autor, fazendo uso do "teorema do limite central", postula que a soma das distribuições das atividades é representada por uma distribuição normal. Desta maneira, se estabelece que o tamanho do buffer equivale a dois desvios padrões, calculado da seguinte forma. Esse método de dimensionamento não pressupõe incertezas não correlacionadas.

$$\sigma = \sqrt{\left(\frac{S_1 - A_1}{2}\right)^2 + \left(\frac{S_2 - A_2}{2}\right)^2 + \ldots + \left(\frac{S_n - A_n}{2}\right)^2}$$

Existem outros métodos de dimensionamento de *buffers* propostos na literatura, mas, que ainda não têm sido aceitos formalmente pela metodologia. Tais propostas podem ser encontradas no artigo Tukel *et al.* (2006) onde os métodos de dimensionamento de *buffers* levam em conta as características de complexidade e do nível de compartilhamento de recursos dos projetos. Outras abordagens aplicam a teoria de filas e algumas ferramentas de simulação.

Alem da função de proteção, os *buffers* têm a finalidade de controlar o andamento do projeto, motivo pelo qual é preciso determinar pontos de gerenciamento específicos para o estado de consumo dos *buffer*s (Figura 3.4). Ao atingir o primeiro ponto de gerenciamento a metodologia recomenda prestar atenção no estado do programa e planejar ações futuras. Essas ações serão aplicadas, só, se o consumo do *buffer* atingir o segundo ponto de gerenciamento.

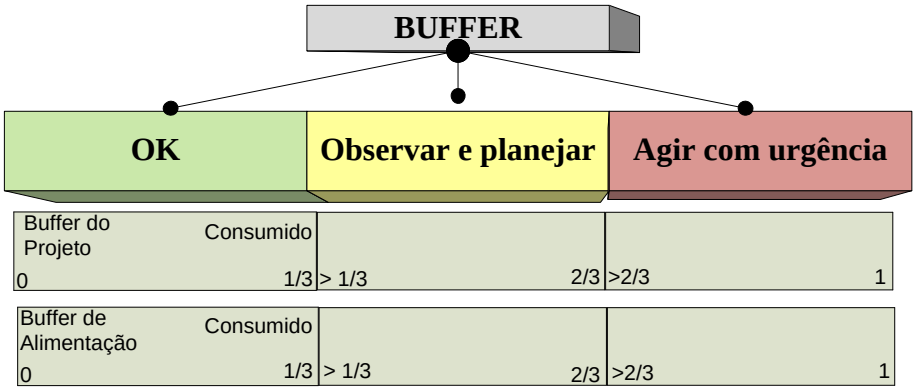

Figura 3.4: Exemplo clássico de gerenciamento de *buffers.*

Fonte: Barcaui & Quelhas (jul. 2004)

- **Mentalidade do papa-léguas (*Road-runner*).** - A idéia dessa mentalidade é permitir que o programa aproveite os ganhos de tempo conseguidos na cadeia crítica. Segundo Leach (2005), o autor da metodologia (Goldratt), define esse conceito como a regra de ouro da subordinação: *"Sempre que uma tarefa seja completada na cadeia crítica (e nas cadeias não críticas, sempre que a decisão não afete a cadeia crítica) as seguintes tarefas na seqüencia devem ser iniciadas o mais rápido possível"*. Quer dizer, que as atividades devem ser executadas o mais rápido possível, é claro, priorizando a cadeia crítica. Segundo Rand (2000). Herroelen *et al.* (2002) e Raz *et al.* (2003) a implementação dessa mentalidade implica que, além do programa normal (*schedule*) obtido com base nos tempos mais tardios, teria que se estruturar um segundo programa baseado nas atividades programadas com os tempos mais cedo possível (mentalidade do papa-léguas). Este programa, chamado como "programa projetado", reflete as atualizações feitas durante a etapa de execução.

3.1.2.2. O programa básico e o programa projetado

A combinação dos elementos acima descritos gera dois tipos de programas, que usadas simultaneamente, permitem focalizar, explodir e elevar a restrição do sistema fazendo com que os projetos sejam terminados dentro do tempo especificado (Figura 3.5). O **programa de base** ou programa básico é aquele que

estrutura a seqüência das atividades utilizando os tempos de início mais tarde e resolvendo os conflitos de compartilhamento de recursos. Nele também são inseridos os três tipos de *buffers* que determinam a duração total do projeto protegido contra as variâncias do sistema. O **programa projetado** é aquele que resulta da aplicação da mentalidade do papa-léguas, onde as atividades são programadas para serem iniciadas o mais rápido possível. Esse programa ao contrário do anterior, não leva em conta os *buffers* e é constantemente atualizado no acompanhamento da execução do projeto. Ambos os programas interagem entre si com a finalidade de controlar o andamento do projeto. O programa básico fica constante na sua estrutura, registrando as mudanças do programa projetado. Este informa a quantidade de trabalho executado, o tempo faltante para o término de cada atividade e o consumo dos *buffers*. Simultaneamente, o programa projetado recolhe essa informação e determina de forma dinâmica as atividades que podem ser executadas no futuro. Isso, subordinando a programação à cadeia crítica (restrição) do sistema.

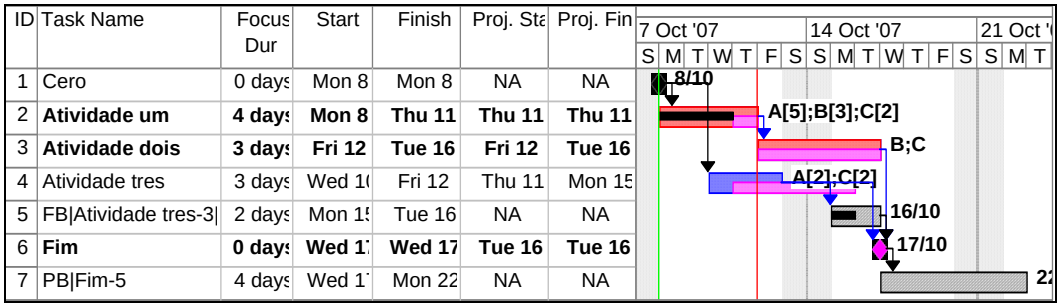

Figura 3.5: O *Programa base* e o *programa projetado*

As barras vermelhas (atividades críticas), barras azuis (atividades não críticas) e barras hachuradas (*buffers)* fazem parte do programa de base. As barras roxas fazem parte do programa projetado. As barras pretas indicam o trabalho executado nas atividades ou o consumo dos *buffers.*
Repare-se na primeira atividade como o programa projetado indica o trabalho a fazer segundo a informação do programa base. O consumo dos *buffers* é calculado em base ao programa projetado, na terceira atividade a barra roxa excede a linha vermelha indicando que essa atividade esta atrasada, o *buffer* é consumido na mesma quantidade.

3.1.2.3. Processo lógico de programação da cadeia crítica

A metodologia se inicia com a redefinição da duração de cada atividade, em forma determinística e sem folgas na estimação. A rede é construída levando em conta as restrições de precedência e de utilização de recursos. Os tempos de início

(*start time*) de cada atividade são programados usando a regra de “início na data mais tarde possível”. Com isso feito, identifica-se a **cadeia crítica** que é a seqüência de atividades que determina a duração do projeto. Nessa cadeia se insere o ***buffer* do projeto,** determinando sua data final. Posteriormente são inseridos os ***buffers* de alimentação**, essa inserção faz com que as cadeias não críticas que o alimentam, sejam empurradas no tempo (ou seja, para o lado esquerdo do programa). Muitas vezes a inserção dos BA faz com que o adiantamento do início das cadeias não críticas provoque novos conflitos de recursos. Em algumas ferramentas de *software* esta situação é definida como falta de espaço e é “solucionada” com a inserção de BA com certo grau de consumo. A segurança perdida nesta operação é recuperada no BP, quer dizer, que a quantidade de tempo que corresponderia aos BA é aumentada no BP. (HERROELEN & LEUS 2001; HERROELEN *et. al.* , 2002).

Finalmente, sempre que for necessário, incluem-se os ***buffers* de recursos** dentro da cadeia crítica. Algumas das ferramentas de *software* podem obter a informação oferecida pelos BR por meio de consultas predefinidas. Nesses casos pode se dispensar sua inserção, já que resulta uma complicação a mais na leitura dos diagramas de *Gantt* dos projetos. (Figura 3.6).

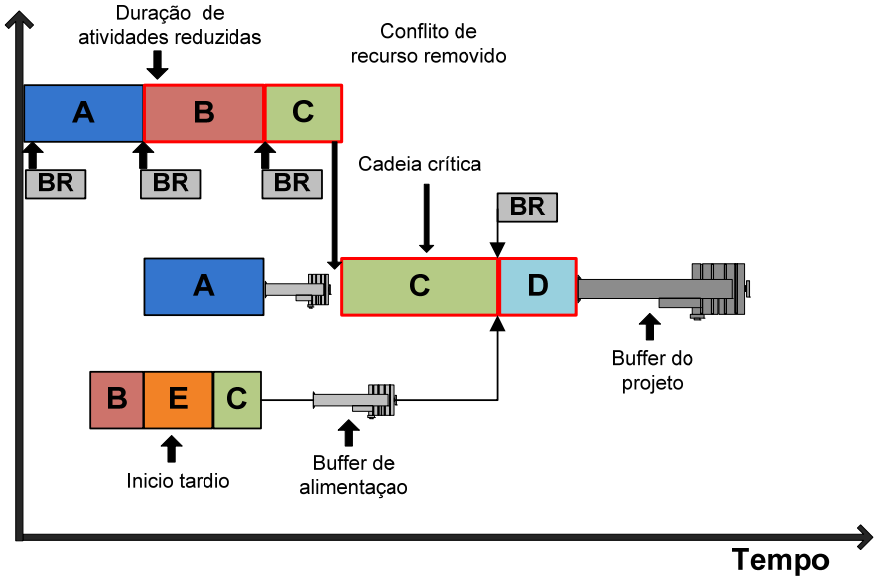

Figura 3.6: Principais elementos da metodologia CC/BM
Fonte: Leach (2005)
Os retângulos de diferentes cores indicam o tipo de recurso utilizado na execução das atividades, os retângulos com nomenclatura BR indicam os *buffer* de recurso utilizados. As atividades que formam parte da cadeia crítica são ressaltadas com um contorno vermelho e seguidas pelo BP. Existe uma restrição de capacidade no recurso C que faz com que a segunda atividade realizada por este seja deslocada. A ultima seqüência de atividades BEC mostram a subordinação da cadeia não critica à cadeia crítica, onde as atividades foram programadas com as datas de inicio tardio (*late start*) e se inseriu um *buffer* de alimentação

Uma vez definido o programa básico se inicia a execução do projeto. O acompanhamento da execução é realizado com a interação dos programas básicos e projetado. Segundo Newbold (1998) o critério de decisão na execução das atividades será priorizando as atividades críticas, em segundo lugar se executarão as atividades que provocariam um maior consumo dos *buffers* dos projetos e finalmente aquelas que provocariam um maior consumo dos *buffers* de alimentação.

3.1.3.
CC/BM em múltiplos projetos

A metodologia se estende também na programação de múltiplos projetos simultâneos que interdependem entre si pelo compartilhamento de um ou vários recursos. O sistema é composto por vários projetos que, do ponto de vista de restrições de seqüenciamento e de objetivos, são independentes entre si. O objetivo do sistema é maximizar o número de projetos executados segundo a

prioridade definida para cada projeto, mantendo o sistema em equilíbrio mediante o nivelamento da carga de trabalho entre os projetos. (LEACH, 2005).

Similarmente à programação de projetos individuais, a metodologia insere *buffers* de tempo localizados estrategicamente, procurando sincronizar a execução dos projetos, garantindo a continuidade do fluxo de projetos no sistema e o cumprimento das datas de término prometidas.

3.1.3.1. Principais Elementos

Ao se tratar de projetos simultâneos, alem dos elementos usuais da CC/BM alguns elementos adicionais são utilizados:

- **Projetos individuais corretamente programados**. - A programação dos múltiplos projetos parte dos projetos individuais corretamente programados de acordo com a metodologia CC/BM. Ao fazer esta primeira etapa, se considera que cada projeto individual conta com toda a disponibilidade de recursos do sistema. Desta maneira se assegura que cada projeto seja protegido contra as variâncias comuns ou naturais de suas próprias atividades.
- **A seqüência crítica**. - Uma vez que os projetos individuais são inseridos de forma prioritária (segundo o critério da empresa), pode-se reconhecer o recurso que representa a maior restrição para manter a continuidade do fluxo de projetos, isto é, o gargalo do sistema. Os proponentes da metodologia explicam que não se originam grandes complicações se esta eleição não é precisamente a correta. De qualquer maneira ao se aplicar os passos da “teoria das restrições” se revelará, eventualmente, o verdadeiro gargalo. Geralmente os critérios para a escolha do gargalo são baseados na demanda que o sistema tem pelo recurso, também pode ser baseada no custo financeiro que implica sua manutenção.

  Analogicamente à programação de projetos individuais, a metodologia procura reunir as restrições do sistema dentro de uma seqüência de atividades (como era na cadeia crítica). No caso de múltiplos projetos, esta é chamada de **seqüência crítica** e está formada por todas as

atividades de todos os projetos que são trabalhadas pelo recurso definido como gargalo.

- ***Buffer* de Capacidade – BC. -** Este buffer é definido em função do recurso gargalo (restrição do sistema) e tem a finalidade de evitar que, numa seqüência de atividades de diferentes projetos, um atraso numa atividade de um projeto venha atrasar outras atividades de outros projetos que a seguem. A meta do BC é inserir um grau de isolamento entre os projetos dos sistemas, fazendo com que a interdependência entre projetos, gerada pelo compartilhamento de recursos, seja menor. O BC permite também, manter a organização e equilíbrio dentro do sistema, evitando que a execução de atividades siga um comportamento reativo. Ou seja, indo de um projeto a outro tentando solucionar estados de emergência.

  A localização dos BC é entre dois projetos consecutivos, especificamente, entre a última atividade do primeiro projeto e a primeira atividade do segundo, sendo que ambas as atividades são executadas pelo mesmo recurso gargalo. Nas ferramentas de *software* não se utiliza uma representação gráfica para o BC, esse é simplesmente um espaço inserido entre as atividades dos projetos.

  O dimensionamento dos BC é baseado na capacidade do recurso gargalo, autores como Leach (2005) estabelecem como na prática é razoável que este seja determinado como 25% da capacidade do recurso. Outros autores como Newbold (1998) o determinam como uma porcentagem complementar à capacidade do recurso, quer dizer, se o desejado é reservar 25% da capacidade, se considerara que a carga atual do recurso corresponderá a 75% de capacidade. Sustentações matemáticas ou estatísticas que expliquem essas recomendações não foram encontradas na literatura.

- ***Buffer* Tambor – BT.-** Este *buffer* tem a função de proteger as atividades executadas pelo recurso estratégico de atrasos que possam acontecer nas atividades executadas por recursos não estratégicos. Além da função protetora, o BT assegura que o recurso estratégico se mantenha

> sempre ocupado, garantindo que seu fluxo de trabalho seja contínuo. A definição dos BT é similar à dos BA, estes ocupam um espaço na estrutura do programa e sua inserção implica empurrar para mais cedo no tempo (para a esquerda) a cadeia que o alimenta. Estudos sobre o comportamento deste tipo de *buffer* não foram achados na literatura pesquisada. Similarmente, na ferramenta de *software* utilizada, a inserção dos BT não é uma aplicação estritamente necessária como é o caso do seu análogo em projetos individuais. (*buffer* de alimentação – BA*).*

Na programação de múltiplos projetos persiste a utilização do "programa base" e do "programa projetado".

3.1.3.2. Processo lógico de programação da seqüência crítica (múltiplos projetos)

A metodologia se inicia com a programação de cada projeto individual segundo as recomendações da CC/BM. Os projetos são ordenados de acordo à prioridade outorgada. Essa prioridade é utilizada também como critério de decisão na escolha dos primeiros projetos a programar dentro do sistema de múltiplos projetos. Segundo Newbold (1998) tais prioridades devem se manter fixas durante todo o processo de desenvolvimento dos projetos, recomendando que sejam alteradas unicamente em casos extremos de perda de controle. Não obstante, autores como Cohen *et al.* (2004) consideram que a prioridade na escolha do próximo projeto (ou atividade) a programar seja feita com base do estado dos *buffers* do projeto- BP, priorizando aquela atividade que pertença ao projeto com o *buffer* mais comprometido.

Uma vez que algum tipo de critério é estabelecido na priorização dos projetos, eles são inseridos num programa especial que alberga o conjunto de múltiplos projetos. Diferentes abordagens, as que são explicadas no capítulo IV, são utilizadas na definição estrutural desse programa especial.

O seguinte passo na programação é a identificação do(s) recurso(s) que restringem a saídas do sistema (a definição da quantidade de restrições do sistema varia com a abordagem utilizada). Com base nessa restrição, no processo de nivelamento do sistema, serão determinadas as datas de início de cada projeto. A finalidade do nivelamento é equilibrar a carga de trabalho no sistema de tal

maneira que esta seja constante. Caso exista um conflito de recursos no sistema, os projetos de menor prioridade serão deslocados no tempo, atrasando sua data de início, mas, sem alterar sua programação interna. Uma vez que os projetos estão corretamente equilibrados, pode-se identificar a "**seqüência crítica"** de atividades (entre todos os projetos) que serão executadas pelo(s) recurso(s) gargalo.

Em base à seqüência crítica se determina o tamanho e localização dos *buffers de* tempo que irão a proteger o sistema. Primeiramente se inserem os *buffers* de capacidade - BC, ao aumentar o espaçamento entre os projetos se reduz a dependência, pelo compartilhamento de recursos, entre eles. Finalmente se inserem os *buffers* tambor – BT que similarmente aos BA provocam adiantamentos na data de início das cadeias que o alimentam. Da mesma maneira do que com os BA, existem casos em que não se conta com o espaço necessário para inserir BT do tamanho recomendado, infelizmente, não foram achados métodos alternativos para recuperar a proteção que seria perdida na inserção de *buffers* de menor dimensão. Herroelen *et al.* (2002), defendem que tanto para BA quanto para BT é necessário a aplicação de algoritmos especiais de reprogramação, não obstante, esse tema ainda não é muito estudado na literatura.

O acompanhamento do programa de múltiplos projetos também é realizado com a interação do programas básico com o programa projetado de cada projeto. A ferramenta de *software* utilizada atualiza o programa projetado de acordo com as prioridades (fixas) definidas inicialmente para cada projeto. Cohen *et al.* (2004) atualizam o programa projetado (no seu experimento) considerando prioridades dinâmicas em base à "seqüência crítica" e às atividades dos projetos de maior consumo de BP e BA.

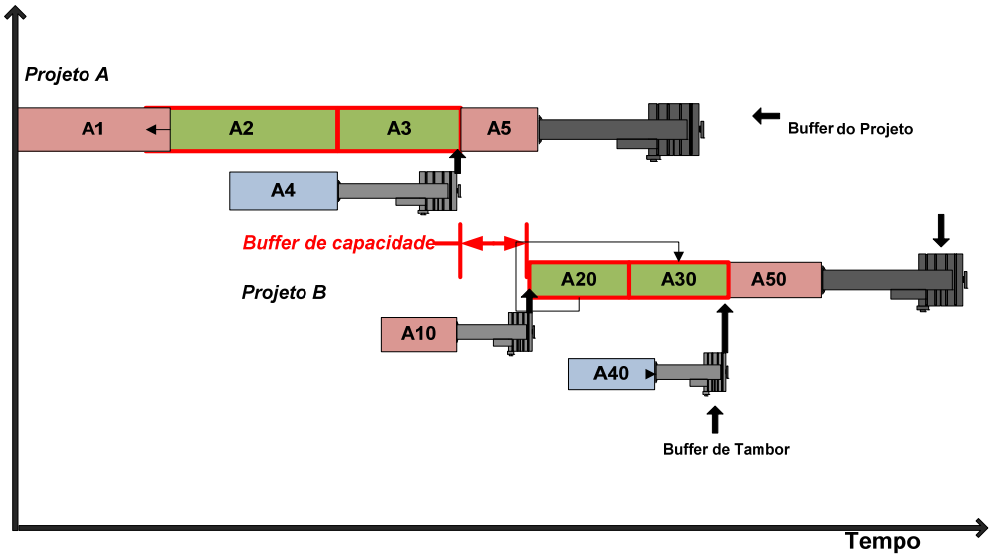

Figura 3.7: Principais elementos da metodologia CC/BM em múltiplos projetos.
Fonte: Leach (2005)

Três diferentes recursos são utilizados na execução de dois projetos simultâneos, estes são diferenciados pela cor do retângulo (rosa, verde e azul), sendo que o recurso verde é definido como “gargalo” do sistema e suas atividades formam a “seqüência crítica”. Ademais dos BP de cada projeto, insere-se o BC entre as atividades executadas pelo mesmo recurso, mas de diferentes projetos, esse é representado por um espaço delimitado por linhas vermelhas. Os BT são inseridos nas cadeias que desembocam na seqüência crítica, empurrando-las à esquerda.

4
Metodologia da pesquisa - Efeito das abordagens

Neste capítulo se estudam as diferentes abordagens propostas por Newbold (1998) para a aplicação da metodologia da cadeia crítica e gestão de buffers de tempo – *CC/BM* – no ambiente de múltiplos projetos. A análise das características, vantagens e desvantagens de cada abordagem são realizadas através de um processo de comparação. A comparação é estruturada pelo "problema-exemplo" escolhido na literatura pesquisada e pelos diferentes "casos de estudo teóricos" criados a partir dele.

A finalidade de criar diferentes variações do mesmo problema é a geração de situações extremas que testem o desempenho dos principais elementos da metodologia dentro do cenário da abordagem utilizada. Essas variações são realizadas aproveitando a facilidade de análise do problema original e sem perder as características de interdependência de recursos que ele possui. A implementação dos problemas estudados é feita com a utilização da ferramenta Prochain - Pipeline da empresa Prochain Solutions Inc (*add-in* do Microsoft Project).

Essa análise preliminar permite, também, estabelecer uma base necessária para o entendimento das funções e efeitos que a implementação dos *buffers* de capacidade – BC tem sobre o sistema. Similarmente, este estudo fornece o alicerce para o entendimento das conseqüências de programação originadas pelas regras de priorização usadas na metodologia. O desenvolvimento desses estudos é feito no capítulo V.

As definições dos gráficos utilizados nesta seção são apresentadas no apêndice A.

4.1.
O problema-exemplo

4.1.1.
O Problema original

O problema-exemplo, proposto por Pritsker B. *et al.* (1969.), foi escolhido da literatura pesquisada por ser um sistema simples de múltiplos projetos. A estrutura da rede é constituída por três projetos com um pequeno número de atividades e relacionamentos do tipo *finish-start* (*i.e.* que a atividade só pode ser iniciada quando sua antecessora tenha terminado). Não obstante a simplicidade dos projetos, eles apresentam, de um ponto de vista global de múltiplos projetos, uma interdependência importante causada pelo compartilhamento de recursos limitados.

A Figura 4.1 mostra a programação ótima do sistema, a qual foi obtida mediante uma abordagem de programação inteira. Contra esse programa são contrastados os resultados do estudo. Contudo, é importante levar em consideração que o problema foi desenhado com base nos conceitos das metodologias PERT/CPM, pelo que se pressupõe que as durações de cada atividade levam uma margem de tempo de segurança embutida. Na aplicação da metodologia CC/BM se utilizaram as durações das atividades correspondentes à mediana (probabilidades de não exceder e de exceder iguais a 50%), ou seja, sem qualquer segurança embutida. Com essa lógica, e, para não diminuir as durações originais a ponto de perderem significância, se considera que o programa ótimo usado como referência comparativa corresponde ao dobro das durações originais, tal como apresentado na Figura 4.2

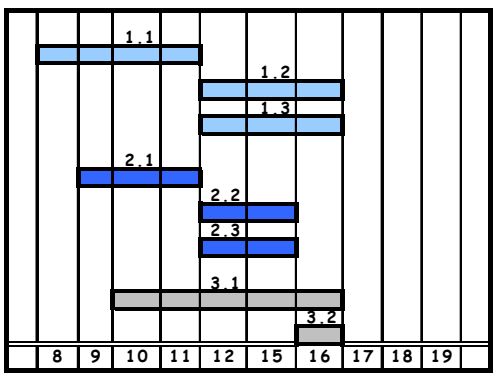

Figura 4.1: Programa ótimo original

Fonte: Pritsker B. *et al.* (1969.)

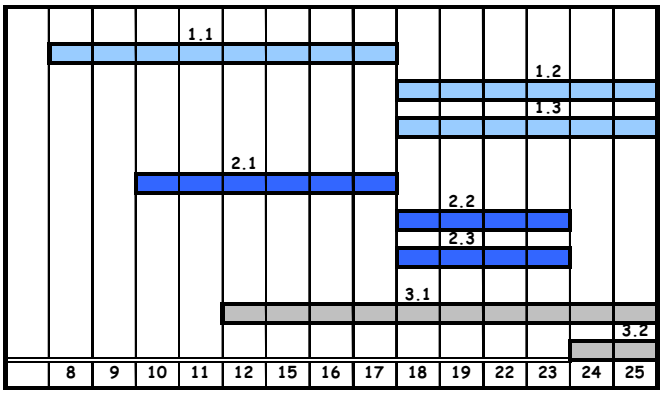

Figura 4.2: Programa ótimo modificado

Para a implementação do problema foi utilizada como data de início o dia 8/10, sendo que os projetos dois e três têm datas de chegada estabelecidas no dia 9 e 10 de outubro, respectivamente. A ferramenta de software utilizada na implementação respeita, sempre que sejam viáveis, as datas de início especificadas. Entretanto, as datas de entrega podem não ser respeitadas pela programação. Os dados da rede se encontram na Tabela 4.1.

Tabela 4.1: Dados das redes dos múltiplos projetos
Fonte: Fonte: Pritsker B. *et al.* (1969.)

Projeto	Atividade	Relação de Precedência	Chegada	Duração	Restrição de data de entrega	Recursos		
						A	B	C
1	1	Nenhuma	1 (8/10)	4	8 (17/10)	5	3	2
1	2	(1,1)	1 (8/10)	3	8 (17/10)	0	1	1
1	3	Nenhuma	1 (8/10)	3	8 (17/10)	2	0	2
2	1	Nenhuma	2 (9/10)	3	9 (18/10)	1	1	1
2	2	Nenhuma	2 (9/10)	2	9 (18/10)	2	0	0
2	3	(2,1)	2 (9/10)	2	9 (18/10)	2	2	0
3	1	Nenhuma	3 (10/10)	5	9 (18/10)	2	1	1
3	2	Nenhuma	3 (10/10)	1	9 (18/10)	1	3	0
Disponibilidade de Recursos						**8**	**5**	**4**

Para melhor entendimento do problema foram adicionadas duas atividades-fantasma (*dummy activities*) de duração zero, elas representam o início e fim de cada projeto. A representação das redes é definida na Figura 4.3

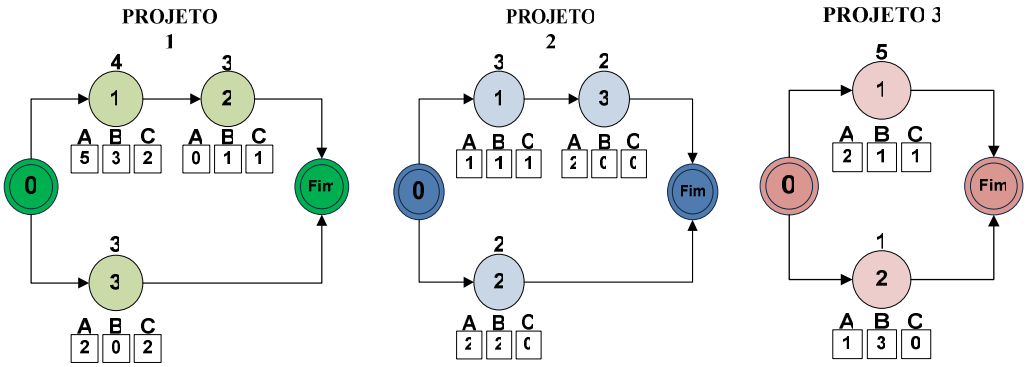

Figura 4.3: Representação gráfica das redes dos projetos.

As setas indicam os relacionamentos de precedência entre atividades, os números sobre os círculos são as durações das atividades e os números enquadrados correspondem às quantidades dos recursos que a atividade necessita para ser executada.

4.1.2. Casos de estudos teóricos: Variação do problema original

A variação do problema original foi desenvolvida com a finalidade de testar o comportamento das diferentes abordagens em situações prováveis de acontecer no mundo real. A análise é estruturada em dois casos de estudo, chamados de "teóricos" por serem desenvolvidos a partir de um problema criado para pesquisas teóricas.

Alem de analisar o comportamento das abordagens em condições normais (definidas na continuação), elas também são avaliadas em situações onde a definição das prioridades dos projetos é conflitante com as respectivas datas de chegada. A análise também é feita na etapa de execução, na qual se provocaram atrasos em algumas atividades da rede. Embora tais distúrbios não sejam exatamente iguais entre os casos, procuram ter o mesmo efeito e ser coerentes com a lógica do programa.

Entre os casos de estudo teóricos temos:

- **Caso 1. - Programação Normal**: Utiliza os parâmetros do *software* estabelecidos por padrão, considerando-se que desde o início da programação básica se conhece com certeza a chegada dos três projetos em diferentes datas, vale dizer, o programa contém os três

projetos todo o tempo. Respeitam-se as prioridades definidas originalmente, considerando que o projeto mais prioritário é o primeiro em carteira. O grau de importância diminui de acordo com a ordem de chegada.

- **Caso 2. - Programação Prioritária:** Difere do caso anterior no ordenamento prioritário dos projetos. Neste caso, a prioridade arbitrariamente estabelecida é preemptiva em relação à prioridade de ordem de chegada. Igualmente trabalha-se o tempo todo com os três projetos.

4.2. As abordagens de aplicação

Segundo Newbold (1998) a programação de múltiplos projetos por meio da CC/BM pode ser implementada seguindo três tipos de abordagens.

4.2.1. Abordagem de "Todos juntos"

Programa todas as atividades dos projetos em forma simultânea dentro de um único projeto criado artificialmente. Esse projeto artificial tem uma única data final que representa o término de todos os projetos. No programa artificial não se distingue explicitamente cada projeto, sendo que a programação das atividades é feita segundo o algoritmo da metodologia para projetos individuais. Isso implica a inserção de um ou mais *BA* e um único *BP*. Para facilitar a análise da programação decidiu-se não incluir *BR*, tornando o programa mais compacto e simples de analisar.

4.2.2. Abordagem de "Projetos sucessivos"

Essa abordagem parte dos projetos individuais corretamente programados segundo as recomendações da metodologia. Quer dizer que, supondo que existe uma disponibilidade total de recursos, em cada projeto se resolvem os conflitos de recursos, se define a cadeia crítica e se inserem os *buffers*.

O sistema que gerencia a programação desses projetos individuais é chamado de **Master**, este é definido como um gerenciador de projetos simultâneos que além de permitir a programação e o acompanhamento individual de cada projeto, equilibra a carga de trabalho do sistema e gera informações sobre o estado do recurso gargalo. A interação dessas funções faz com que o objetivo de manter equilibrado o fluxo de projetos, atingindo as datas prometidas seja mais factível.

A abordagem estabelece que o nivelamento de carga de trabalho do sistema é através de todos os recursos, ou seja, que não é permitido sobrecarga em qualquer recurso. Segundo Newbold (1998) à medida que o número de projetos simultâneos aumenta e o grau de complexidade, em função das dependências de recursos, se eleva, a eficiência do programa irá se deteriorando, podendo ser impossível atingir o nivelamento adequado de todos os recursos.

4.2.3.
Abordagem do "Recurso estratégico ou gargalo"

Essa abordagem tem a mesma concepção da aproximação anterior, também é desenvolvida no programa máster. Contudo, esta última se diferencia por resolver os conflitos de recursos entre projetos unicamente para o recurso definido como gargalo do sistema, permitindo-se a existência de sobrecargas. A abordagem supõe que ao resolver os conflitos do gargalo se criam espaços no programa que diminuem os conflitos dos outros recursos. Quanto a isso, Newbold (1998) adverte que são necessários procedimentos alternativos de gestão e programação que mitiguem o efeito desses conflitos.

4.3.
Implementação dos casos de estudo

O caso da programação normal é aplicado nas três abordagens. Já o caso de programação prioritária é aplicado unicamente na abordagem de "projetos sucessivos". A aplicação da programação prioritária tem características similares nas abordagens dos "projetos sucessivos" e do "recurso gargalo", razão pela qual só é desenvolvida na primeira delas. Pelas características de projeto individual que

a abordagem de "todos juntos" tem, não fazem sentido as variações do tipo de programação prioritária.

4.3.1. Abordagem de "Todos juntos"

4.3.1.1. Programação normal

Nessa abordagem, a implementação do programa base é relativamente simples e fácil de estruturar, não obstante, tal facilidade origina um efeito contrário na correta interpretação e consecução de objetivos individuais (de cada projeto) e globais (do sistema).

A estrutura do programa "fictício" contém todas as atividades dos três projetos, incluindo as "*dummy*" ou "*milestone*" de início e fim. A cadeia crítica, que não chega a ser representativa dos três projetos, é formada unicamente pelas duas primeiras atividades do projeto um e pelo *milestone* de término do projeto artificial. Isso faz com que se insiram um *buffer* de projeto- *BP* e cinco *buffers* de alimentação – *BA*. Estes, ao não ter espaço suficiente no programa, são inseridos com algum grau de consumação, que é compensado aumentando-se o tamanho do BP. Esse procedimento, aplicado no caso de projetos individuais, é eficiente para assegurar a proteção do projeto, evitando a origem de novos conflitos de recursos.

Contudo, sua aplicação em múltiplos projetos não é bem sucedida pela distorção das datas de término individuais, as quais são igualadas a uma mesma data. (25/10 – superior ao ótimo do projeto dois do artigo original, 23/10).

Usando a ferramenta *"toggle deadline"* (setas vermelhas), que permite visualizar a porção do BP que corresponderia a cada atividade, pode definir-se a segurança que as atividades dos projetos precisam. Isto é, pode-se definir a data de entrega para cada projeto.

O efeito negativo da não representatividade da estrutura da cadeia crítica se estende à etapa da execução, na qual se originam discrepâncias entre os relatórios de acompanhamento do projeto e o estado real do consumo do BP. Isso pode ser observado na Figura 4.4 onde o relatório da coluna 7 (*check list*) indica que as primeiras atividades de cada projeto têm maior prioridade de programação.

Contudo, ao não pertencerem à cadeia crítica, é muito provável que elas não sejam consideradas nas decisões de execução.

Essas discrepâncias se acentuam na presença de atrasos na execução do programa, no qual, dar prioridade de programação às atividades críticas (o que é correto segundo a metodologia), nem sempre assegura o bom desempenho do projeto. Pior ainda, pode ocasionar atrasos maiores dos projetos. Veja bem na Figura 4.5, como ao atrasar as primeiras atividades (não criticas) dos projetos dois e três ocasiona que o programa projetado para o dia 15/10 (*Monday*) priorize a programação da segunda atividade crítica do projeto um, fazendo com que as atividades do projeto dois sejam programadas três dias depois (18/10-*Thursday* e 19/10-*Friday*) e que o BP seja altamente consumindo.

Mesmo que essa operação gere um consumo do BP importante, ele não compromete a data final do programa fictício. Não obstante, o projeto dois se vê muito danificado em relação a seu programa base (determinado pelo termino marcado pelas setas vermelhas).

Contrariamente ao esperado, gerar esse mesmo atraso numa das atividades críticas (Figura 4.6), faz com que o dano no sistema e nos projetos seja muito menor.

Certamente, em outras condições o resultado seria diferente. Não obstante, a análise se interessa em demonstrar que a abordagem não considera a complexidade da interação dos três projetos. Assim ela gera um efeito negativo na estabilidade da rede e na qualidade de informação que ela oferece. As decisões de programação passam de incongruentes e confusas (nas etapas de planejamento e de execução em condições normais), a erráticas (quando o sistema é afetado por atividades atrasadas). Tal deficiência informativa do programa é conseqüência da pouca representatividade que os elementos chave da metodologia para projetos individuais (Cadeia crítica e *buffers* de tempo) têm para o controle de múltiplos de projetos, fazendo com que as vantagens da CC/BM sejam perdidas.

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**4 days**	**0**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**0**	**0**	**0**			
4	1 Atividade tres	3 days	0	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	100	0	0	7-7,2-2;2 Atividade 1	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	3 days	0	0	0			
8	2 Atividade 2	2 days	0	0	0			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	7-7,2-2;2 Atividade 1	2d	2d
10	2 Atividade 3	2 days	0	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	67	0	0	7-7,2-2;2 Atividade 1	5d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	5 days	0	100	0			
14	FB|3 Atividade 1-12|fim-16	3 days	100	0	0	12-11;3 Cero	5d	5d
15	3 Atividade 2	1 day	0	0	0			
16	FB|3 Atividade 2-13|fim-16	1 day	100	0	0	7-7,2-2;2 Atividade 1	1d	1d
17	**fim**	**0 days**	**0**	**100**	**0**			
18	PB|fim-16	7 days	0	0	0	12-11,7-7,2-2;2 Atividade 1	6d	7d

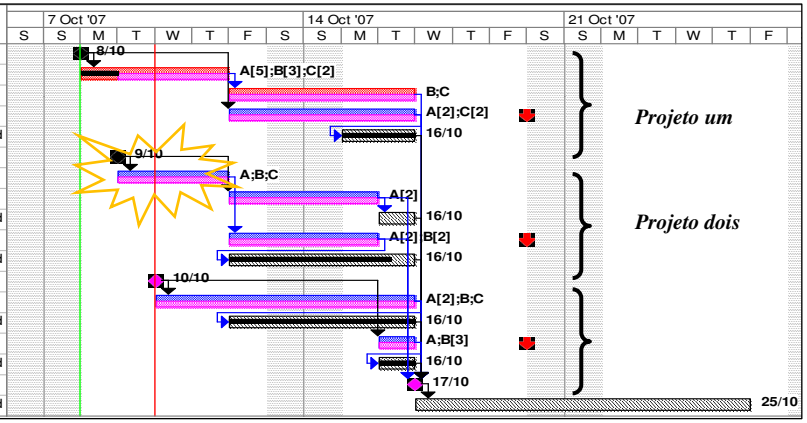

Figura 4.4: Incongruência entre importância de atividades e definição da cadeia crítica. Status ao 9/10

A atividade ressaltada é a atividade que segundo o relatorio *check task* é a atividade que poderia causar maior consumo do BP. A cadeia crítica está formada pelas duas primeiras atividades vermelhas. As setas vermelhas indicam a percentagem do BP que corresponde á atividade, no caso indicaria o que seria o término do projeto com a proteção embutida

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**5 days**	**57**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**57**	**0**	**0**			
4	1 Atividade tres	3 days	57	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	150	0	0	2-2;1 Atividade um	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	5 days	57	0	0			
8	2 Atividade 2	2 days	-14	0	5			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	8-8;2 Atividade 2	2d	2d
10	2 Atividade 3	2 days	57	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	233	0	0	2-2;1 Atividade um	3d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	6 days	29	100	2			
14	FB|3 Atividade 1-12|fim-16	3 days	167	0	0	2-2;1 Atividade um	4d	5d
15	3 Atividade 2	1 day	-14	0	5			
16	FB|3 Atividade 2-13|fim-16	1 day	0	0	0	2-2;1 Atividade um	1d	1d
17	**fim**	**0 days**	**57**	**100**	**0**			
18	PB|fim-16	7 days	57	0	0	2-2;1 Atividade um	7d	7d

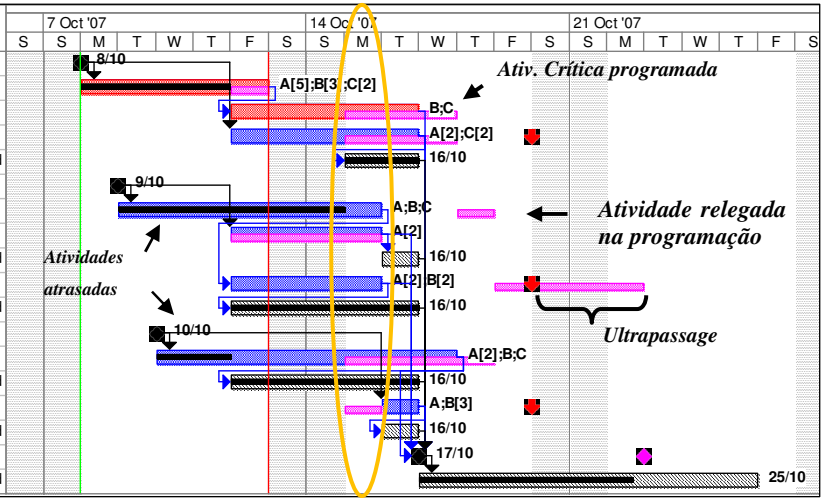

Figura 4.5: Atrasos nos projetos dois e três. Status no dia 12/10

A atividade 1 do projeto 2 (primeira marcada) é atrasada em dois dias. A atividade 1 do projeto 3 é atrasada em um dia (segunda marcada). Repare-se no dia 15/10 (marcado em laranja) como o programa projetado (linhas roxas), ao dar preferência à segunda atividade crítica, relega a atividade 1 do projeto 2 gerando o consumo do BP e a ultrapassagem do que seria sua data de término programada (seta vermelha).

ID	Task Name	Focus Dur	% Impact	eductic Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	1 Cero	0 days	0	0	0			
2	**1 Atividade um**	**6 days**	**29**	**0**	**0**			
3	**1 Atividade dois**	**3 days**	**29**	**0**	**0**			
4	1 Atividade tres	3 days	29	0	0			
5	FB|1 Atividade tres-4|fim-16	2 days	200	0	0	8-8,2-2,7-7;2 Atividade 2	3d	3d
6	2 Cero	0 days	0	0	0			
7	2 Atividade 1	5 days	29	0	0			
8	2 Atividade 2	2 days	29	0	0			
9	FB|2 Atividade 2-8|fim-16	1 day	0	0	0	8-8;2 Atividade 2	2d	2d
10	2 Atividade 3	2 days	29	0	0			
11	FB|2 Atividade 3-9|fim-16	3 days	133	0	0	7-7,8-8,2-2;2 Atividade 2	3d	5d
12	3 Cero	0 days	0	100	0			
13	3 Atividade 1	5 days	29	100	0			
14	FB|3 Atividade 1-12|fim-16	3 days	167	0	0	8-8,2-2,7-7;2 Atividade 2	3d	5d
15	3 Atividade 2	1 day	29	0	0			
16	FB|3 Atividade 2-13|fim-16	1 day	300	0	0	7-7,8-8,2-2;2 Atividade 2	1d	1d
17	**fim**	**0 days**	**29**	**100**	**0**			
18	PB|fim-16	7 days	29	0	0	8-8,2-2,7-7;2 Atividade 2	5d	7d

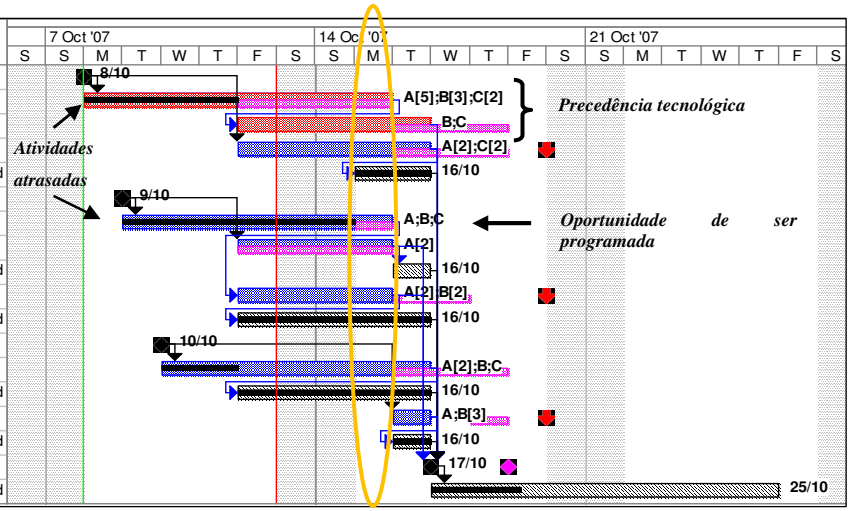

Figura 4.6:Atrasos nos projetos um e dois. Status ao dia 12/10

A atividade 1 do projeto 1 é atrasada um dia. A atividade 1 do projeto 2 é atrasada dois dias. Repare que o atraso no projeto 1 faz com que pelas dependências de precedência, a atividade 2 do projeto 1 (atividade crítica) não possa ser programada no dia 15/10, dando oportunidade de programação à atividade 1 do projeto 2. Isso faz que o BP seja menos consumido e as datas de entrega de projetos (setas vermelhas) não sejam ultrapassadas.

.

4.3.2.
Abordagem de "Projetos sucessivos"

4.3.2.1.
Programação normal

Essa abordagem estabelece um método de programação de múltiplos projetos totalmente diferente da abordagem anterior. A diferença tem origem na utilização do máster como cenário para o desenvolvimento da metodologia CC/BM. Certamente, o fato de que essa abordagem parte de programas individuais corretamente programados, permitindo o nivelamento da carga de trabalho entre eles sem alterar sua solução interna, faz com que alguns aspectos negativos da abordagem anterior sejam eliminados e se atinja um programa mais adequado para a administração de múltiplos projetos simultâneos.

A Figura 4.7 mostra como o máster, ao permitir a presença ativa e explicita das cadeias críticas, os BA, e, os BP dos três projetos, faz com que o sistema considere os objetivos individuais. Entretanto, para a consecução do objetivo global, (estabilidade do sistema) o máster exige que os começos dos projetos dois e três sejam postergados. Este deslocamento mitiga o efeito das interdependências entre os projetos, aumentando o *makespan* ou duração total do programa. Isso poderia ser visto como uma incongruência do programa com os objetivos teóricos da metodologia; não obstante, é importante esclarecer que a preocupação principal da metodologia é definir datas de entrega que sejam realmente factíveis. O objetivo de atingir um programa de múltiplos projetos com o mínimo *makespan* é subordinado à satisfação desse objetivo principal. Repare-se que a data final do conjunto de projetos não é superior ao programa ótimo original. (Figura 4.2)

Contudo, os efeitos originados pelas interdependências constituem um problema ainda mal solucionado. Entende-se por interdependências aos efeitos ocasionados num projeto por distúrbios (atrasos, re-processos, etc.) ocorridos num outro projeto. A origem desses efeitos está no compartilhamento de recursos entre projetos.

Para evitar todo tipo de sobrecargas a ação gerencial é mantida também durante a etapa da execução. Nela o programa projetado é atualizado com base no nivelamento completo (em todos os recursos). Com esta lógica se definem duas formas de realizar o acompanhamento e controle da execução dos projetos: sem alterar o programa de base, mantendo o tamanho original dos *buffers* (obtidos na programação individual); ou reprogramando algum ou todos os projetos do master cada vez que ocorre uma mudança significativa em relação ao programado no sistema. Segundo Herroelen & Leus (2001) é recomendado fazer reprogramações apenas se o BP estiver fora de controle. Já a teoria da CC/BM considera que o programa original foi elaborado para suportar qualquer distúrbio (não excepcional), sendo desnecessário reprogramá-lo (LEACH, 2005).

Certamente, para controlar os efeitos das interdependências, é necessário manter o programa original. A reprogramação do sistema implicaria zerar os *buffers* e recalcular a rede, perdendo a informação do estado dos projetos em relação ao plano original. A informação do consumo dos *buffers* é o único meio que permite inferir que atividade fora do projeto poderia ser a causa de atrasos. Voltando ao exemplo, a Figura 4.7 mostra o programa master com os recursos nivelados. Repare-se como a mesma atualização aplicada na Figura 4.8 origina informações importantes relativas às interdependências na Figura 4.9, na qual pode se observar como o projeto um afeta o consumo dos BP dos projetos dois e três, os que sofrem um consumo de 67% antes mesmo de começar a executar os projetos.

Ao ter os três projetos em evidência, reparou-se que o programa projetado é atualizado segundo as prioridades de projetos definidas na etapa de programação, sem levar em consideração o estado de consumo dos BP, contrariamente às considerações feitas por Cohen *et al.* (2004). No experimento que esses autores fizeram para comparar o desempenho da CC/BM, fazem que a escolha das atividades a executar foi baseada no estado dos *BP*, quer dizer, que as prioridades de programação mudavam dinamicamente com a execução dos projetos. Essa lógica é aplicada na transição do programa da Figura 4.10 ao programa da Figura 4.11, onde ignorando o programa projetado sugerido, executam-se prioritariamente as atividades dos projetos com BP mais consumidos. Essa

decisão não só faz com que os projetos dois e três cumpram com as datas de entrega prometidas, mas também diminui a duração efetiva do projeto. Isso, claro, sem considerar que novos atrasos possam acontecer no sistema. Este resultado deixa a dúvida sobre as razões pelas quais a ferramenta de *software* mantém fixa as prioridades das atividades e se, efetivamente, atualizá-las dinamicamente, tornaria o programa mais eficiente. O esclarecimento destas questões é abordado no Capítulo V.

ID	Task Name	Focus Dur	% Impact	Reductior Impact	Days Slack	Check Task	Chain Left
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**		
1	Cero	0 days	0	0	0		
2	**Atividade um**	**4 days**	**0**	**100**	**0**		
3	**Atividade dois**	**3 days**	**0**	**100**	**0**		
4	Atividade tres	3 days	-100	0	4		
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d
6	**Fim**	**0 days**	**0**	**100**	**0**		
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**		
1	**Cero**	**0 days**	**0**	**100**	**0**		
2	**Atividade 1**	**3 days**	**0**	**100**	**0**		
3	Atividade 2	2 days	-100	0	3		
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	1-1;Cero	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**		
6	**Fim**	**0 days**	**0**	**100**	**0**		
7	PB|Fim-5	3 days	0	0	0	1-1;Cero	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**		
1	**Cero**	**0 days**	**0**	**0**	**0**		
2	**Atividade 1**	**5 days**	**0**	**0**	**0**		
3	Atividade 2	1 day?	0	100	0		
4	FB|Atividade 2-3|Fim-4	1 day	100	0	0	3-3;Atividade 2	1d
5	**Fim**	**0 days**	**0**	**100**	**0**		
6	PB|Fim-4	3 days	0	0	0	3-3,1-1;Cero	5d

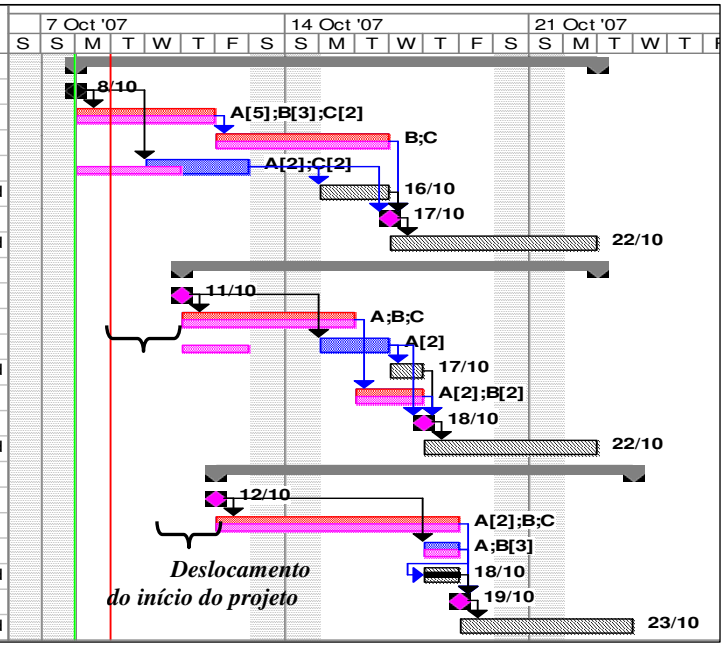

Figura 4.7 : Programa máster nivelando todos os recursos. Status do dia 09/10

No máster cada projeto esta diferenciado pelas linhas cinza escuras que indicam o início de um novo projeto. Cada projeto contém: cadeia crítica (retângulos vermelhos), BA e BP (retângulos hachurados). O máster também indica o *programa de linha de base* (tudo o que não é da cor roxa) e o *programa projetado* (linhas roxas). Os projetos dois e três são deslocados no tempo devido ao nivelamento de todos os recursos definindo um *makespan* de 12 dias (do 8/10 ao 23/10 sem considerar finais de semana).

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**6 days**	**49**	**100**	**0**			
3	**Atividade dois**	**3 days**	**49**	**100**	**0**			
4	Atividade tres	5 days	-25	0	2			
5	FB|Atividade tres-3|Fim-5	2 days	50	0	0	4-3;Atividade tres	2d	3d
6	**Fim**	**0 days**	**49**	**100**	**0**			
7	PB|Fim-5	4 days	49	0	0	2-1;Atividade um	4,98d	7d
2	**Multiproject_Pristsker 2**	**#######**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade 1**	**3 days**	**0**	**100**	**0**			
3	Atividade 2	2 days	-100	0	3			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 3**	**#######**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade 1**	**5 days**	**0**	**0**	**0**			
3	Atividade 2	1 day?	0	100	0			
4	FB|Atividade 2-3|Fim-4	1 day	100	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	3-3,2-2;Atividade 1	5d	5d

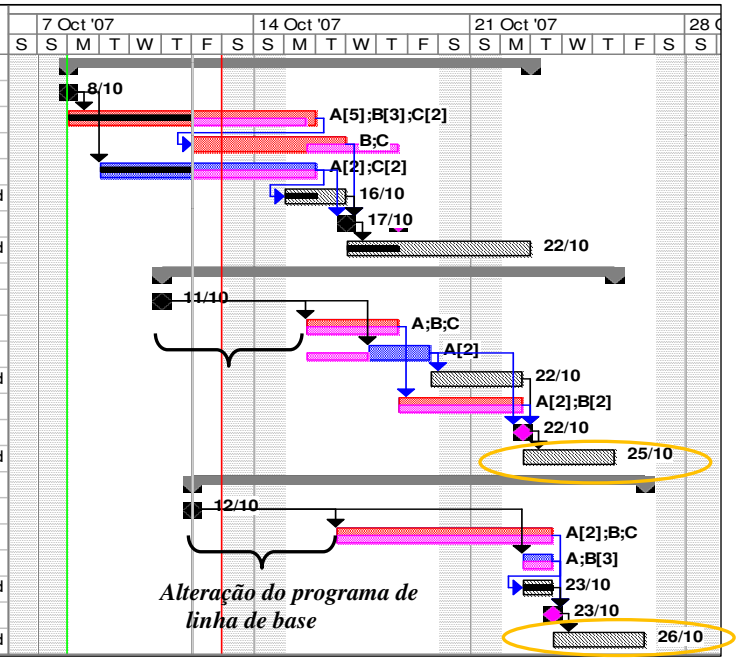

Figura 4.8: Reprogramando o máster. Status do 12/10

As atividades 1 e 3 do projeto 1 são atrasadas em um dia. Ao reprogramar o sistema o *programa de linha de base* é alterado, alterando tanto datas de término (um dia mais tarde - 26/10) como o estado dos BP dos projetos 2 e 3.O programa não da indícios do atraso nem de seu efeito no sistema.

Task Name	Focus Dur	% Impact	Reductior Impact	Days Slack	Check Task	Chain Left	Orig. Chain
Multiproject_Pristsker	11 days	0	0	0			
Cero	0 days	0	0	0			
Atividade um	6 days	49	100	0			
Atividade dois	3 days	49	100	0			
Atividade tres	5 days	-50	0	3			
FB|Atividade tres-3|Fim-	2 days	0	0	0	4-3;Atividade tres	1d	3d
Fim	0 days	49	100	0			
PB|Fim-5	4 days	49	0	0	2-1;Atividade um	4.98d	7d
Multiproject_Pristsker 2	8 days	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	3 days	67	100	0			
Atividade 2	2 days	-33	0	3			
FB|Atividade 2-3|Fim-5	1 day	0	0	0	3-3;Atividade 2	2d	2d
Atividade 3	2 days	67	100	0			
Fim	0 days	67	100	0			
PB|Fim-5	3 days	67	0	0	2-2;Atividade 1	5d	5d
Multiproject_Pristsker 3	8 days?	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	5 days	66	0	0			
Atividade 2	1 day?	67	100	0			
FB|Atividade 2-3|Fim -4	1 day	300	0	0	3-3;Atividade 2	1d	1d
Fim	0 days	67	100	0			
PB|Fim -4	3 days	67	0	0	3-3;Atividade 2	5.02d	5d

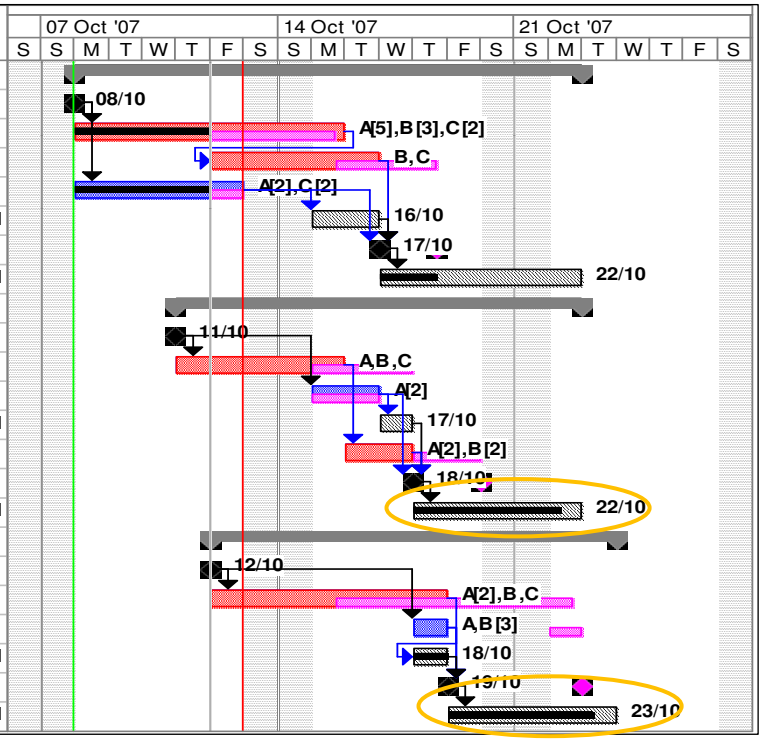

Figura 4.9: Sem reprogramar o máster. Status do 12/10

As atividades 1 e 3 do projeto 1 são atrasadas em um dia. O sistema mantém fixo o *programa de linha de base*, podendo-se observar o efeito do atraso no consumo dos BP dos projetos 2 e 3. Este consumo é gerado pela comparação do *programa projetado,* o qual programa as atividades o mais cedo possível, e o programa estabelecido na etapa de planejamento. Segundo o reporte dos BP o consumo registrado é de 67%, atingindo á área crítica de controle (> 50% de consumo).

Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
Multiproject_Pristsker	11 days	0	0	0			
Cero	0 days	0	0	0			
Atividade um	6 days	51	100	0			
Atividade dois	3 days	51	100	0			
Atividade tres	5 days	-25	0	3			
FB|Atividade tres-3|Fim-	2 days	50	0	0	4-3;Atividade tres	1d	3d
Fim	0 days	51	100	0			
PB|Fim-5	4 days	51	0	0	2-1;Atividade um	4.02d	7d
Multiproject_Pristsker 2	8 days	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	3 days	67	100	0			
Atividade 2	2 days	0	0	2			
FB|Atividade 2-3|Fim-5	1 day	100	0	0	3-3;Atividade 2	2d	2d
Atividade 3	2 days	67	100	0			
Fim	0 days	67	100	0			
PB|Fim-5	3 days	67	0	0	2-2;Atividade 1	4.01d	5d
Multiproject_Pristsker 3	8 days?	0	0	0			
Cero	0 days	0	0	0			
Atividade 1	5 days	34	0	0			
Atividade 2	1 day?	67	100	0			
FB|Atividade 2-3|Fim -4	1 day	301	0	0	3-3;Atividade 2	1d	1d
Fim	0 days	67	100	0			
PB|Fim -4	3 days	67	0	0	3-3;Atividade 2	4.99d	5d

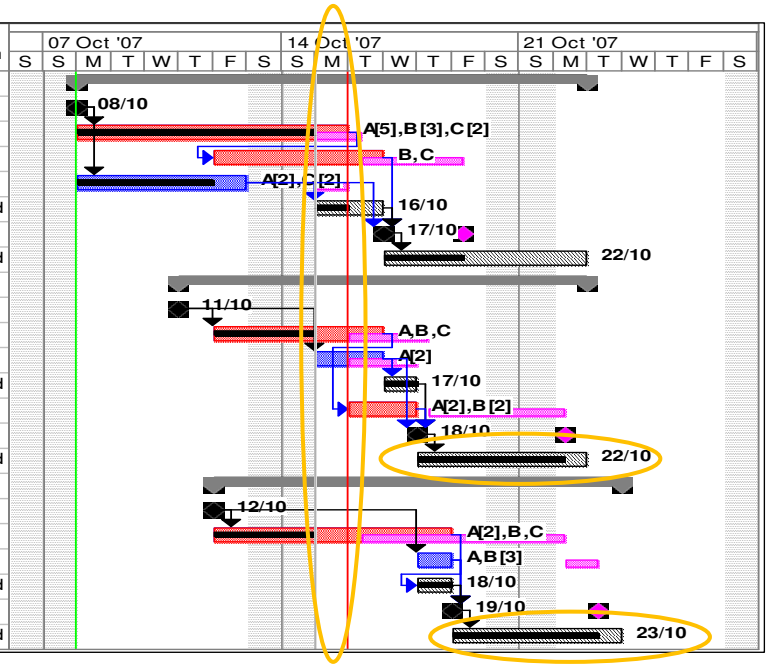

Figura 4.10: Programa projetado sugerido em base a prioridades fixas. Status ao dia 15/10

O programa projetado para o dia 15/10 sugere executar as atividades 1 e 3 do projeto 1. Os BP mostram o consumo que essa execução faria no sistema. Repare-se que esta programação é feita baseada nas prioridades definidas na etapa de planejamento (ordem de projetos 1,2 e 3) e que a disponibilidade de recursos não permite programar mais atividades. (Disp: A=8, B=5,C=4)

Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left
Multiproject_Pristsker	11 days	0	0	0		
Cero	0 days	0	0	0		
Atividade um	6 days	0	100	0		
Atividade dois	3 days	50	100	0		
Atividade tres	5 days	0	0	2		
FB|Atividade tres-3|Fim-	2 days	100	0	0	4-3;Atividade tres	1d
Fim	0 days	50	100	0		
PB|Fim-5	4 days	50	0	0	3-2;Atividade dois	3d
Multiproject_Pristsker 2	8 days	0	0	0		
Cero	0 days	0	0	0		
Atividade 1	3 days	33	100	0		
Atividade 2	2 days	0	0	0		
FB|Atividade 2-3|Fim-5	1 day	100	0	0	3-3;Atividade 2	2d
Atividade 3	2 days	33	100	0		
Fim	0 days	33	100	0		
PB|Fim-5	3 days	33	0	0	2-2;Atividade 1	2.99d
Multiproject_Pristsker 3	8 days?	0	0	0		
Cero	0 days	0	0	0		
Atividade 1	5 days	33	0	0		
Atividade 2	1 day?	33	100	0		
FB|Atividade 2-3|Fim -4	1 day	199	0	0	3-3;Atividade 2	1d
Fim	0 days	33	100	0		
PB|Fim -4	3 days	33	0	0	3-3,2-2;Atividade 1	3d

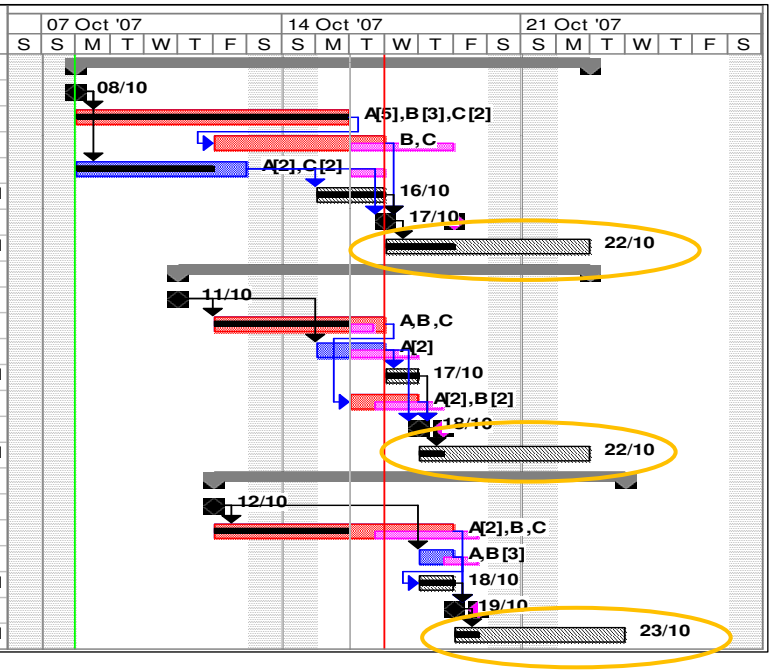

Figura 4.11: Programa resultante da aplicação de prioridades dinâmicas. Status do dia 16/10

O programa projetado para o dia 15/10 da figura anterior foi obviado, levando em consideração o estado dos BP se executaram as atividades que causavam o maior consumo. Isso é: atividade 1 do projeto 2, atividade 1 do projeto 3 e atividade 1 do projeto 1. Repare-se que se deu prioridade às atividades críticas dos projetos com maior consumo de BP, relegando a atividade 3 do projeto 1 por não ser crítica e estar protegida com um BA. O resultado da execução mostra um melhor resultado do que o previsto com a aplicação das prioridades fixas.

4.3.2.2. Programa Prioritário

Esse caso considera que o projeto três, mesmo tendo uma data de início posterior a do projeto dois, possui uma prioridade maior na programação do máster.

Ao inverter as prioridades entre os projetos dois e três se espera que os programas de ambos os projetos, em relação às datas de início, também sejam invertidas. Comparando o programa deste caso (Figura 4.12) com o resultante na programação normal *(*Figura 4.9) corroboramos que o projeto três é iniciado na mesma data do que o projeto dois, quando este tinha maior prioridade. Porém, projeto dois é programado um dia depois (15/10) da data de início do projeto três, quando este ocupava a menor prioridade.

O deslocamento do início do projeto dois é devido à programação antecipada da segunda atividade do projeto três, a qual passa de não ter folga (na programação normal) para ter três dias de folga. Isso faz com que a capacidade disponível dos recursos seja ocupada antecipadamente, provocando o atraso no projeto dois.

Esta programação é efeito da forma como se consideram as prioridades das atividades na etapa do nivelamento do master, no qual se assume que as atividades (críticas e não críticas) têm a mesma prioridade definida para o projeto ao qual pertencem. Desta maneira, acaba dando-se preferência na alocação de recursos às atividades não criticas que pertencem a projetos de maior prioridade.

Esse critério de priorização não considera que as atividades não críticas, pela natureza da metodologia, levam consigo um *buffer* de alimentação*,* ou seja, já estão protegidas para serem relegadas na programação. Ao programar essas atividades antes das atividades críticas geram-se folgas ociosas que implicam o atraso do início dos projetos seguintes.

Note-se na Figura 4.12 como o fato de programar mais tarde a atividade dois do projeto três não afetaria em nada à data de término do projeto, isso devido ao *BA* que protege à cadeia crítica e a data de entrega. Entretanto, tal atraso sim afetaria positivamente o término do projeto dois, devido à antecipação da data de início.

Esta análise identifica o critério utilizado na priorização das atividades que determinam o programa de base do sistema. Tal critério determina a programação considerando unicamente as prioridades relativas definidas para cada projeto, isto é, eliminando o efeito que as cadeias críticas têm sobre o programa. As conseqüências dessa lógica de programação são discutidas no Capítulo V

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-33**	**0**	**1**			
2	**Atividade 1**	**5 days**	**0**	**100**	**0**			
3	Atividade 2	1 day?	-100	0	3			
4	FB|Atividade 2-3|Fim-4	1 day	-200	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-133**	**0**	**4**			
2	**Atividade 1**	**3 days**	**0**	**100**	**0**			
3	Atividade 2	2 days	-133	0	4			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**100**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d

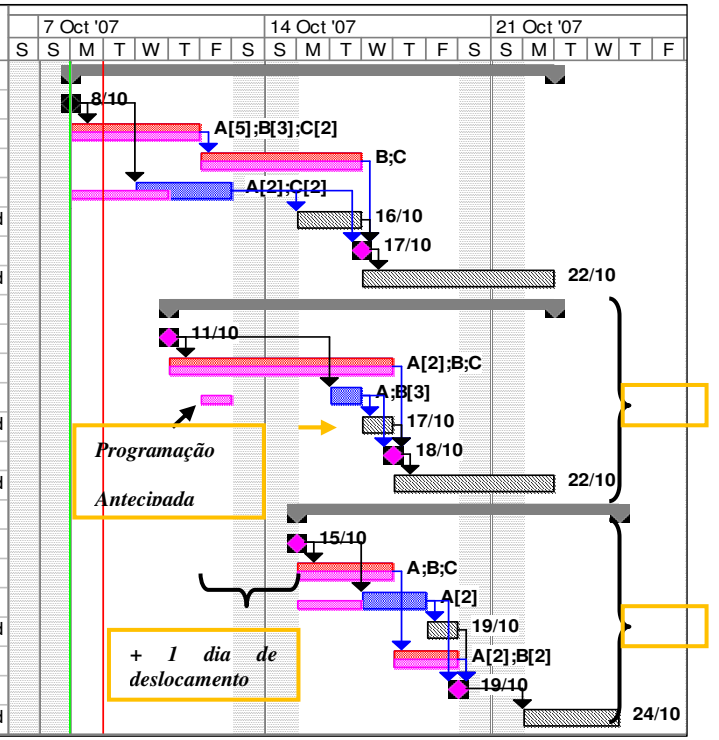

Figura 4.12: Programa máster segundo prioridades 1,3,2. Status ao dia 9/10

O projeto 2 é deslocado um dia mais da posição que ocupava o projeto 3 quando tinha a terceira prioridade. Isso é pela programação antecipada do ativ.2 do proj. 3 (que agora tem a prioridade dois). Repare-se que o programa deixa a ativ.2 do proj 3 com uma folga de 3 dias (ressaltada em laranja) ademais do espaço que o BA (um dia) gera. A antecipação da programação (no programa projetado) dessa atividade ocasiona que não exista mais disponibilidade do recurso B para programar o projeto dois

Atrasar o programa projetado do projeto 3 não afeta suas datas de termino, mas sim permite que a data de termino do projeto 2 seja menor num dia.

4.3.3.
Abordagem do "Recurso estratégico ou gargalo"

4.3.3.1.
Programa Normal.

Esse programa também é desenvolvido no cenário proporcionado pelo máster. A abordagem aplica fielmente as recomendações da teoria de restrições, definindo como única restrição do sistema, num dado momento, o recurso mais escasso. Contudo, a ferramenta de *software* utilizada permite definir mais de um recurso gargalo, tolerando a implementação de programas híbridos que interpolam entre "o programa do recurso estratégico" e "o programa dos projetos sucessivos", tal e como menciona Newbold (1998).

Para o exemplo, se define como gargalo do sistema, o recurso "A", por ser ele o recurso que inicialmente possui a maior carga de trabalho; além de ser o que tem participação no maior número de atividades. A Figura 4.13 mostra o máster resultante da abordagem. Repare-se que em comparação com a abordagem anterior, o projeto dois tem uma data de início mais cedo, implicando em recursos não gargalo sobrecarregados.

Nessa abordagem a programação se interessa por deixar ao recurso "A" livre de sobrecargas e folgas, é dizer, programam-se o maior número de atividades que o gargalo possa suportar (Figura 4.14). Neste contexto, os recursos "B" e "C" são deliberadamente sobrecarregados (Figura 4.15 e Figura 4.16) seguindo a lógica de que é mais conveniente, em simplicidade e economia, aumentar estes recursos do que deixar o gargalo folgado ou sobrecarregado. Para mitigar tais conflitos, a metodologia utiliza os *buffers* de capacidade – *BC*, que, embora tenham como função principal a proteção do recurso gargalo, também reduzem as sobrecargas dos recursos em conseqüência do espaço que o BC gera entre os projetos.

O programa da Figura 4.17 mostra o máster que considera BC's dimensionados a 25% (ver lógica de cálculo no apêndice A). Os efeitos que tais *buffers* têm sobre os recursos podem ser observados nos relatórios de carga de trabalho na Figura 4.18 e na Figura 4.19. Repare-se como a inserção do BC provoca que o excesso de carga no recurso "B" seja relegado no final do projeto e que a carga do recurso "C" seja totalmente eliminada. Repare-se também, que

essa solução faz com que as datas de entrega desses projetos sejam postergadas no tempo. Da mesma maneira, a data final do máster é postergada para o dia 26/10, ultrapassando o programa ótimo (23/10) apresentado por Pritsker B. *et al.* (1969.).

O dimensionamento do BC gera uma curva de *"trade-off"* na decisão de ter um sistema bem equilibrado e com altos níveis de proteção, ou um sistema de *makespan* curto, projetos menos deslocados no tempo, mas, com maior probabilidade de interferência entre eles. . Esse efeito explica que projetos mais isolados, com maiores BC, geram sistemas mais seguros e de maior duração. É possível que tais projetos sejam considerados como menos competitivos em situações de concorrência no mercado.

Certamente a localização ótima dentro desta curva de *"trade-off"* será diferente para cada situação, entretanto, para tomar essa decisão é necessário entender o comportamento e os efeitos que o BC tem. Para tal fim, criaram-se diferentes cenários de programação do BC. Estes são analisados e comparados durante a etapa de execução.

Inicialmente se trabalha com o programa de base que considera BC de 25%, a Figura 4.20 mostra o programa correspondente à etapa final da execução. Repare-se como a inserção desse tamanho de BC provoca um grau de isolamento de projetos superior à necessidade do sistema, fazendo com que o consumo dos BP dos projetos dois e três seja mínimo. O tempo não utilizado do BP é análogo à matéria prima desperdiçada num sistema de manufatura, o tempo e recursos disponibilizados no projeto sobre-dimensionado poderiam ter sido alocados em algum outro projeto.

A partir disso surge o questionamento de como saber oportunamente se os projetos precisam de mais ou menos segurança em relação à interação com outros projetos do sistema. Mediante o uso de "programas híbridos",que em vez de inserir os BC no programa de base os insere no programa projetado, permite-se observar, no primeiro dia execução, o efeito que o deslocamento dos projetos (ocasionado pelos BC) teria sobre os BP

Essa análise demonstra alguns indícios que ajudam a determinar o tamanho do BC. Ao observar o efeito que este tem no consumo dos BP, pode-se ter uma idéia da dimensão do BC como complemento de segurança do BP na proteção do projeto. Quer dizer que, se as inserções dos BC no programa projetado provocam o consumo total (ou grande parte) do BP, é muito provável que o BC esteja sobre-

dimensionado. Da mesma maneira, se o consumo do BP é nulo (ou pouco) é possível que o projeto precise de maior proteção para agüentar os efeitos das interdependências.

Essa lógica pode ser observada no programa híbrido da Figura 4.21, onde o BP do projeto dois é ultrapassado pela programação projetada da terceira atividade, enquanto o consumo do BP do projeto 3 oscila em torno de 50%. Tais características indicam que o espaço gerado pelo BC que protege o projeto dois está claramente sobre-dimensionado. Isto se explica observando o recurso gargalo nos projetos um e dois. Repare-se que, mesmo compartilhando o recurso "A" não precisamente interdependem por causa dele. Isso porque a disponibilidade do gargalo suporta as atividades de ambos os recursos, fazendo com que a necessidade do BC seja unicamente para nivelar os recursos não gargalos. No entanto, o projeto três interdepende de todos os projetos que o precedem, motivo pelo qual, precisaria um BC maior do que o projeto dois. Lembre-se que o BC dimensionado a 25% não deu indícios de estar errando nem por falta nem por excesso, isto é corroborado na Figura 4.20, onde o BP chega a ter certo grau de funcionalidade.

Essa programação híbrida sugere que a determinação dos BC teria que ser dependente das características dos projetos. Contudo, a ferramenta de *software* obriga que os BC sejam dimensionados unicamente com base no recurso gargalo, sem permitir utilizar diferentes percentagens de dimensionamento para cada projeto. Entretanto, a metodologia não se aprofunda nos critérios e métodos de dimensionamento dos BC. A análise da relação entre os BC e as características dos projetos é aprofundada no Capítulo V.

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-5	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**50**	**0**			
2	**Atividade 1**	**3 days**	**0**	**50**	**0**			
3	Atividade 2	2 days	-33	50	1			
4	FB|Atividade 2-3|Fim-5	1 day	0	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**0**	**50**	**0**			
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	3 days	0	0	0	1-1;Cero	5d	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**-67**	**0**	**2**			
2	**Atividade 1**	**5 days**	**0**	**100**	**0**			
3	Atividade 2	1 day?	-133	0	4			
4	FB|Atividade 2-3|Fim-4	1 day	-300	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**0**	**100**	**0**			
6	PB|Fim-4	3 days	0	0	0	2-2;Atividade 1	5d	5d

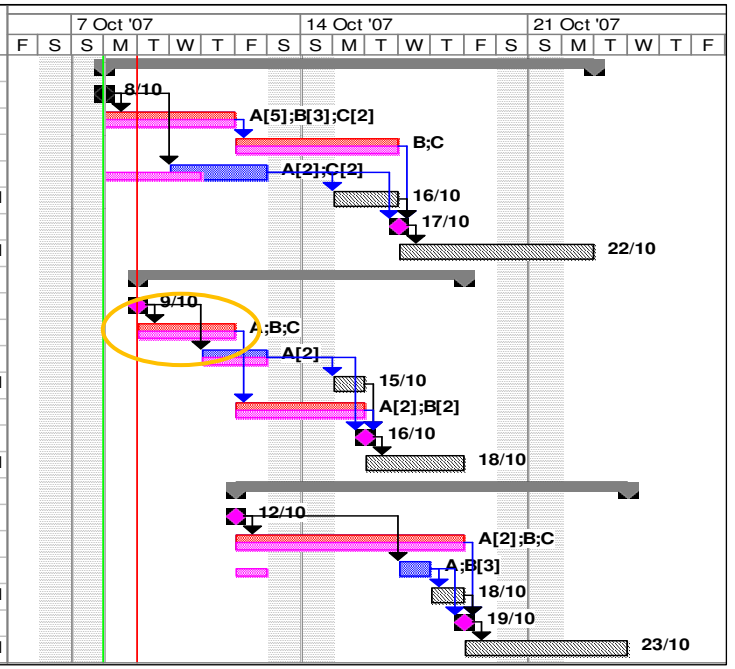

Figura 4.13: Programa master baseado no recurso gargalo "A". Status do dia 8/10

A data de início do projeto dois é antecipada dois dias em relação á figura 4.11. Repare-se que o *programa projetado* permite sobrecargas dos recursos não gargalos (C e B). Isso define uma data de término mais cedo para o projeto 2 (18/10).

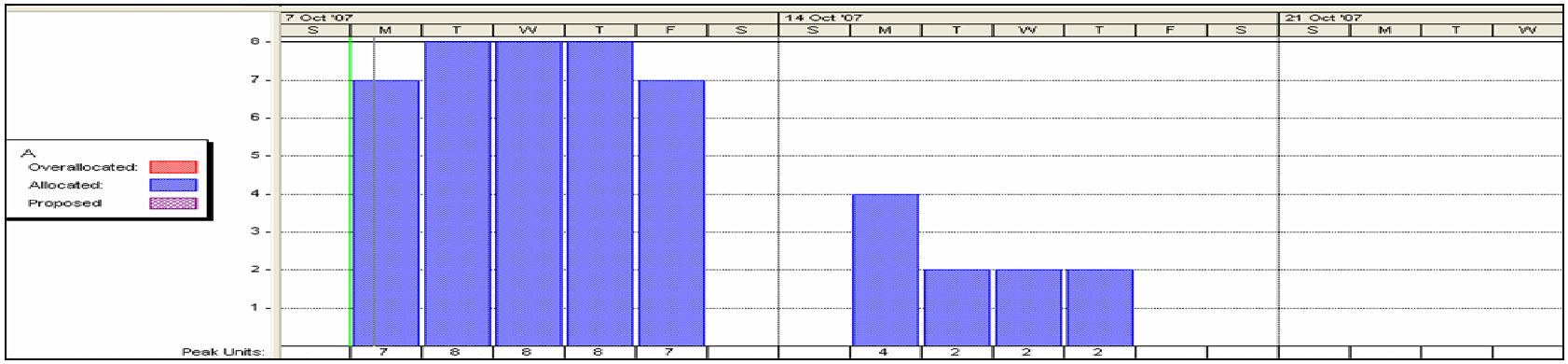

Figura 4.14: Diagrama de carga de trabalho, Recurso "A" (Programa sem BC)

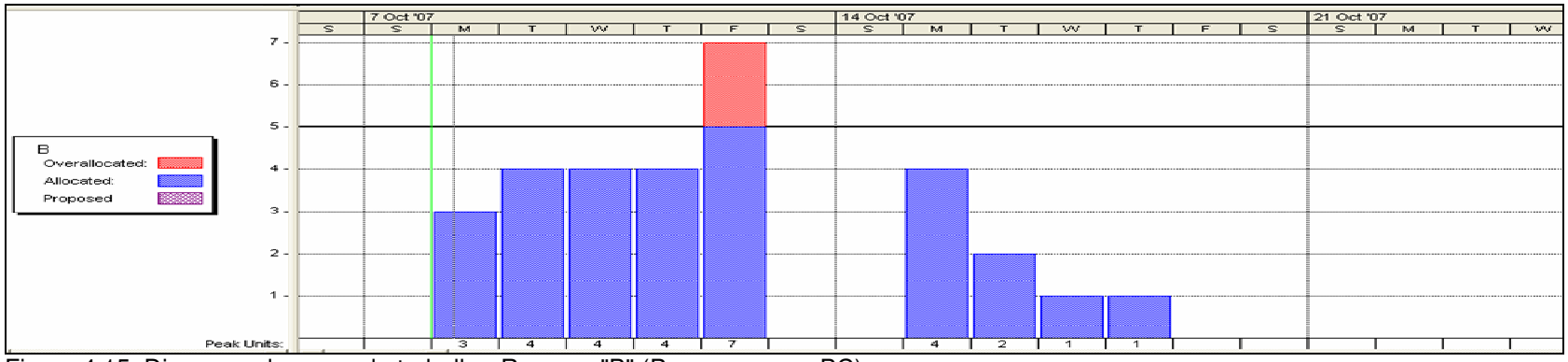

Figura 4.15: Diagrama de carga de trabalho, Recurso "B" (Programa sem BC)

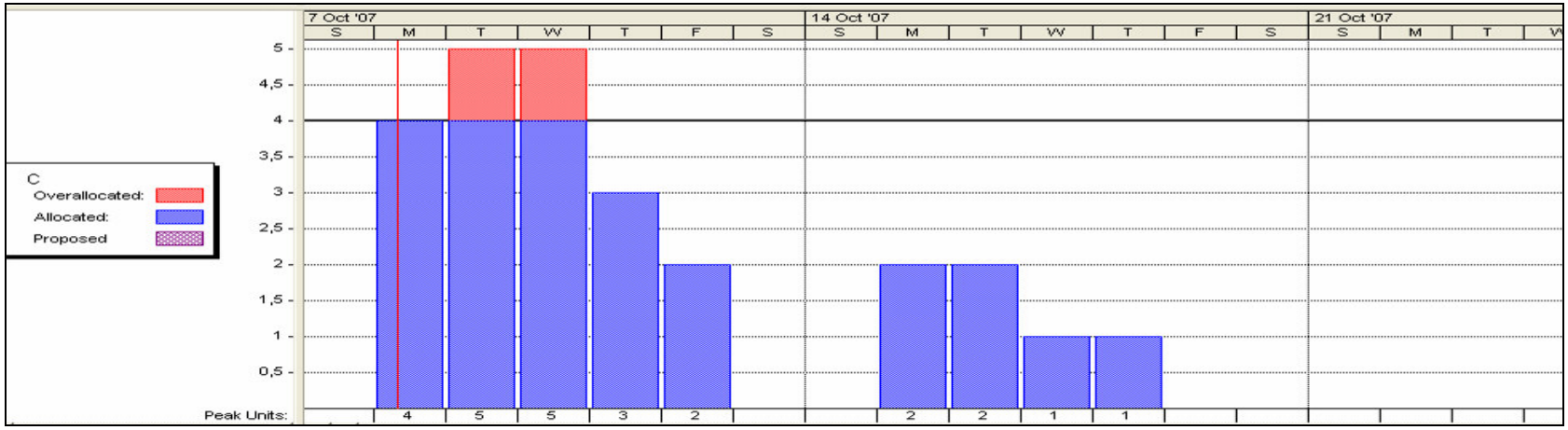

Figura 4.16: Diagrama de carga de trabalho, Recurso "C"(Programa sem BC)

ID	Task Name	Focus Dur	% Impact	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	Multiproject_Pristsker	11 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade um	4 days	0	100	0			
3	Atividade dois	3 days	0	100	0			
4	Atividade tres	3 days	-100	0	4			
5	FB|Atividade tres-3|Fim-	2 days	-100	0	0	4-3;Atividade tres	3d	3d
6	Fim	0 days	0	100	0			
7	PB|Fim-5	4 days	0	0	0	2-1;Atividade um	7d	7d
2	Multiproject_Pristsker 2	8 days	0	0	0			
1	Cero	0 days	-111	0	3			
2	Atividade 1	3 days	0	100	0			
3	Atividade 2	2 days	-100	0	3			
4	FB|Atividade 2-3|Fim-5	1 day	-200	0	0	3-3;Atividade 2	2d	2d
5	Atividade 3	2 days	0	100	0			
6	Fim	0 days	0	100	0			
7	PB|Fim-5	3 days	0	0	0	2-2;Atividade 1	5d	5d
3	Multiproject_Pristsker 3	8 days?	0	0	0			
1	Cero	0 days	-102	0	3			
2	Atividade 1	5 days	0	100	0			
3	Atividade 2	1 day?	-133	0	4			
4	FB|Atividade 2-3|Fim -4	1 day	-300	0	0	3-3;Atividade 2	1d	1d
5	Fim	0 days	0	100	0			
6	PB|Fim -4	3 days	0	0	0	2-2;Atividade 1	5d	5d

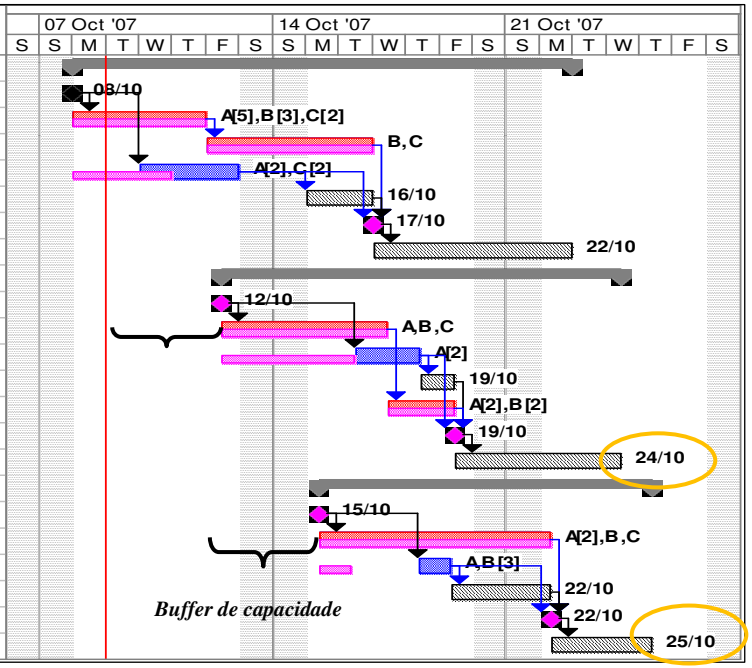

Figura 4.17: Programa máster com BC ao 25% . Status do dia 8/10

A inserção de BC definidos em 25% da carga do recurso gargalo (A), faz que o projeto 2 seja deslocado do 9/10 ao 12/10, e que o projeto 3 seja deslocado do 12/10 ao 15/10. Isto ocasiona que o *makespan* do sistema seja incrementado em 14 dias, (do 8/10 ao 25/10), fazendo que as datas de término dos projetos 2 e 3 sejam mais tarde.

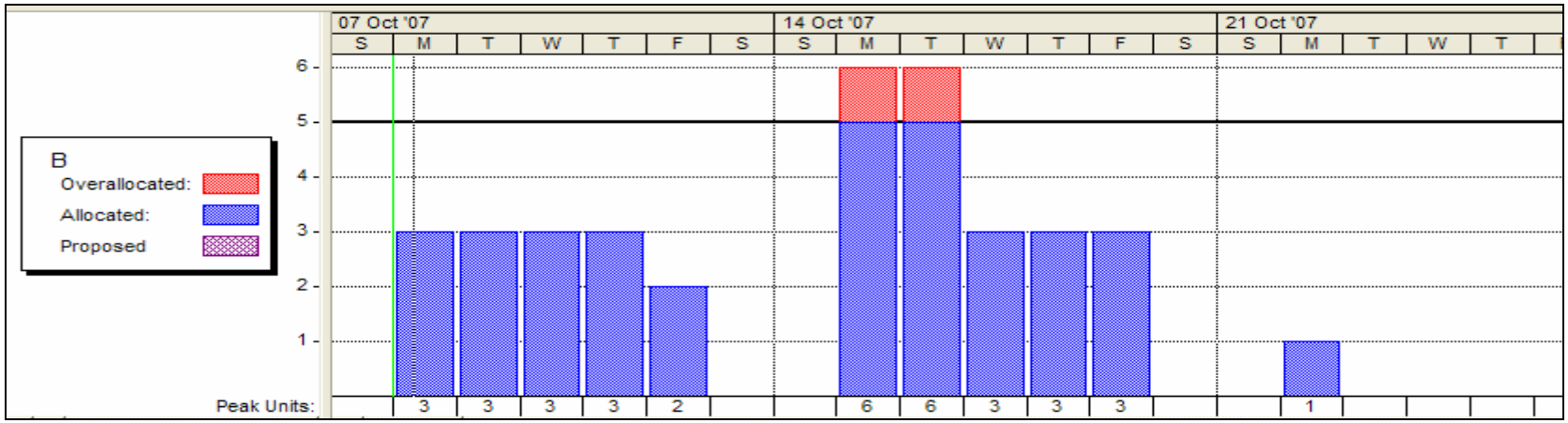

Figura 4.18: Diagrama de carga de trabalho, Recurso "B" (Programa com BC 25%)

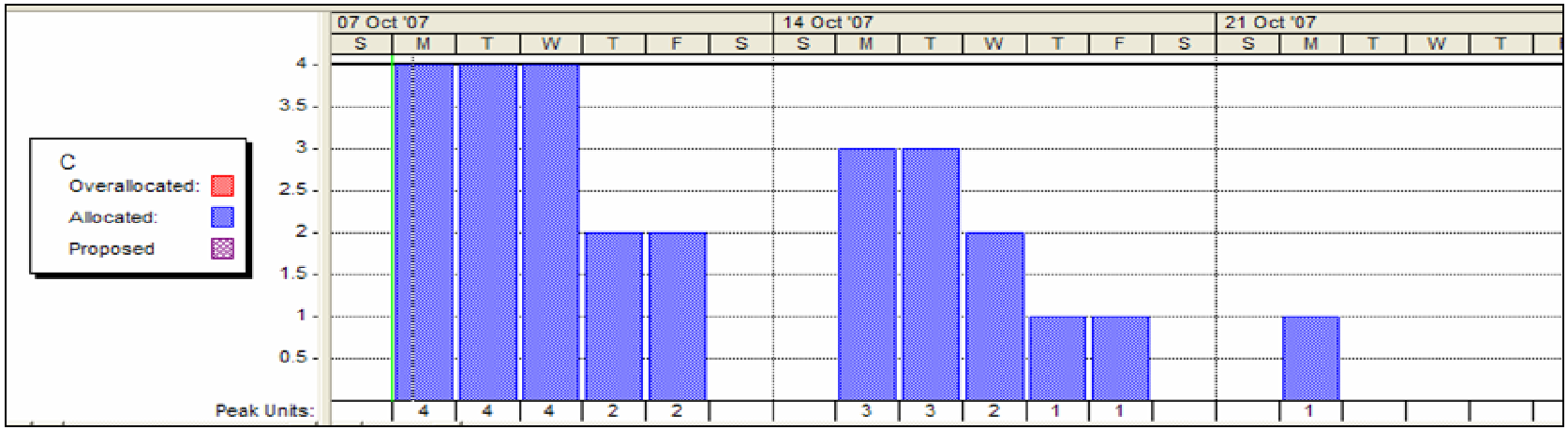

Figura 4.19: Diagrama de carga de trabalho, Recurso "C" (Programa com BC 25%)

ID	Task Name	Focus Dur	% Impac	Reduction Impact	Days Slack	Check Task	Chain Left	Orig. Chair
1	Multiproject_Pristsker	11 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade um	5 days	0	100	0			
3	Atividade dois	3 days	25	100	0			
4	Atividade tres	4 days	0	0	0			
5	FB|Atividade tres-3|Fim-	2 days	-50	0	0		0d	3d
6	Fim	0 days	25	100	0			
7	PB|Fim-5	4 days	25	0	0	3-2;Atividade dois	0.99d	7d
2	Multiproject_Pristsker 2	8.33 days	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade 1	3 days	0	100	0			
3	Atividade 2	2 days	0	0	0			
4	FB|Atividade 2-3|Fim-5	1 day	-233	0	0		0d	2d
5	Atividade 3	2 days	-11	100	0			
6	Fim	0 days	-11	100	0			
7	PB|Fim-5	3 days	-11	0	0	5-4;Atividade 3	2d	5d
3	Multiproject_Pristsker 3	8.06 days?	0	0	0			
1	Cero	0 days	0	0	0			
2	Atividade 1	5 days	39	100	0			
3	Atividade 2	1 day?	-61	0	3			
4	FB|Atividade 2-3|Fim -4	1 day	-84	0	0	3-3;Atividade 2	1d	1d
5	Fim	0 days	39	100	0			
6	PB|Fim -4	3 days	39	0	0	2-2;Atividade 1	4d	5d

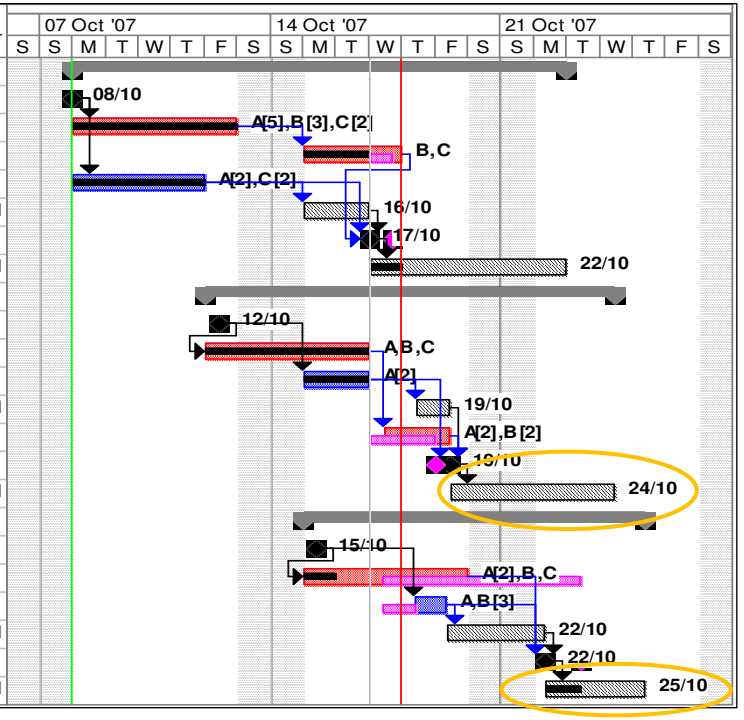

Figura 4.20: Programa máster com BC 25%. Status do dia 17/10

A inserção de BC definidos em 25% isola os projetos, evitando que estes tenham alto grau de compartilhamento de recursos. Não obstante, atrasando as mesmas atividades do que nos casos anteriores se percebe que na etapa final de execução o BP do projeto dois se mantém inativo. Ou seja, houve uma alocação de tempo no projeto que foi desperdiçada, podendo ter se alocado nesse tempo outros projetos ou atividades. O BP do projeto 3 apresenta um consumo provocado unicamente pelas interdependências indicando que o complemento de segurança que o BC lhe da é necessário.

ID	Task Name	Focus Dur	% Impact	Reductio Impact	Days Slack	Check Task	Chain Left	Orig. Chain
1	**Multiproject_Pristsker**	**11 days**	**0**	**0**	**0**			
1	Cero	0 days	0	0	0			
2	**Atividade um**	**4 days**	**0**	**100**	**0**			
3	**Atividade dois**	**3 days**	**0**	**100**	**0**			
4	Atividade tres	3 days	0	0	0			
5	FB|Atividade tres-3|Fim-5	2 days	-50	0	0	4-3;Atividade tres	3d	3d
6	**Fim**	**0 days**	**0**	**100**	**0**			
7	PB|Fim-5	4 days	0	0	0	2-1,4-3;Atividade um	6d	7d
2	**Multiproject_Pristsker 2**	**8 days**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**50**	**0**			
2	**Atividade 1**	**3 days**	**100**	**50**	**0**			
3	Atividade 2	2 days	17	50	3			
4	FB|Atividade 2-3|Fim-5	1 day	150	0	0	3-3;Atividade 2	2d	2d
5	**Atividade 3**	**2 days**	**100**	**50**	**0**			
6	**Fim**	**0 days**	**100**	**100**	**0**			
7	PB|Fim-5	3 days	117	0	0	2-2;Atividade 1	5d	5d
3	**Multiproject_Pristsker 3**	**8 days?**	**0**	**0**	**0**			
1	**Cero**	**0 days**	**0**	**0**	**0**			
2	**Atividade 1**	**5 days**	**41**	**100**	**0**			
3	Atividade 2	1 day?	-92	0	4			
4	FB|Atividade 2-3|Fim-4	1 day	-177	0	0	3-3;Atividade 2	1d	1d
5	**Fim**	**0 days**	**41**	**100**	**0**			
6	PB|Fim-4	3 days	41	0	0	2-2;Atividade 1	5d	5d

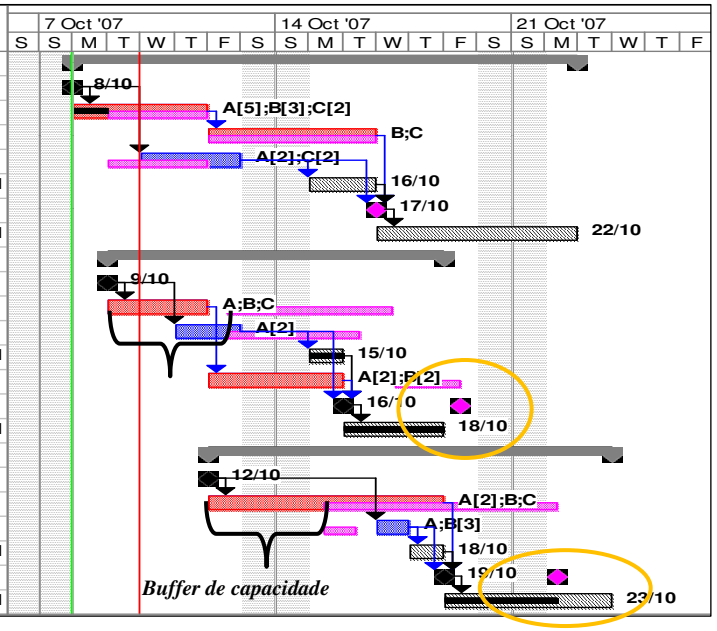

Figura 4.21: Programa máster híbrido. Status do dia 09/10 (Programa projetado com BC 25%)

O programa híbrido em vez de inserir o BC dentro do *programa de linha de base* o insere no *programa projetado*. Repare-se como existe um espaçamento entre o início das atividades de linha de base e as projetadas. Isso permite observar, no primeiro dia de execução, a relação de tamanho que existe entre os BC e BP em função do consumo que o primeiro ocasiona no segundo. Veja como no projeto 2 o consumo do BP ultrapassa o programado, indicando que o BC está sobre-dimensionado. Ao fazer esta comparação se espera que o consumo dos BP ocasionado pelos BC não seja próximo do total nem do zero.
A necessidade que cada projeto tem do BC é em diferentes dimensões.

5
Efeitos do buffer de capacidade e da priorização de atividades

Neste capítulo se analisam os efeitos que a implementação dos *buffers* de capacidade – BC têm sobre o sistema. Para tal fim, construiu-se um modelo de simulação que representa a execução de múltiplos projetos, permitindo separar os atrasos causados por conflitos de recursos daqueles causados pela aleatoriedade das durações das atividades.

Neste capítulo também são analisados os efeitos que o tipo de priorização de atividades utilizado durante a etapa de execução tem sobre o sistema. Com essa finalidade foi implementado, no modelo anterior, um processo decisório que permite executar programas baseados em prioridades fixas e dinâmicas.

A apresentação dos resultados é iniciada com a definição dos objetivos, parâmetros, cenários e lógica de simulação a cada experimento. Finalmente se apresentam os gráficos e tabelas relevantes para o entendimento dos resultados. Outras informações complementares são apresentadas no apêndice B.

5.1.
Efeitos dos *buffers* de capacidade - BC

5.1.1.
Objetivos

O modelo de simulação tem como objetivo representar a etapa de execução dos múltiplos projetos do máster, permitindo a identificação e entendimento dos efeitos que os BC têm sobre os objetivos do sistema e dos projetos individuais. Com o modelo, pretende-se esclarecer se efetivamente há necessidade de dimensionar os BC de acordo com cada projeto e não como uma porcentagem fixa (da capacidade do recurso gargalo) para todos os projetos.

Do mesmo modo, busca-se explicar o desempenho do BC como complemento de segurança do BP e analisar o compromisso (*trade-off)* entre

atingir um sistema de maior estabilidade (menos atrasos por falta de recursos) e estabelecer um sistema de menor *makespan*.

5.1.2. Estrutura da Simulação

O modelo de simulação é baseado na programação híbrida (definida no Capítulo IV), em que os BC são inseridos somente no "programa projetado". A simulação representa a interação entre o "programa base", definido pelo recurso gargalo, e a execução probabilística das atividades do "programa projetado" com os BC inseridos.

A simulação é realizada em processos que permitem corridas de até 40 horas de duração, o que é equivalente a programas que se estendam até 40 unidades de tempo (*t)*. Para cada cenário são efetuadas 80 réplicas com valores aleatórios distintos entre si. Os valores aleatórios das 80 réplicas são similarmente reproduzidos em cada cenário, permitindo, que sua análise seja sob as mesmas condições. A decisão de 80 replicações for arbitraria e poderá ser estendida para menor variância nos resultados.

- **Programa de base**. - O modelo é baseado na programação base, obtida na abordagem do recurso estratégico ou gargalo, no exemplo o recurso "A" é definido como gargalo do sistema.

  Com a finalidade de isolar o comportamento dos BC em relação aos BP, são observadas algumas particularidades no programa *master* da Figura 5.1

  - São representadas apenas as atividades críticas e os respectivos BP para cada projeto. Isso porque, para a finalidade da análise, as atividades não críticas e seus respectivos BA representam um aumento desnecessário da complexidade do modelo.
  - Para manter um significativo compartilhamento de recursos entre os projetos do sistema, as disponibilidades de recursos do exemplo são redefinidas: "A" = 6, "B" = 4, "C" = 3.
  - Mesmo que o programa de base seja nivelado em relação ao recurso "A", o modelo exige que a execução das atividades satisfaça o nivelamento de todos os recursos. Dessa maneira se consegue estudar os BC num

ambiente de exigência máxima. Esse procedimento de execução, ao desconsiderar as atividades não críticas, não entra em conflito com o programa básico original.

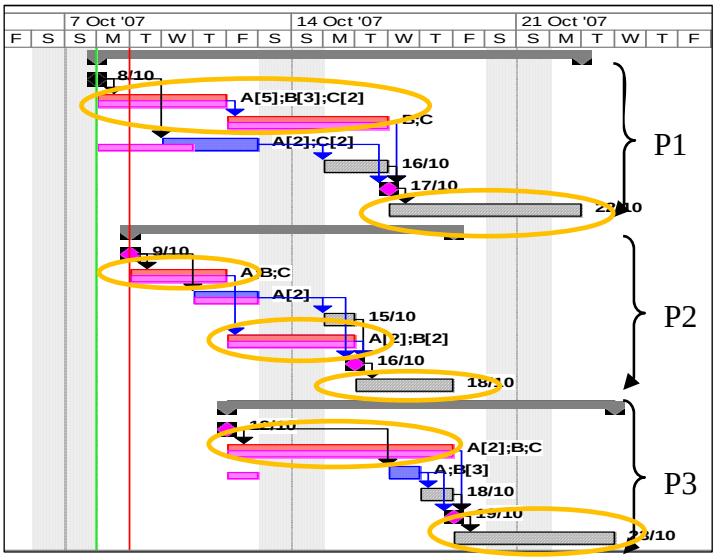

Figura 5.1: Programa base do modelo de simulação

A definição do programa de base é feita através de variáveis que, de acordo com a seqüência estabelecida na Figura 5.1, determinam o início e fim de cada atividade. Dessa maneira o programa de base se mantém fixo durante o processo de simulação e é usado como referência na determinação do estado de execução das atividades. Como exemplo se mostra na Tabela 5.1 as variáveis definidas para as atividades do projeto um.

Tabela 5.1: Descrição das variáveis que definem o programa de linha de base

Variáveis do programa de linha de base do projeto um
P1A1_TIB = 0
O instante zero é atribuído ao tempo de início básico da primeira atividade.
P1A1_TFB = P1A1_TIB + 4
O tempo final básico da primeira atividade está definido pelo tempo de início mais a duração fixa da atividade.
P1A2_TIB = P1A1_TFB
Ao tempo de início básico da segunda atividade é atribuído o mesmo instante em que a primeira atividade é terminada, existe uma restrição de precedência.
P1A2_TFB = P1A2_TIB + 3
O tempo de término básico da segunda atividade é definido pelo tempo de início mais a duração fixa da atividade.

- **Programa Projetado. -** O programa projetado é determinado por meio de variáveis, que mudam dinamicamente com o avanço da simulação. A inserção do espaço definido pelos BC é realizada no momento um da execução (*t = 1*), sendo que a simulação é iniciada no momento zero.

A Tabela 5.2 mostra a definição das variáveis correspondentes ao projeto dois. O código completo pode ser encontrado no apêndice B.

Tabela 5.2: Descrição das variáveis que definem o programa projetado

Variáveis do programa projetado do projeto dois
P2A1_TIP = P2A1_TIB + P2A1_varTIP + P2_BC
O tempo de início projetado da primeira atividade é definido pelo tempo de início básico mais o incremento de uma unidade de tempo, a qual é somada quando a atividade é efetivamente executada. A partir do momento um da simulação se adiciona ao cálculo o valor fixo do *buffer* de capacidade correspondente.
P2A1_TFP = P2A1_TIP + 3 - P2A1_varTFP
O tempo final projetado da primeira atividade é definido pelo tempo de início projetado mais a duração fixa da atividade. Quando a atividade é efetivamente executada se diminui uma unidade de tempo ao valor calculado. Isso permite que o tempo final projetado se mantenha fixo e que o cálculo da quantidade de trabalho não completado seja diminuído.
P2A3_TIP = P2A1_TFP+ P2A3_varTIP
O tempo inicial projetado da terceira atividade é definido pelo tempo final projetado da atividade anterior, (mantendo a restrição de precedência definida no programa base). O incremento da unidade de tempo tem a mesma lógica que a atividade P2A1_TIP. O BC é aumentado unicamente na primeira atividade e transmitido na cadeia através das dependências de seqüenciamento.
P2A3_TFP = P2A3_TIP + 2 - P2A3_varTFP
O tempo de término da terceira atividade tem a mesma lógica da atividade P2A1_TFP.

- **Distribuição de Probabilidade. –** O modelo utiliza a função de distribuição binomial negativa para descrever o comportamento da duração das atividades na etapa de execução.

A binomial negativa é uma distribuição de probabilidade discreta que pode ser usada para descrever experimentos que consistem numa seqüência de tentativas independentes. Cada tentativa pode resultar em sucesso ou falha, sendo a probabilidade de sucesso, ***p.*** constante para todas as tentativas do experimento.

A binomial negativa pode ser definida tanto em função do número de tentativas como em função do número de falhas (num experimento de Bernoulli) requeridas até se obter um dado número de sucessos. (CASELLA & BERGER, 2002)

A lógica do modelo de simulação usa a definição da distribuição baseada no número de falhas. Cada iteração simulada corresponde a uma tentativa de execução da atividade num dado intervalo de tempo, podendo esta resultar em uma correta execução do trabalho planejado (sucesso) ou na perda desse tempo (fracasso). Não são considerados atrasos de frações de tempo, permitindo somente durações de valores múltiplos (t) da unidade de tempo..

O processo de simulação é repetido em cada atividade até que o número de sucessos iguale à duração planejada da atividade. Dessa maneira, o numero de tentativas (somatória do número de falhas e do número de sucessos) determina a duração total da atividade.

A definição formal da função de distribuição binomial negativa em função da variável aleatória Y = número de falhas antes do r^{ésimo} sucesso, é dada por:

$$P(Y = y) = \binom{r + y - 1}{y} p^r (1 - p)^y, \qquad y = 0,1,\ldots$$

Parâmetros	*p = probabilidade de sucesso* *r = número de sucessos alvo*
Média	$= r\frac{(1-p)}{p}$
Variância	$= r\frac{(1-p)}{p^2}$

Onde: Y = X – r; X = número de tentativas até o r^{ésimo} sucesso.

- A forma da distribuição binomial negativa para o problema modelado (

Figura 5.2) é similar à distribuição que descreve as atividades de um projeto, definida por Leach (2005). Nesse tipo de distribuições as durações extremamente curtas como as durações extremamente longas têm uma chance

aceitável de acontecer. Para valores grandes de r a distribuição tende a ser mais simétrica.

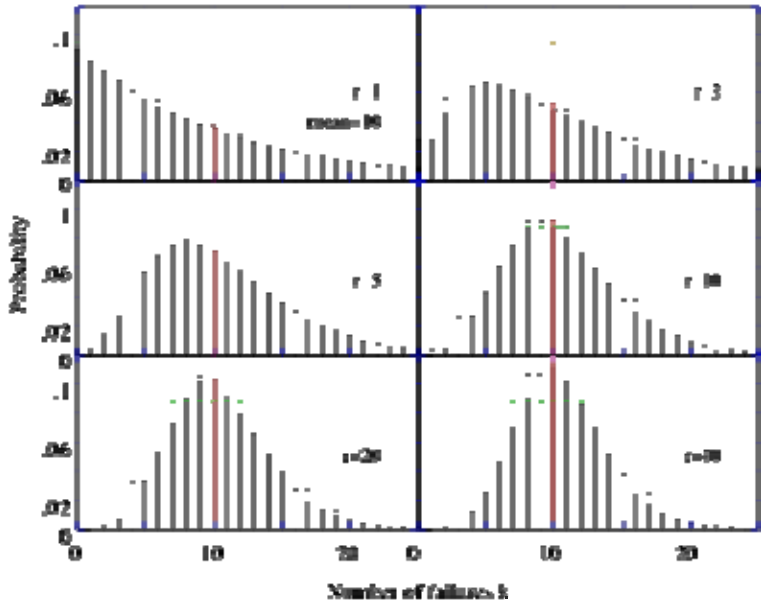

Figura 5.2: Distribuição Binomial negativa
Fonte: http://en.wikipedia.org

Para implementar a distribuição binomial negativa no modelo, faz-se uso da propriedade das distribuições binomiais de que a soma de duas ou mais variáveis binomiais independentes com o mesmo parâmetro p é ainda uma variável binomial com parâmetros r igual ao somatório dos ri e o mesmo p.

Desse modo se considera que as probabilidades de sucesso das diversas atividades de um mesmo projeto são independentes e de mesmo valor p. Conseqüentemente, o valor de **r** é dado pela duração enxuta (sem considerar o buffer) do projeto.

Com esses pressupostos, calcula-se o valor de p em função da probabilidade média de atraso (ou falhas) aceitável na estimação da duração do projeto. Ou seja, com base na probabilidade de ter tantos atrasos na execução das atividades do projeto quanto o buffer do projeto possa suportar. Esse cálculo é feito na planilha eletrônica (*Excel*) e utilizado numa ferramenta de simulação (*ProccessModel*) que onde se insere o caráter probabilístico do modelo . Diferentes valores de probabilidade de atrasos foram definidos para a análise dos modelos.

- **Os parâmetros e cenários de simulação**. – Os diferentes cenários de simulação são definidos por dois parâmetros principais:

  1. A porcentagem utilizada para dimensionar o BC: O cálculo dessa porcentagem considera que o tempo reservado pelo *buffer* é complemento do tempo que o recurso investe na atividade anterior da seqüência crítica (porcentagem que completa o 100% da disponibilidade do recurso). Lembrar que os BC são inseridos entre duas atividades que iniciem consecutivamente dentro da seqüência crítica e que pertençam a projetos diferentes. Portanto, no modelo são inseridos dois BC calculados segundo o exemplo:

P2_BC = ((P1A1_TFB - P1A1_TIB)* BC2_tam)/(1- BC2_tam)
O dimensionamento do BC que protege ao projeto dois é definido pela multiplicação da duração da primeira atividade do projeto um e a porcentagem do dimensionamento do BC, tudo divido pelo complemento de tal porcentagem.

  2. O valor da probabilidade de sucesso (execução conforme programa) e falha (sem progresso) que define o comportamento aleatório da duração real das atividades: Devido que a ferramenta de software não suporta a implementação direta da distribuição binomial negativa, ela foi implementada dentro da lógica de programação. A ferramenta de simulação permite definir duas probabilidades independentes (de sucesso e falha) para determinar o sorteio aleatório da execução de atividades. Por meio da lógica de programação se dispõe que o sorteio seja efetuado até que o número de sucessos aleatórios seja igual à duração planejada das atividades, tal e como é o processo gerador da distribuição binomial negativa.

  A combinação desses dois parâmetros define o conjunto de cenários avaliados. Para diferentes probabilidades de atraso do projeto (25%, 20%, 15% e 10%) foram experimentados diferentes tamanhos de BC (0%, 15%, 25%, 35% e 45%), definindo 20 cenários de avaliação.

- **A lógica do processo de execução**. – O modelo define o “Tempo” como a principal entidade que aciona o processo de execução de atividades, este se desloca através das atividades, ativando a execução daquelas que segundo o

programa de linha de base e o programa projetado estão prontas para ser trabalhadas. Para simplificar a representação dos projetos, define-se que uma hora é equivalente a uma jornada laboral (comumente dias de 8 horas).

A Figura 5.3 mostra a representação gráfica do modelo. As esferas correspondem às unidades de tempo que se deslocam através do processo de execução de atividades. A existência de três entidades "Tempo" se movimentando em forma paralela indica a simultaneidade da execução dos projetos. Três tipos de saídas indicam a mudança de ciclo de processamento, ou seja, o término de um dia de trabalho.

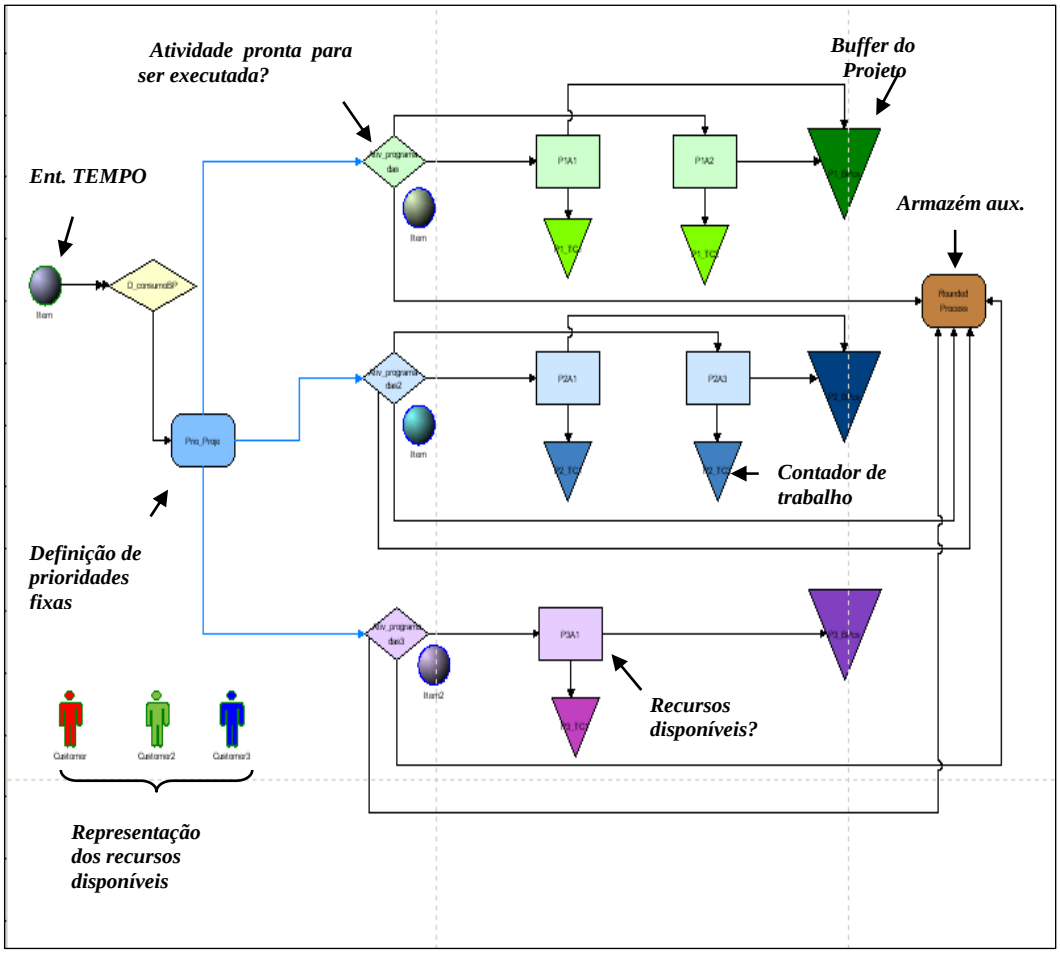

Figura 5.3: Modelo de simulação dos efeitos do BC

A entidade "Tempo" pode sair do sistema através de um "armazém" ou "contador do trabalho efetivamente realizado" (quando a execução da atividade não é retardada), através do buffer do projeto (quando a execução da atividade é retardada) ou por meio de um "armazém auxiliar" (quando não

existe nenhuma atividade pronta para ser executada), esta última opção não implica mudanças nos tempos projetados.

O processo de execução é iniciado com a definição das prioridades de cada projeto, neste caso as prioridades são determinadas segundo a ordem de chegada (1,2,3) e mantidas fixas durante todo o processo.

A partir das prioridades estabelecidas, se define uma lógica interna que permite liberar primeiro as entidades de tempo dos projetos prioritários. O "*delay*" aplicado às outras atividades, não afeta a cronometragem do modelo, mas, sim, permite que os recursos sejam alocados às atividades segundo as prioridades definidas.

A primeira decisão de execução é tomada nos losangos do diagrama, cada projeto tem entre três e quatro caminhos opcionais pra entidade tempo. Para poder entrar na primeira atividade (de qualquer projeto) a condição: Tempo de início base da atividade (TIB) < = Momento de execução atual, ou, Tempo de início projetado (TIP) = Momento de execução, tem que ser cumprida. Para entrar numa atividade, é preciso que as atividades anteriores tenham sido completadas. As outras duas opções chegam ao armazém auxiliar, trasladando a entidade quando o projeto tiver terminado e quando o projeto é deslocado no tempo por causa dos BC (condição: TIB < = Momento de execução e TIP > momento de execução).

Uma vez que a entidade "Tempo" aciona a atividade, significa que esta está pronta para ser executada, não obstante, é preciso se avaliar a disponibilidade dos recursos. Para que a atividade possa ser executada, a necessidade de todos seus recursos tem que ser satisfeita, caso não seja assim, tal atividade será atrasada e o BP será consumido numa unidade de tempo.

Depois de alocar os recursos necessários para trabalhar a atividade e de atualizar o número disponível de cada recurso, procede-se com o sorteio aleatório que representa o resultado (sucesso ou falha) da execução. Caso esse resultado seja negativo (falha), a entidade "Tempo" será trasladada ao BP (consumindo-o). Caso o resultado seja positivo (sucesso), a entidade Tempo será trasladada ao armazém do trabalho realizado.

Este processo é efetuado de forma simultânea nos três projetos, assim também as atualizações de todos os tempos de início e fim projetados são feitas cada vez que a entidade “Tempo” encontre uma saída do processo.

Ademais do consumo do BP originado pelos atrasos das atividades, existe um consumo imediato originado pela inserção dos BC.

Os BP podem ser consumidos infinitamente, sendo que outro contador (*delay* ou atraso do projeto) é ativado no momento em que o consumo ultrapassa o limite do *buffer*.

Repare-se que o modelo permite identificar e contabilizar as diferentes naturezas do atraso do projeto, facilitando o isolamento do efeito dos BC.

5.2. Efeitos da priorização de atividades.

5.2.1. Objetivos

O modelo de simulação tem como objetivo representar o processo e as conseqüências que a aplicação de diferentes critérios de priorização de atividades durante a execução tem sobre o sistema. O propósito é esclarecer se o uso do critério de priorização dinâmica, com base no estado de consumo dos *buffers* do projeto – BP, obtém melhores resultados (em termos de tempo) do que a priorização fixa.

Neste contexto, o modelo busca, também, proporcionar a informação necessária para entender a essência das diferenças entre ambos os critérios e o porquê de seus efeitos no sistema.

5.2.2. Estrutura da simulação

O modelo utilizado é uma adequação do modelo anterior. Adiciona-se uma decisão no início do processo para determinar o critério a utilizar. O modelo permite apenas a utilização de um único critério em cada simulação. Na priorização dinâmica se avalia o estado de consumo dos BPs, definindo as novas prioridades como em termos da ordem crescente dos tempos neles restantes. A

priorização mista é utilizada como referência de comportamento. Sua utilização é iniciada segundo as prioridades fixas dos projetos, esse critério passa a ser dinâmico só depois de que algum BP atinja a área crítica de consumo.

A simulação está baseada na execução do mesmo programa utilizado no modelo anterior. É assim, que tanto o programa base, o programa projetado e a distribuição de probabilidade são os mesmos definidos para o modelo anterior.

- **Os parâmetros e cenários de simulação**. - Para isolar o efeito das priorizações de algum outro efeito que pudesse distorcer os resultados, desconsiderou-se a inserção de *buffers* de capacidade. Assim os diferentes cenários são definidos pelos valores das probabilidades de sucesso e falha que determinam a duração real das atividades e pela ativação do tipo de critério de priorização a utilizar.

  3. Os valores da probabilidade de sucesso (evolução de um dia da tarefa) e falha (não evolução da tarefa no dia) que define o comportamento aleatório da duração real das atividades: A implementação da distribuição binomial negativa por meio das probabilidades de sucesso e falha é igual ao modelo anterior. Neste caso se trabalha com um intervalo maior de probabilidade de atraso do projeto, isso, com a finalidade de observar o comportamento dos critérios de priorização num intervalo que abarque cenários pessimistas e otimistas.

  4. Os critérios de priorização de atividades: O critério de priorização de atividades é a lógica utilizada na escolha da seguinte atividade a executar. A “priorização fixa” toma essa decisão considerando unicamente as prioridades dos projetos definidas na etapa de planejamento, ou seja, levam-se em conta as características das atividades (se são críticas ou não) e se despreza o estado de consumo dos *buffers*.

     A “priorização dinâmica” escolhe a seguinte atividade a executar considerando tanto o nível crítico das atividades como o estado de consumo dos *buffers*. Desde que o modelo representa unicamente as atividades críticas, o critério da escolha é baseado somente no estado de consumo dos BP.

O conjunto de cenários avaliados está formado pela combinação de ambos os parâmetros. Para diferentes probabilidades de atraso do projeto (80%, 60%, 47%, 25%, 20%, 15%, 10% e 5%) são experimentados os três critérios de priorização (fixa, dinâmica e mista), estabelecendo 24 diferentes cenários.

- **A lógica do processo de execução. -** Ademais da lógica explicada na simulação anterior, o modelo adiciona um processo inicial que avalia o estado dos BP. Os projetos são organizados em forma decrescente, segundo o consumo relativo dos respectivos *buffers*. Para tal fim, utiliza-se o cálculo do consumo dos BP em função da sua dimensão planejada. O consumo relativo dos BP é dado por:

$$CR_i = \frac{consumo\_do\_BP_i}{duração\_planejada\_do\_BP_i}$$

Deste modo se assegura que as prioridades outorgadas aos projetos sejam efetivamente baseadas no estado crítico dos *buffers* do projeto.

Para implementar o critério de priorização mista se define que a partir do instante que algum BP atinge o consumo do 45% da sua capacidade, o critério de priorização deixa de ser fixo para adotar o critério de escolha de atividades dinâmica. A representação gráfica do modelo é mostrada na Figura 5.4

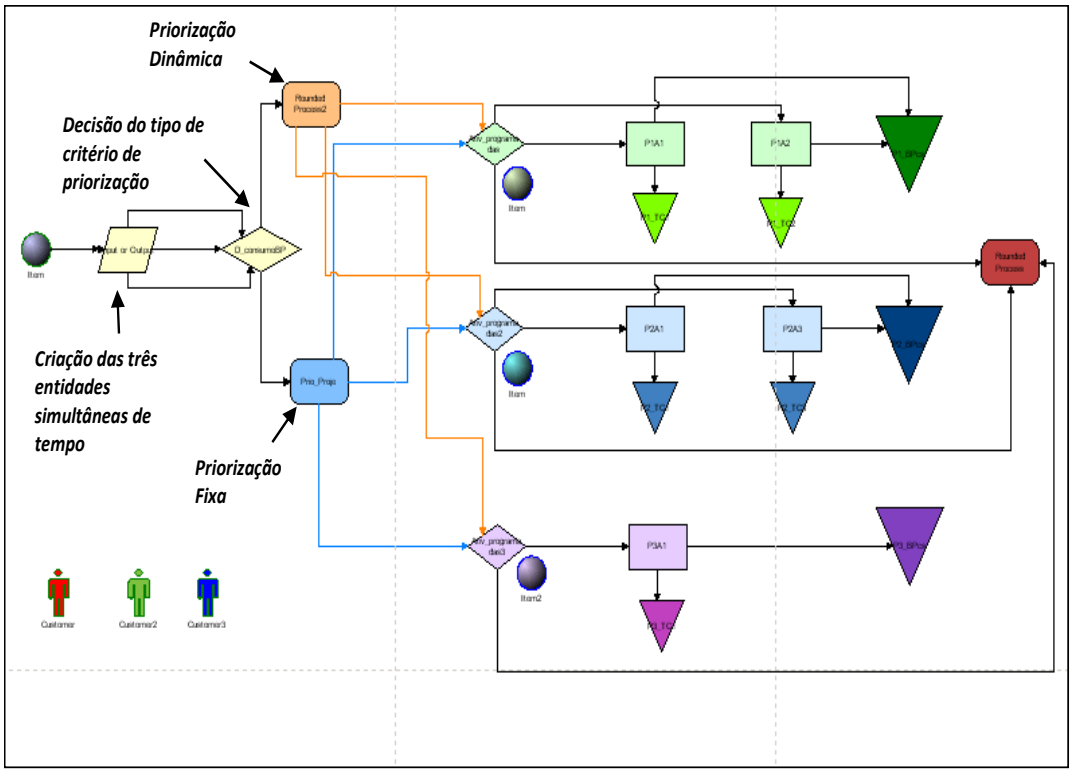

Figura 5.4: Modelo de simulação dos efeitos do critério de priorização de atividades

5.3. Análise dos resultados

5.3.1. Efeitos dos BC

Antes de apresentar a análise dos resultados é importante lembrar que o experimento é baseado na programação "híbrida" que insere os BC dentro do programa projetado, mantendo-se intacto o programa base. Isso ocasiona que os BP tenham um consumo inicial equivalente ao tamanho do BC. É assim que a análise dos efeitos do BC é avaliada em função da quantidade consumida do BP.

5.3.1.1. Dimensionamento dos BC em função do BP

No início da pesquisa, definiu-se que os BC agem como um complemento dos BP, adicionando a segurança que possa absorver os atrasos causados pelo compartilhamento de recursos entre projetos do mesmo sistema. A partir de essa definição, surge a necessidade de esclarecer se, contrariamente ao estabelecido

nas ferramentas de software, os BC teriam que ser dimensionados em base às necessidades específicas de cada projeto e não como uma porcentagem fixa da carga do recurso gargalo.

No capítulo IV se concluiu que quando o consumo do BP ocasionado pela inserção do BC é pequeno, em relação à vizinhança do ponto de início do BP, é um indício de que o BC estaria sub-dimensionado. Similarmente, quando a inserção do BC consume totalmente ou excede a capacidade do BP, é um indicio de que este estaria super-dimensionado.

Com essa lógica, estabelece-se que o ponto de referência mínimo de análise está no ponto inicial das curvas de consumo de todos os casos de probabilidade de atraso do projeto. Esse ponto inicial corresponde ao consumo dos BP quando o tamanho do BC é zero, ou seja, que é o consumo ocasionado unicamente pelos atrasos aleatórios das atividades. Deste modo, tamanhos de BC que gerem consumos do BP próximos ao valor de referência são desconsiderados por não cobrir as necessidades de segurança do projeto.

O ponto de referência que indica o super-dimensionamento dos BC é aquele, que ao estar mais longe do ponto inicial, marca o início de uma tendência crescente. Tamanhos de BC que gerem pontos posteriores a esse, também são desconsiderados por exceder a necessidade de segurança do projeto.

Os gráficos do comportamento dos três BP na presença de diferentes níveis de probabilidade de atraso e de diferentes tamanhos de BC são mostrados nas Figura 5.5 a Figura 5.8.

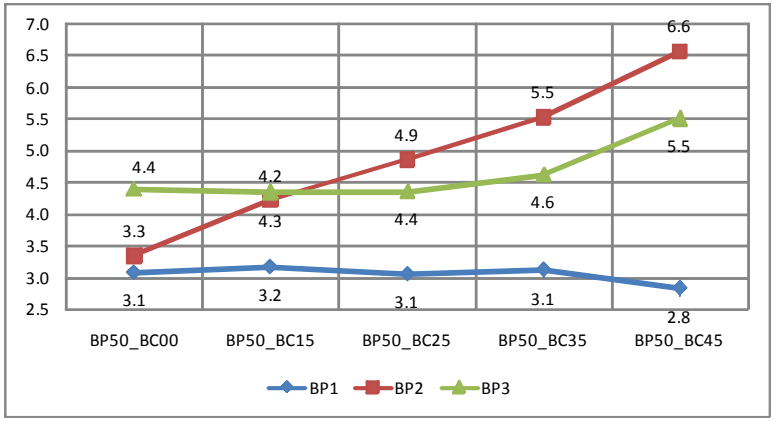

Figura 5.5: Consumo de BP - 25% prob. Atraso

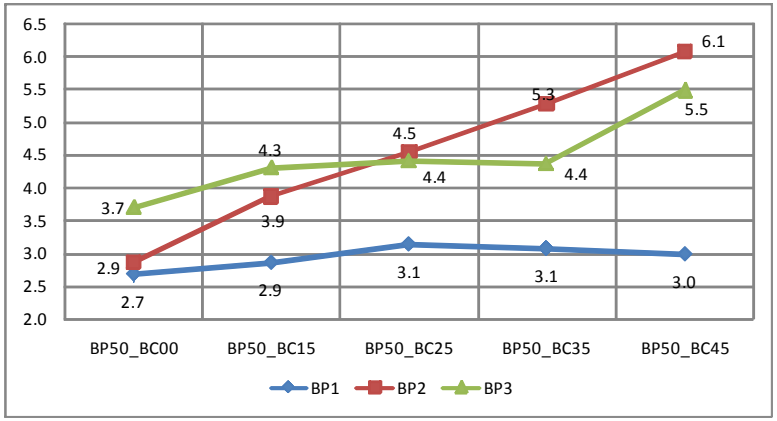

Figura 5.6: Consumo de BP - 20% prob. Atraso

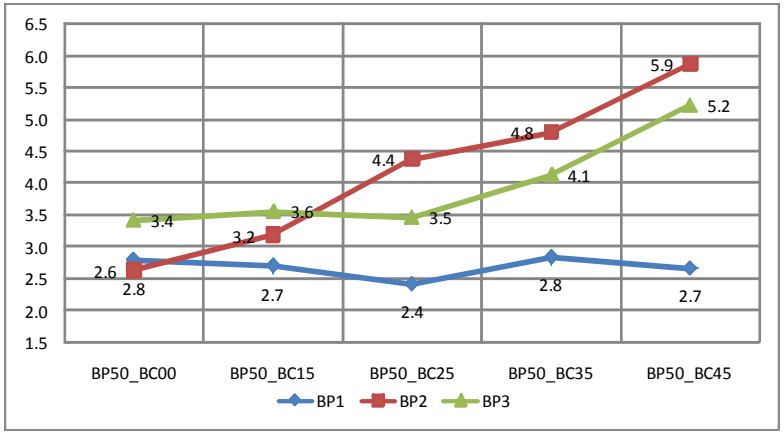

Figura 5.7: Consumo de BP - 15% prob. Atraso

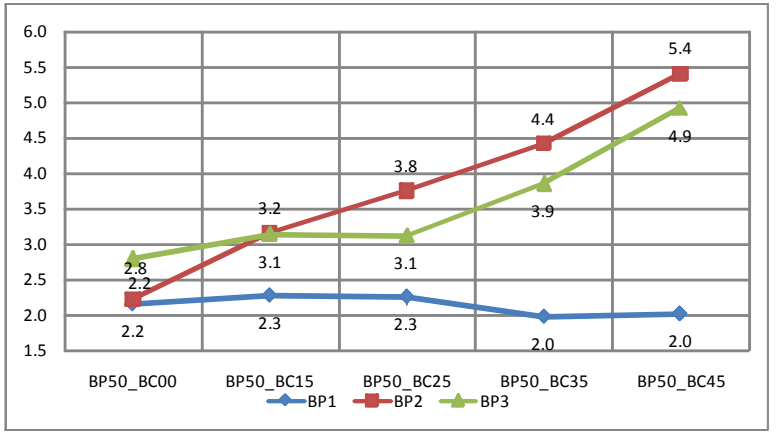

Figura 5.8: Consumo de BP - 10% prob. Atraso

O projeto um (linha azul), ao ser sempre o projeto de maior prioridade, não apresenta atrasos causado por conflitos de recursos nem pelo efeito da inserção de BC. A utilização da mesma série de variações aleatórias em todos os cenários faz com que a curva de consumo do BP1 seja quase constante.

O projeto dois, em todos os casos, apresenta uma curva linearmente crescente (vermelha), indicando que as inserções dos BC atingem rapidamente o ponto de sobre-dimensionamento. Esta condição não permite uma definição clara do tamanho de BC adequado, não obstante, pode-se afirmar que a necessidade de segurança extra do projeto é mínima ou nula.

O projeto três apresenta um comportamento totalmente distinto. Na Figura 5.5 o consumo gerado pelos BC de 15%, 20% e 25%, desenham uma curva quase constante ao redor do ponto de referência. O BC dimensionado a 35% é o primeiro ponto que se desloca da vizinhança do ponto inicial de referência, indicando que este seria o dimensionamento mais adequado para as condições do projeto.

Para o caso de 20% de probabilidade de atraso dos projetos (Figura 5.6), o BC mais adequado oscila entre tamanhos de 25% e 35%. Para o caso de 15% e 10% de probabilidade de atraso, o tamanho de BC mais adequado oscila entre 15 e 25%. Repare-se que com a diminuição da probabilidade de atraso do projeto a distância entre os primeiros pontos da curva define uma tendência constante, similar à gerada pelas variações aleatórias do projeto um. Isso indicaria que a necessidade de adicionar segurança ao projeto é mínima ou nula.

Esta análise, ademais, de dar indícios para a determinação dos BC, permite esclarecer que as características de cada projeto e o grau de dependência que existe entre os diferentes projetos do sistema, fazem que a necessidade de segurança extra dos projetos seja distinta. É assim, que projetos definidos com a menor prioridade, ao depender de um maior número de projetos, não podem utilizar a mesma porcentagem de dimensionamento de BC do que aqueles projetos de maior prioridade. Se esse fosse o caso, certamente o projeto incorreria atrasos. Da mesma maneira, ao dimensionar todos os BC com a porcentagem definida para o projeto menos prioritário, os projetos de prioridade intermediária serão excessivamente deslocados, prejudicando sua competitividade no mercado e provocando o desaproveitamento da disponibilidade dos recursos.

5.3.1.2.
Análise das causas de atraso das atividades

A análise é completada com a decomposição das causas que ocasionam o atraso das atividades. O atraso pode ser dado devido às condições aleatórias do experimento ou à falta de disponibilidade de recursos necessários para executá-las. A última causa é a que deve ser mitigada através da inserção dos BC.

Comparando o efeito que cada tamanho de BC têm na mitigação de atrasos causados por falta de recurso e o atraso efetivo dos projetos, determina-se claramente o tamanho de BC que cada projeto necessita. Assim, também se permite o entendimento das conseqüências que essa mitigação tem no cumprimento da data prometida dos projetos.

As figuras Figura 5.9 a Figura 5.16 mostram, para cada projeto e cada probabilidade de atraso, a relação entre o atraso efetivo dos projetos (dado pelo consumo do BP sem considerar o consumo ocasionado pela inserção dos BC) e o atraso ocasionado por falta de recursos. Ambos os eixos estão descritos em unidades gerais de tempo (*t*).

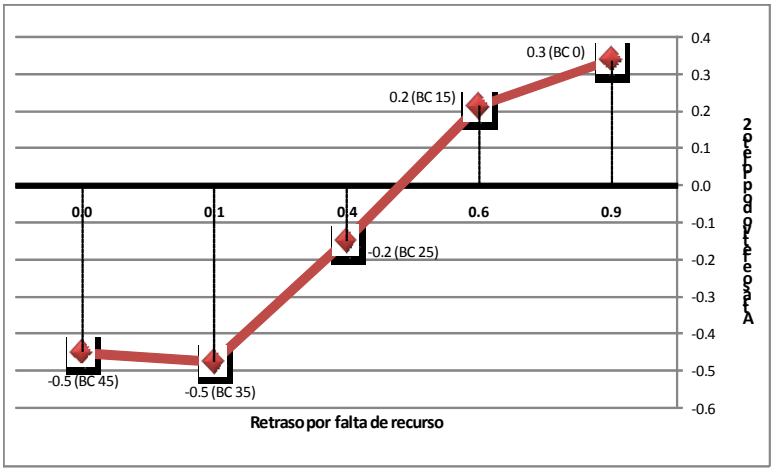

Figura 5.9: Mitigação dos atrasos por falta de recurso
(Projeto dois - 25% de probabilidade de atraso)

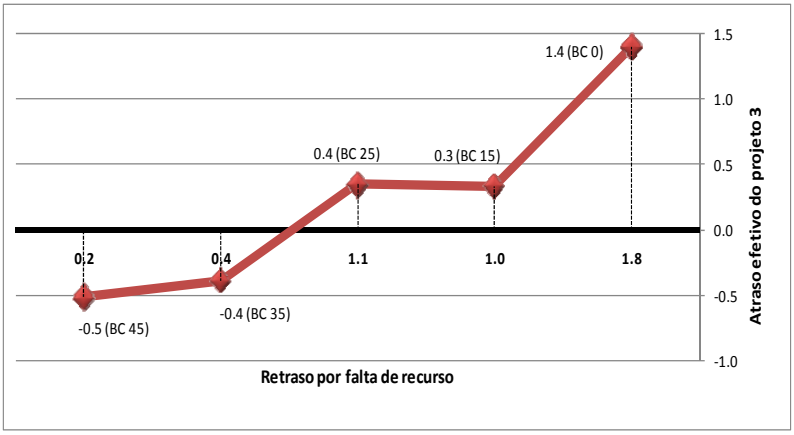

Figura 5.10: Mitigação dos atrasos por falta de recurso (Projeto três - 25% de probabilidade de atraso)

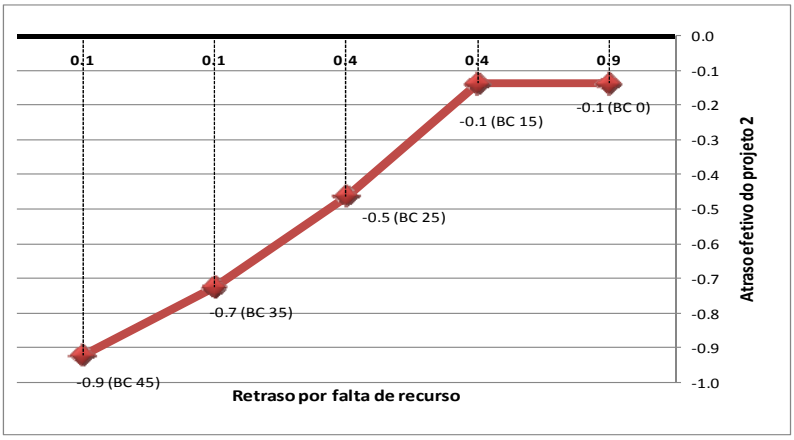

Figura 5.11: Mitigação dos atrasos por falta de recurso (Projeto dois - 20% de probabilidade de atraso)

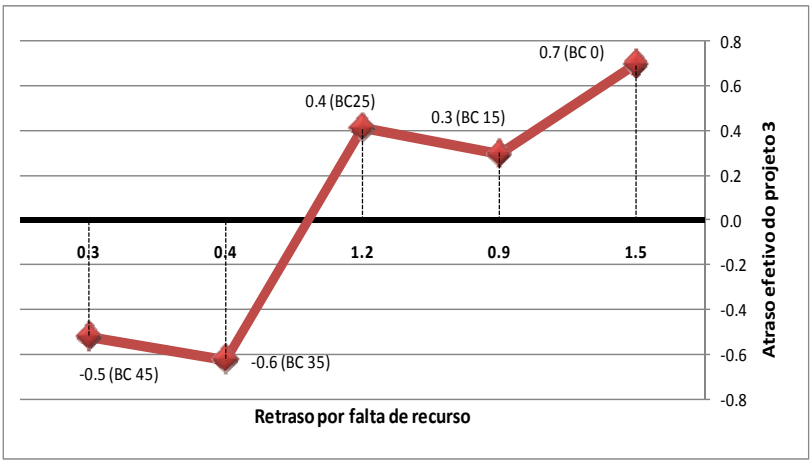

Figura 5.12: Mitigação dos atrasos por falta de recurso (Projeto três - 15% de probabilidade de atraso)

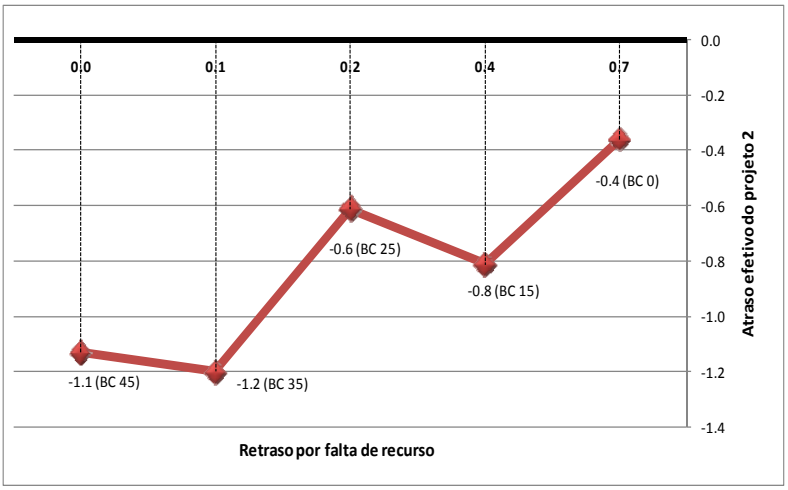

Figura 5.13: Mitigação dos atrasos por falta de recurso
(Projeto dois - 15% de probabilidade de atraso)

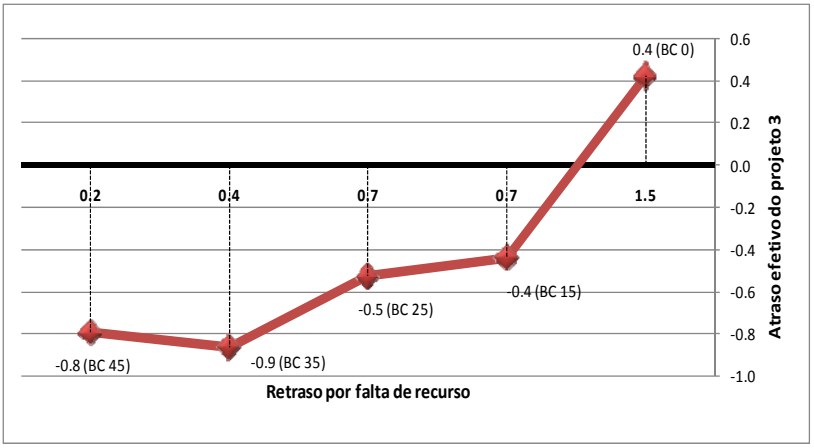

Figura 5.14: Mitigação dos atrasos por falta de recurso
(Projeto três - 15% de probabilidade de atraso)

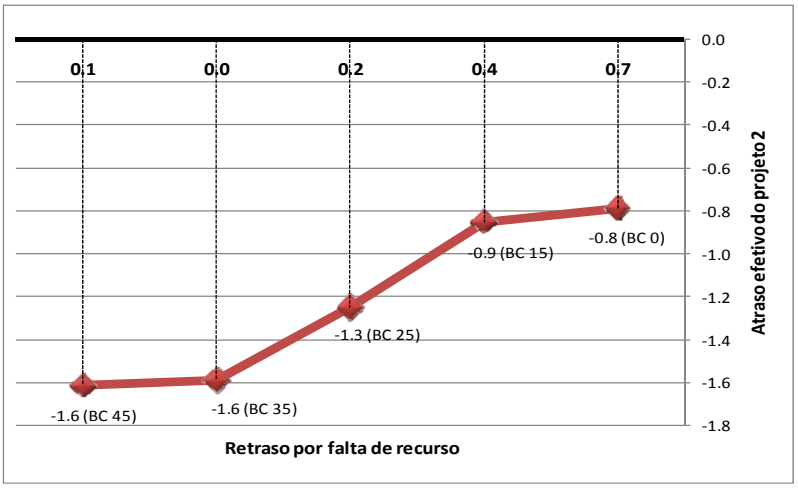

Figura 5.15: Mitigação dos atrasos por falta de recurso
(Projeto dois - 10% de probabilidade de atraso)

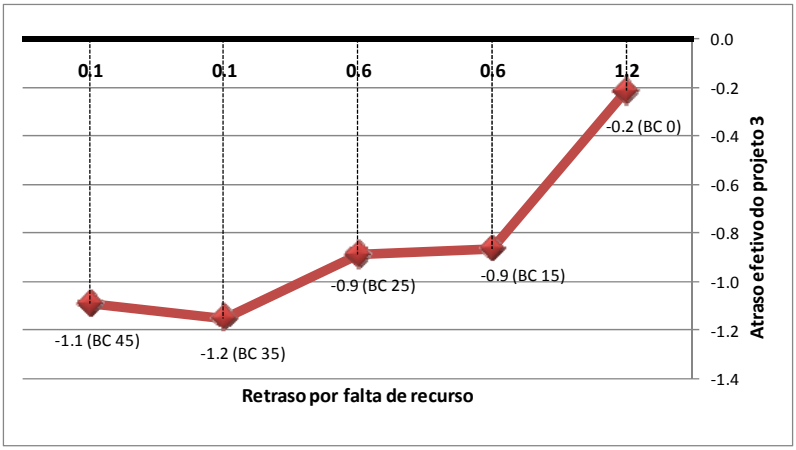

Figura 5.16: Mitigação dos atrasos por falta de recurso
(Projeto três - 10% de probabilidade de atraso)

A Figura 5.9 mostra que, efetivamente como a análise anterior indicou, o projeto dois tem uma necessidade ínfima de aumentar sua segurança. Com 25% de probabilidade de atraso do projeto, os atrasos dos projetos são descritos por frações menores à unidade mínima de medida. Contudo, a determinação de um BC de 25% asseguraria o término do projeto antes da data prometida.

Probabilidades de atraso do projeto menores de 25% não requerem a inserção do BC para o projeto 2 (Figura 5.11, Figura 5.13, Figura 5.15). Repare que em todos os casos do projeto dois a inserção de BC de tamanhos maiores que 25% não geram efeitos significantes na redução do tempo atrasado por falta de recursos (eixo horizontal).

Como se determinou na análise prévia, o projeto três apresenta uma necessidade maior de segurança que o proteja das interdependências com os projetos de maior prioridade. Note-se na Figura 5.10 como a inserção de um BC dimensionado em 15% reduz o tempo de atraso do projeto de 1.4 a 0.3 unidades de tempo, o que é equivalente à redução de 1.8 a 1.0 unidades de tempo atrasadas por causa da falta de recursos. O efeito de mitigação dos atrasos por falta de recurso diminui gradualmente até a inserção de BC dimensionados a 35%. Nesse ponto se atinge o término do projeto antes da data prometida. Incrementos maiores que 35% não geram efeitos significantes na redução de atrasos por falta de recursos.

A necessidade de segurança extra do projeto três, somente muda quando a probabilidade de atraso é definida em 15% (Figura 5.12). Nesse cenário os atrasos

do projeto são menores à unidade de tempo e a inserção de um BC de 15% faz que o projeto seja terminado antes da data prometida. Probabilidades de atraso menores que 15% não requerem a inserção do BC.

5.3.1.3. "Trade-off" entre mitigação de atrasos ou diminuição do "makespan"

Na análise do efeito de mitigação do BC se utilizou como referência o tamanho de BC que atinge o término antecipado da data prometida. Certamente, em termos de mitigação, esses tamanhos de BC seriam adequados para as características do projeto e do cenário analisado. Contudo, vale à pena lembrar que a data prometida atingida é aquela data resultante da deslocação do início dos projetos. Ou seja, uma data que inclui a inserção dos BC sugere o aumento do *makespan* dos projetos, situação que, poderia resultar desfavorável para algumas empresas.

A inserção dos BC leva consigo a decisão do intercâmbio (*trade-off)* entre: a obtenção de programas que mitiguem consideravelmente os atrasos causados por falta de recursos, criando projetos mais estáveis; e a obtenção de programas que mesmo correndo algum risco de atraso, criam projetos de menor permanência no sistema, aumentando a taxa de rotação de projetos.

A análise contrapõe os atrasos ocasionados pelo compartilhamento de recursos entre projetos (atrasos por falta de recurso) com o tempo inserido pelo BC na deslocação dos projetos (aumento do *makespan*). Ambas as medidas são expressas como uma percentagem da duração enxuta dos projetos. Entendendo-se como duração enxuta àquela que não considera os BP.

È assim que desde a Figura 5.17 à Figura 5.20 se mostra o incremento da porcentagem do *makespan* e a diminuição da porcentagem do atraso ocasionado por falta de recursos, para cada tamanho de BC. O cálculo de ditas variações é sempre feito em relação ao ponto de início.

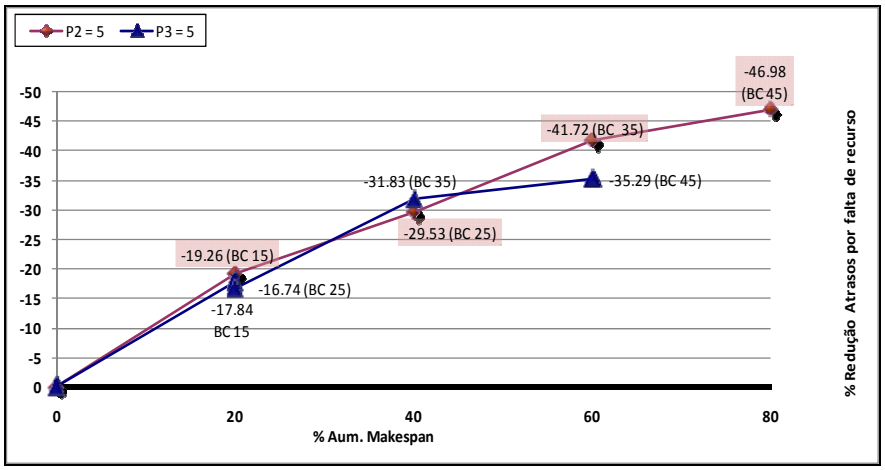

Figura 5.17: T*rade-off* entre *makespan* e redução de atrasos por falta de recurso (25% probabilidade de atraso)

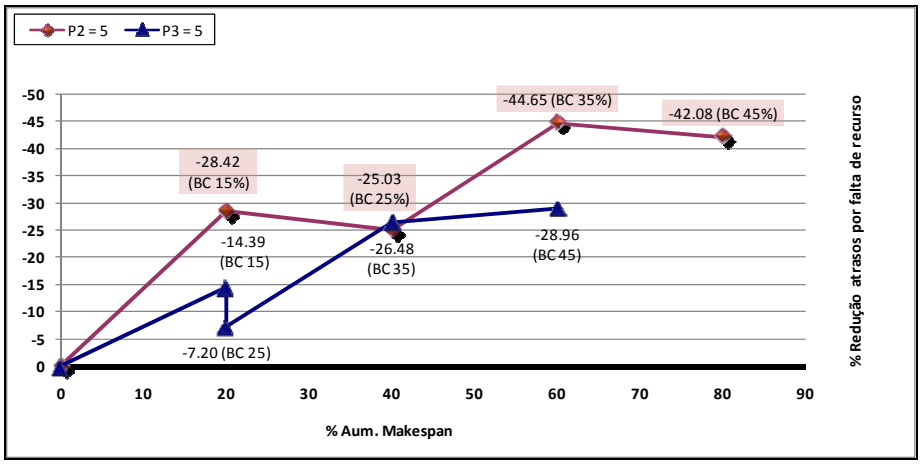

Figura 5.18: T*rade-off* entre *makespan* e redução de atrasos por falta de recurso (20% probabilidade de atraso)

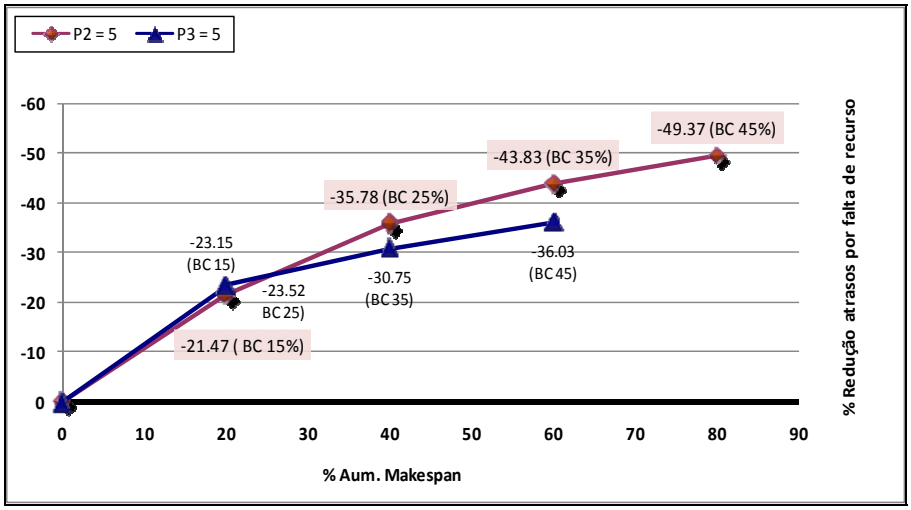

Figura 5.19: T*rade-off* entre *makespan* e redução de atrasos por falta de recurso (15% probabilidade de atraso)

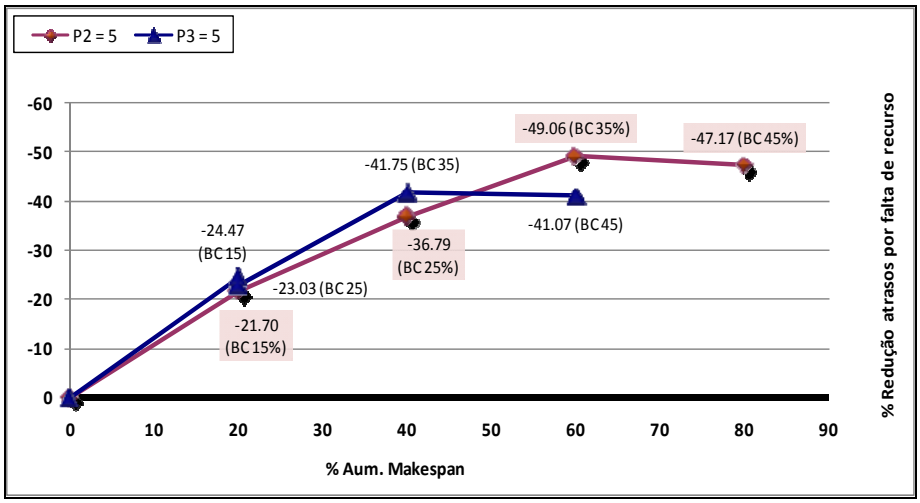

Figura 5.20: T*rade-off* entre *makespan* e redução de atrasos por falta de recurso (10% probabilidade de atraso)

Repare-se que a porcentagem de tempo que os BC incrementam no *makespan* não é a mesma em todos os projetos. Lembre-se que o BC é determinado em base à capacidade do recurso gargalo. Ou seja, para obter um BC dimensionado a 25%, faz-se o tempo ocupado pelo gargalo nas atividades previas da "seqüência crítica" (as que pertencem a outro projeto), igual a 75%.

O projeto dois tem BC equivalentes a: 15%= 1t, 25% = 2t, 35% = 3t e 45% = 4t.
O projeto três tem BC equivalentes a: 15%= 1t., 25% = 1t , 35% = 2t e 45% = 3t.

Como o modelo não contempla atrasos em frações de tempo, o cálculo do BC considera arredondamento superior para qualquer excesso do valor inteiro.

A combinação do *makespan* e do nível de compartilhamento de recursos que possa ser definida como "apropriada", carece de regras e patrões. Cada situação, incluso cada projeto imerso dentro de um mesmo cenário, tem diferentes características e necessidades que determinam um conceito particular do "apropriado".

Repare-se como para o projeto dois (Figura 5.17) o incremento de 20% no *makespan* gera a maior **taxa de diminuição** dos atrasos por falta de recurso. Conforme vimos na análise anterior, incrementos do *makespan* maiores de 20% (ou BC maiores de 15%) ocasionam datas de entrega antecipadas.

Não obstante, o objetivo da empresa poderia ser justamente a definição da data de entrega mais antecipada. Em situações onde o projeto não tem urgência de implementação, este poderia ser programado longe dos projetos conflitantes, de tal maneira, que sua duração efetiva seja menor. Se esse fosse o caso, escolher-se-ia o BC de 35%, *buffer* que gera a maior taxa de diminuição de atrasos em relação aos incrementos do *makespan.*

No projeto três a maior taxa de diminuição de atrasos é definida pelo incremento de 40% do *makespan* (ou BC de 35%). Contudo, o projeto poderia ser caracterizado por não permitir grandes postergações da data de início e ser mais permissível com extensões do término do trabalho. Essa situação é comum em projetos de manutenções de chão de fábrica. Se esse fosse o caso, teria que se escolher um BC de 15%, que é o dimensionamento que produz o menor adiamento da data de início, mesmo sacrificando o cumprimento da data de entrega prometida.

A Figura 5.18 mostra como a aleatoriedade na execução das atividades dos projetos predecessores altera o efeito de mitigação dos BC, definindo um comportamento singular da curva de *trade-off* entre o *makespan* e a diminuição de atrasos por falta de recurso.

Tal efeito é mais bem explicado na abstração da Figura 5.21. A área conflitante do projeto um está formada pelas atividades que ademais de ter um alto compartilhamento de recursos com o projeto dois, também concentra a maior quantidade de atrasos aleatórios do projeto. Nesse contexto, a inserção de BC intermediários, como o caso de 25%, faz que as atividades mais conflitantes do projeto dois (marcadas em azul solido) sejam programadas durante mais tempo dentro da área de conflito do projeto um. Isso ocasiona um menor efeito de mitigação de atrasos do que com a inserção de *buffers* menores, tal e como é visto na análise dos projetos dois e três da Figura 5.18.

Deste modo, conclui-se que o desempenho esperado dos BC depende fortemente das características do cenário no qual são inseridos. É assim, que inserção de BC maiores não significa que necessariamente se produza um melhor resultado na mitigação dos atrasos por falta de recurso. Especialmente, quando o nivelamento da carga de trabalho é baseado num único recurso gargalo.

Obviar os efeitos que os cenários têm sobre o comportamento dos BC pode levar a dimensionar BC que criem conflitos de recursos tão importantes, que acabem gerando um novo gargalo.

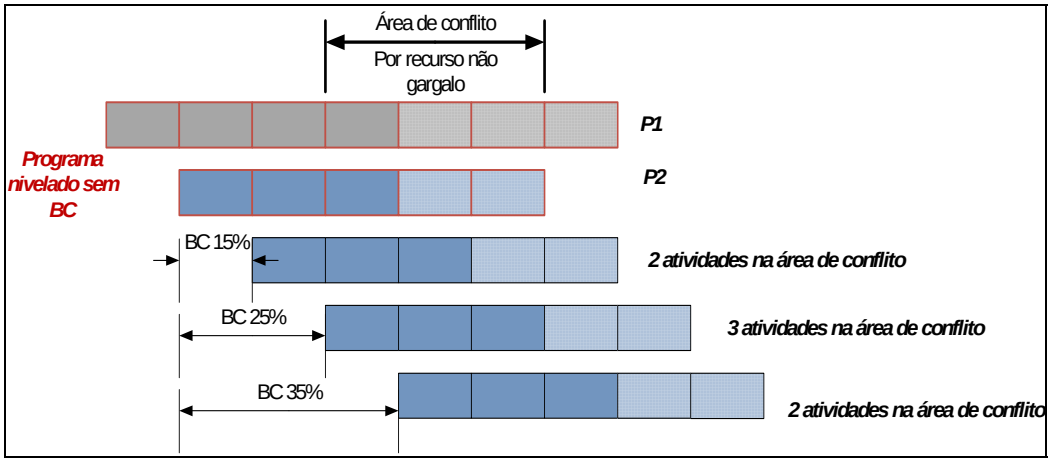

Figura 5.21: Caso especial dos efeitos do BC

5.3.2. Efeitos dos critérios de priorização

Ao estudar o comportamento das abordagens de aplicação durante a etapa de execução (capítulo IV), provocaram-se atrasos em certas atividades que, de acordo com a lógica do programa, permitiam gerar situações especificas de análise. A partir dessas situações, observou-se que as decisões de execução das atividades são baseadas nas prioridades definidas inicialmente para os projetos. Mostrando-se que ao dar prioridade de execução às atividades que correspondiam aos BP mais consumidos, poder-se-ia produzir um resultado melhor.

Esse critério de priorização foi também utilizado como regra de execução da CC/BM na implementação do experimento desenhado por Cohen *et al*. (2004). No artigo se compara o desempenho da metodológica CC/BM em relação a outras regras de programação de múltiplos projetos. Concluindo-se que o desempenho da metodologia, nas condições avaliadas, não supera o resultado obtido através de algumas das regras observadas.

Desta maneira, com base nas observações da análise anterior e na existência de referências literárias que sustentem a validade dos fenômenos observados, levanta-se a hipótese de que o uso de prioridades de execução dinâmicas faz com que os projetos possam ser terminados em menor tempo.

Para poder responder à dito questionamento, analisa-se os resultados do experimento em função do cumprimento da data de entrega prometida dos projetos (incluindo o BP). Os critérios de priorização são avaliados através de uma série de cenários definidos por probabilidades de atraso do projeto entre 5% e 80%.

As Figura 5.22 e Figura 5.23 mostram as curvas que descrevem o efeito que os critérios de priorização fixa, dinâmica e mista tem no sistema.

O eixo vertical expressa o atraso dos projetos, no qual se interpretam os valores negativos como se fossem términos anteriores à data de entrega prometida. O eixo horizontal indica a probabilidade de atraso do projeto com que foram geradas as durações das atividades.

Para facilitar a análise, o gráfico é apresentado em duas partes. Na primeira parte se mostram os cenários altamente pessimistas, descritos por probabilidades de 80%, 60% e 47% (Figura 5.22). Na segunda parte se mostram os cenários descritos por probabilidades de 25%, 20%, 15%, 10% e 5% (Figura 5.23).

O elevado número de atrasos totais do sistema era de se esperar. Esses atrasos são ocasionados tanto pela alta probabilidade de atraso imersa nos cenários, como pela ausência de BC nos projetos. Contudo, o uso da priorização fixa produz um resultado ligeiramente melhor em situações onde a probabilidade média de atraso é superior a 60%.

Conforme diminui a probabilidade de atraso, as curvas de prioridades fixas e dinâmicas tendem a coincidir, obtendo o mesmo resultado (ou resultados muito próximos) na consecução da data de término prometida.

Com base nesses resultados, certamente, não pode se afirmar que o uso de prioridades dinâmicas melhora o desempenho da programação de projetos. Pelo contrario, o desempenho dos projetos através da utilização de prioridades dinâmicas tende a ser pior do que o resultado obtido através da manutenção das prioridades fixas.

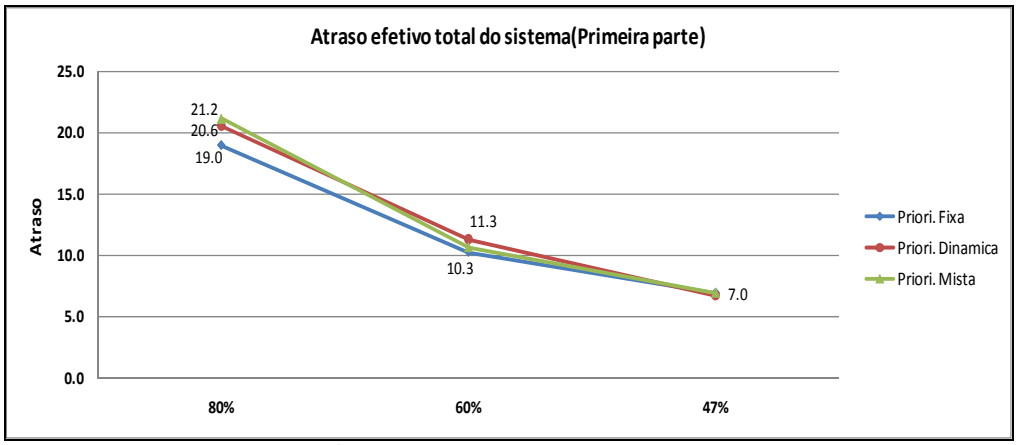

Figura 5.22: Efeitos dos critérios da priorização no sistema

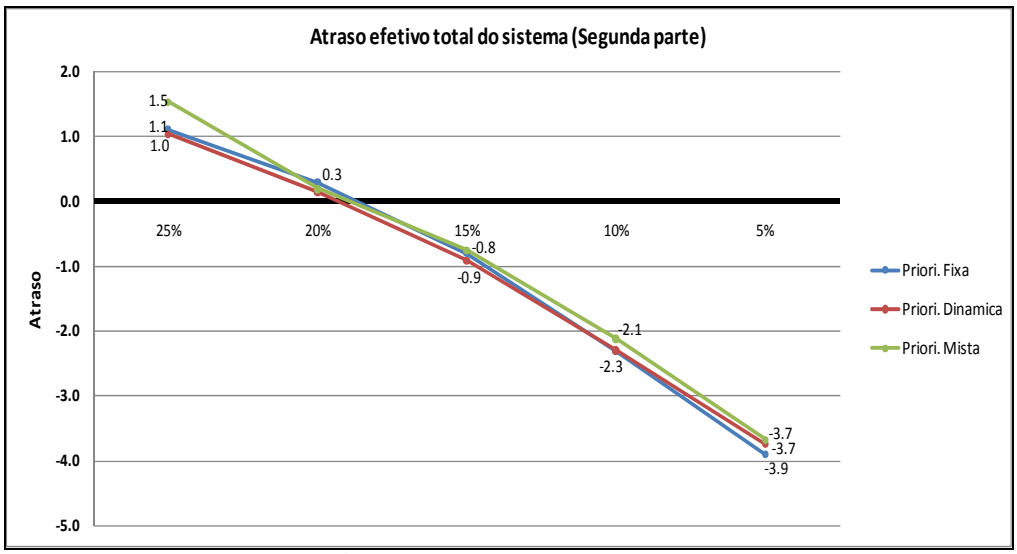

Figura 5.23 : Efeitos dos critérios da priorização no sistema

Para entender o porquê do critério de priorização dinâmica não gerar os resultados esperados, compara-se os efeitos que os critérios de priorização têm sobre cada projeto num ambiente de elevadas probabilidades de atraso.

Quando os projetos são executados segundo suas prioridades definidas inicialmente (priorização fixa), as atividades do projeto um são sempre as primeiras em serem executadas. Ao aplicar o critério de priorização dinâmica ou mista, as datas de término do projeto um acabam sendo muito mais atrasadas do que o adiantamento ganho nas execuções dos projetos dois e três. (Figura 5.24, Figura 5.25 e Figura 5.26)

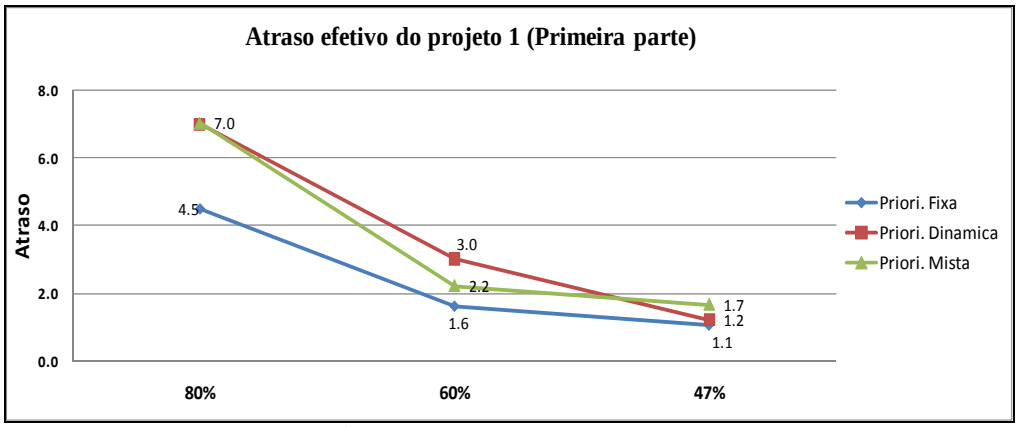

Figura 5.24 : Efeitos dos critérios da priorização no projeto 1

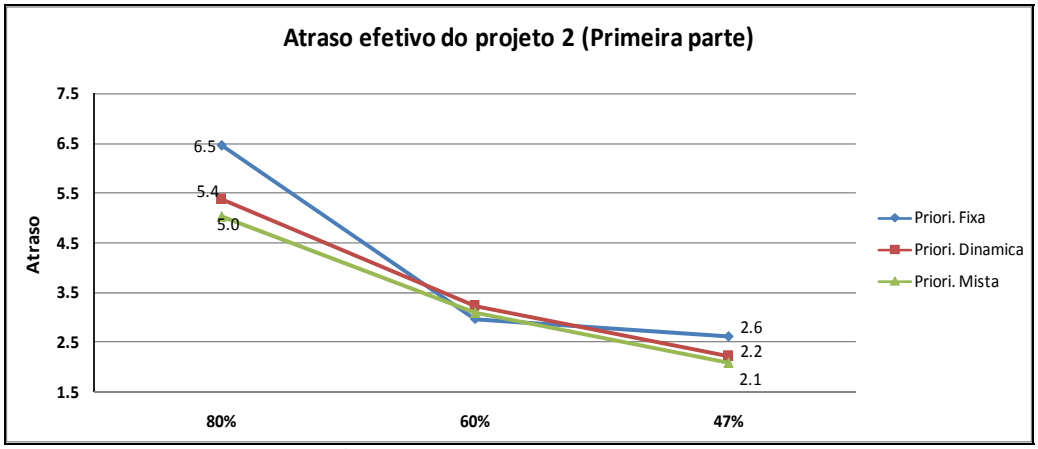

Figura 5.25 : Efeitos dos critérios da priorização no projeto dois

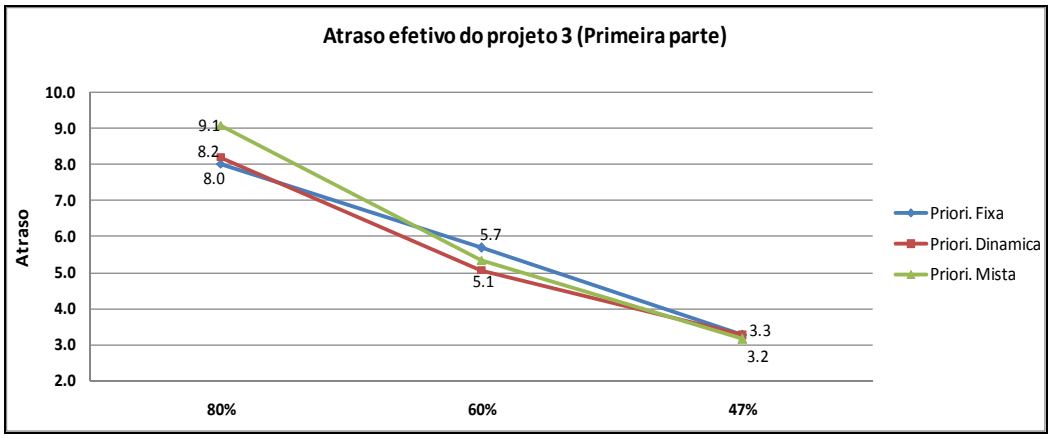

Figura 5.26: Efeitos dos critérios da priorização no projeto três.

Esse fenômeno pode ser explicado pelos conceitos de variações de *causas comuns* e variações de *causas especiais*. Segundo Leach (2005) as variações são de causas comuns quando a fonte da variação é o próprio sistema, e são de causas

especiais quando as variações são provocadas por eventos provenientes fora do sistema.

É na presença dessas últimas variações que a gerência de projetos (e de qualquer sistema produtivo) deve tomar ações corretivas. Leach (2005) enfatiza que o fato de tratar as variações comuns do sistema como se fossem de causas especiais, gera o incremento da variância total do sistema, e conseqüentemente seu atraso.

Na execução de múltiplos projetos, os atrasos ocasionados tanto pela aleatoriedade da duração das atividades como pelo compartilhamento de recursos são variâncias comuns do sistema. Ditas variações, são mitigadas através da inserção dos diferentes *buffers* durante a etapa de planejamento do programa. É assim que, a tentativa de melhorar os resultados da execução do sistema, dando prioridade às atividades dos projetos mais críticos (BP mais consumido) ocasiona o desequilíbrio do programa. Gerando-se uma situação caótica, na qual os recursos vão de um projeto a outro criando um tipo de múltiplas tarefas (*multi-tasking*).

A razão pela qual tais efeitos não foram percebidos na análise anterior, é que as variações de execução foram geradas segundo uma lógica similar aos casos de probabilidade de atraso menores de 25%. A simplicidade do problema estudado deixa identificar os efeitos de usar prioridades dinâmicas, somente na presença de altas probabilidades de atraso.

Contudo, a negatividade dos efeitos incrementa-se conforme a complexidade do problema aumenta. Do mesmo modo, as conseqüências empioram quando se agregam BC na análise.

6
Conclusões e sugestões

Esta dissertação tratou da gestão de múltiplos projetos por meio da metodologia da CC/BM. Concentrou-se na análise e entendimento dos efeitos que a implementação usual do dimensionamento dos *buffers* de capacidade – BC e dos critérios de priorização de atividades têm sob os objetivos da empresa e dos próprios projetos.

As abordagens para programação conjunta dos projetos foram as definidas por Newbold (1998), a saber: "Todos juntos", "Projetos sucessivos" e "Recurso estratégico ou gargalo". O estudo permitiu identificar, primeiramente, que a abordagem de "todos juntos", ao utilizar um projeto artificial que engloba todas as atividades dos projetos sem fazer distinção explicita entre elas, elimina a noção de projeto e, assim, não considera a complexidade da interação de múltiplos projetos. A estrutura da rede resultante não é representativa dos objetivos de cada projeto e certamente limita a eficiência das funções de proteção e controle dos *buffers* de projeto, gerando um efeito negativo na estabilidade da rede e na qualidade de informação que esta oferece. É assim que as informações para decisões gerenciais de programação passam a ser incongruentes e confusas nas etapas de planejamento e de execução, mesmo em condições normais e, ainda mais, quando o sistema é afetado por atrasos

As abordagens dos "projetos sucessivos" e do "recurso estratégico ou gargalo", ao serem desenvolvidas dentro do programa máster, reduzem as deficiências da abordagem anterior auxiliando corretamente a gerência de múltiplos projetos. Contudo, o programa perde eficiência quando as atualizações do sistema implicam reprogramar os projetos. Isso origina a perda de informação do estado de consumo dos *buffers* e, conseqüentemente, a perda de controle na consecução dos objetivos traçados.

A abordagem de "projetos sucessivos" nivela todos os recursos e não permite sobrecargas, fazendo com que ela seja mais eficiente em sistemas de projetos simples e com baixo nível de compartilhamento de recursos. Para casos

de projetos mais complexos é recomendada a utilização da abordagem do "recurso estratégico", que mesmo permitindo sobrecargas no sistema permite gerir a complexidade por meio da inserção de *buffers* de capacidade.

A partir da análise do comportamento do programa máster, reparou-se que as programações projetadas dos múltiplos projetos não consideram o estado de consumo dos *buffers* de projeto nem fazem diferença entre atividades críticas e não críticas na escolha das atividades a executar. A ferramenta de *software* utilizada para implementação escolhe tais atividades segundo as prioridades definidas inicialmente para cada projeto. Ainda que a prática mais usual da metodologia defenda tal proceder, as observações feitas durante os ensaios inicialmente realizados deram indícios de que o uso de prioridades dinâmicas poderia dar melhores resultados.

Para melhor entender os fenômenos observados nesses ensaios, recorreu-se a uma simulação probabilística do problema-teste de múltiplos projetos. Essa análise indicou que o uso da priorização fixa produz melhores resultados em situações onde a probabilidade média de atraso é superior a 60% e que conforme diminui a probabilidade de atraso, as curvas de prioridades fixas e dinâmicas tendem a coincidir, obtendo o mesmo resultado (ou resultados muito próximos) no que tange a data de término prometida. O fato de o desempenho dos projetos através da utilização de prioridades dinâmicas piorar o resultado, é explicado pela diferença existente entre as variações de **causas comuns** e variações de **causas especiais**.

Os atrasos ocasionados tanto pela aleatoriedade da duração das atividades como pelo compartilhamento de recursos são causas comuns do sistema relacionadas às imprecisões na sua concepção. Tais variações são mitigadas através da inserção dos diferentes *buffers* durante a etapa de planejamento do programa. Variações provocadas por eventos não considerados na concepção do sistema tais como falhas no desempenho de equipamentos e de recursos utilizados, são consideradas como causas especiais que podem e precisam ser corrigidas e controladas. Tentativas de melhorar os resultados da execução do sistema, alterando as prioridades das atividades dos projetos mais críticos (BP mais consumido) é equivalente a tratar as variações de causas comuns como se fossem especiais, aumentando a variação do sistema e ocasionando o desequilíbrio

do programa. Isso gera uma situação de instabilidade caótica, na qual os recursos vão de um projeto a outro criando um tipo de múltiplas tarefas (*multi-tasking*).

Esse efeito negativo tende a agravar-se conforme a complexidade do problema aumenta. Similarmente, as conseqüências de desequilíbrio e a desorganização pioram quando se agregam BCs na análise.

Reparou-se que, na ferramenta de software utilizada, não é permitido dimensionar individualmente os BCs para cada projeto, definindo-se como padrão o mesmo valor porcentual para todos os BC. Os casos estudados deram indícios de que o efeito desse dimensionamento generalizado afetava negativamente os objetivos de alguns projetos e que a necessidade dos BC dependia tanto da carga do recurso gargalo como das características particulares de cada projeto.

Por meio do modelo de simulação probabilística experimentaram-se cenários que, a partir de um programa "híbrido", criado para avaliar o efeito de dimensionamento dos BCs com base no consumo dos BPs, testou-se o efeito de uma serie de tamanhos de BC. Os resultados da análise corroboraram a hipótese de que as características de cada projeto e o grau de dependência entre os diferentes projetos do sistema, fazem com que as necessidades de segurança extra dos projetos sejam distintas. Assim, projetos definidos com menor prioridade, por dependerem de um maior número de outros projetos, devem utilizar BCs proporcionalmente maiores do que aqueles de maior prioridade.

Na análise do *trade-off* entre estabilidade e makespan explicou-se o efeito da inserção dos BCs sobre o aumento do *makespan* dos projetos. A robustez que os BC dão ao sistema, assegurando que os projetos possam ser terminados na data prometida ou até antes dela, é baseada num adiamento planejado do termino do projeto. Quer dizer, a robustez do programa é obtida a expensas de datas prometidas mais tardias. A análise dessa relação pode ser feita através da comparação da redução de atrasos por falta de recurso e o aumento dos *makespans* dos projetos simulados com os diferentes tamanhos de BC. Observou-se que essa relação depende fortemente das características dos projetos e dos cenários no qual são inseridos. Ou seja, o tamanho relativo de BC que gera a maior taxa de redução de atrasos não é o mesmo para todos os projetos considerados no mesmo cenário (i.e. com mesmas probabilidades de atraso nas atividades). Da mesma maneira, esse tamanho do BC ("ótimo") para um mesmo projeto difere para os diferentes

cenários avaliados. Corroborando-se que a determinação do BC não pode ser satisfatória com uma única regra general sem levar em conta as particularidades de cada projeto e as características do ambiente onde ele é desenvolvido. A análise marginal da relação entre a diminuição de atraso e o aumento do makespan dos projetos permite entender que aumentos no tamanho do BC provocam inicialmente uma elevada diminuição de atrasos, justificando a postergação da data de termino do projeto. Não obstante, essa taxa de redução tende a decrescer com o aumento do tamanho do BC, podendo chegar a ser negativa. Ou seja, BCs muito grandes podem ocasionar conflitos entre outros recursos de alto nível de compartilhamento mais adiante no projeto, podendo atrasar a execução das atividades, fazendo com que o término dos projetos seja mais tarde. A inserção de um BC super-dimensionado pode criar novos gargalos, que dependendo da complexidade do conflito de atividades no uso de outros recursos pode chegar a prejudicar o cumprimento da data prometida. No exemplo estudado, observou-se que tais conflitos não chegaram a atrasar os projetos, não obstante, a avaliação de seu desempenho foi baseada nos atrasos provocados pela falta de recurso na execução de atividades.

Esta pesquisa identificou e analisou alguns aspectos teóricos e práticos que permitem entender o comportamento importantes elementos na aplicação da metodologia CC/BM na gestão de múltiplos projetos simultâneos. Contudo, no estudo da gestão de múltiplos projetos existe ainda muito caminho a percorrer, especialmente na exploração da recente metodologia da CC/BM.

Os experimentos aqui realizados foram limitados, caracterizando o início de uma linha de pesquisa num tema relevante e ainda pouco explorado na literatura. A extensão dos experimentos para um número maior de casos e melhor avaliação dos efeitos seria uma importante continuação deste trabalho. O exame dos fenômenos aqui estudados em problemas reais é um aspecto importante e de difícil realização, pois as ações gerenciais, na prática, são muito mais flexíveis e amplas do que se pode imaginar a priori. Além disso, a ocorrência de fatos imprevistos é, na prática, muito maior do que se pode tratar de forma produtiva de forma sistemática.

De qualquer forma estudos empíricos para examinar o que, de fato, ocorre em situações reais são absolutamente essenciais para o progresso dos métodos formais de gerenciamento de projetos. Um estudo de caso em empresa de

engenharia de projetos sob encomenda estava inicialmente incluído na proposta desta pesquisa. Entretanto logo se percebeu a inviabilidade de se obter informações úteis num prazo compatível com uma dissertação de mestrado.

Pesquisas futuras que abordem o comportamento dos ***buffers*** **tambor-BT** e a aplicação da metodologia em problemas reais, seriam interessantes para ampliar o nível de entendimento da metodologia e verificar a freqüência e gravidade dos fenômenos aqui estudados. Da mesma maneira pesquisas que aprofundem no desenvolvimento de algoritmos de programação e reprogramação dos múltiplos projetos segundo as bases teóricas da metodologia CC/BM são necessárias para melhor atender às necessidades do mundo real.

7
Bibliografia

Anavi-Isakow, S. & Golany, B. Managing multi-project environments through constant work-in-process. International Journal of Project Management, v.21, n.1, p.9-18. 2003.

Barcaui, Andre & Quelhas, Osvaldo. Corrente crítica: Uma alternativa à gerência de projetos tradicional. Revista Pesquisa e Desenvolvimento Engenharia de Produção**:** 1 - 21 p. jul. 2004.

Bernardi, G & Walter, Claudio. Especificações dos requisitose proposta de um modelo de referencia de um sistema de informação para produção. Semana Academica do CPGCC, III. Rio GRande do Sul 1998.

Casella, G. & Berger, R. L. Statistical inference. 2: Belmont, CA: Duxbury Press**:** 95-97 p. 2002.

Chen, Chin Sheng. Concurrent engineer-to-order operation in the manufacturing engineering contracting industries. Int. Industrial and Systems Engineering, v.1, n.1/2. 2006.

Chiu-Chi, Wei ; Ping-Hung, Liu & Ying-Chin, Tsai. Resource-constrained project management using enhanced theory of constraint. International Journal of Project Management, v.20, n.7, p.561-567. 2002.

Cohen, Izack; Mandelbaum, Avishai & Shtub, Avraham. Multi-project scheduling and control: A process-based comparative study of the critical chain methodology and some alternatives [1]. Project Management Journal, v.35, n.2, p.39. 2004.

Demeulemeester, Erik & Herroelen, Willy. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. Management Science, v.38, n.12, p.1803. 1992.

Goldratt, Eliyahu M. The goal. New York: North River Press. 1992

Hans, E. W.; Herroelen, W. ; Leus, R. & Wullink, G. A hierarchical approach to multi-project planning under uncertainty. Omega, v.35, n.5, p.563-577. 2007.

Herroelen, Willy. Project scheduling-theory and practice. Production and Operations Management, v.14, n.4, p.413. 2005.

Herroelen, Willy & Leus, Roel. On the merits and pitfalls of critical chain scheduling. Journal of Operations Management, v.19, n.5, p.559-577. 2001.

Herroelen, Willy; Leus, Roel & Demeulemeester, Erik. Critical chain project scheduling: Do not oversimplify. Project Management Journal, v.33, n.4, p.48. 2002.

Jyh-Bin, Yang. How the critical chain scheduling method is working for construction. Cost Engineering, v.49, n.4, p.25. 2007.

Kjersti, Hoel & Taylor, Sam G. . Quantifying buffers for project schedules. Production and Inventory Management Journal, v.40, n.2, p.43. 1999.

Kolisch, Rainer. Resource allocation capabilities of commercial project management software packages. Interfaces, v.29, n.4, p.19. 1999.

Kurtulus, I. & Davis, E. W. Multi-project scheduling: Categorization of heuristic rules performance. Management Science, v.28, n.2, p.161. 1982.

Leach, Lawrence P. Critical chain project management: Artech House, INC. 2005

Newbold, Robert C. Project management in the fast lane: Applying the theory of constraints. Boca Raton, FL: CRC Press Taylor & Francis Group, LLc. 1998. 284 p. (The st. Lucie press/apics series on constraints management)

Patterson, James H. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. Management Science, v.30, n.7, p.854 - 867. Jul. 1984.

Pritsker B., Alan A.; Watters, Laurence J. & Wolfe, Philip M. Multiproject scheduling with limited resources: A zero-one programming approach. Management Science, v.16, n.1, set. 1969. 1969.

Rand, Graham K. Critical chain: The theory of constraits applied to project management. International Journal of Operations & Production Management, v.18, p.173 - 177. 2000.

Raz, Tzvi; Barnes, Robert & Dvir, Dov. A critical look at critical chain project management. Project Management Journal, v.34, n.4, p.24. 2003.

Rozenes, Shai; Vitner, Gad & Spraggett, Stuart. Project control: Literature review. Project Management Journal, v.37, n.4, p.5. 2006.

Steyn, H. Project management applications of the theory of constraints beyond critical chain scheduling. International Journal of Project Management, v.20, n.1, p.75-80. 2002.

Tukel, Oya I. ; Rom, Walter O. & Eksioglu, Sandra Duni. An investigation of buffer sizing techniques in critical chain scheduling. European Journal of Operational Research, v.172, n.2, 16 July, p.401-416. 2006.

8
Apêndice A: Prochain & Pipeline

8.1.
Parâmetros de programação

Prochain e Pipeline são as ferramentas utilizadas na aplicação da metodologia CC/BM. Os parâmetros de programação do Prochain, utilizados nos principais passos para a obtenção do *programa de linha de base* do projeto individual são:

Tabela 8.1: Parâmetros de programação de projetos individuais - *Prochain*

Passos de programação	Opções de programação	Escolha
Nivelamento	Otimizar o seqüenciamento de atividades	*Sim*
Identificação da cadeia critica	O caminho mais longo	*Sim*
	O caminho mais restrito	*Não*
Criação de *buffers*	Método das porcentagens	*Sim Herroelen & Leus) Não*
	Método da soma de quadrados	
Inserção de *buffers*	Consolidar o risco	*Sim*
	Criar o programa de linha de base	*Sim*
Atualizações do programa:	Aplicar os tempos atuais nos relacionamentos entre atividades	*Sim*
	Criar um quadro de estado dos *buffers*	*Sim*
	Otimizar os tempos projetados	*Não*

A opção de *consolidar o risco* permite compactar visualmente o programa Steyn), trasladando a segurança dos *buffers de alimentação* para os *buffers do projeto* quando não existe espaço suficiente para sua inserção.

Os parâmetros de programação do Pipeline utilizados nos principais passos para a obtenção do programa máster de múltiplos projetos são:

Tabela 8.2: Parâmetros de programação de múltiplos projetos - *Pipeline*

Passos de programação	**Opções de programação**	**Escolha**
Dos projetos que formam o *Máster*	Priorizar os projetos	*Sim*
	Usar opções individuais ou usar as opções do máster	*Opções do máster*
	Copiar no Máster alguns ou todos os recursos utilizados no projeto individual	*Todos os recursos*
Dos recursos utilizados no *Máster*	Marcar recurso gargalo	*Sim*
	Indicar a porcentagem de capacidade reservada	*Sim (20%,25%)*
	Indicar se forma parte de uma classe de recursos (tem *parent*)	*Não*
	Marcar o recurso para criar uma lista de tarefas pendentes ou não	*Sim*
Das opções do perfile de programação atual:	Salvar câmbios no projeto individual	*Sim*
	Salvar uma copia do projeto individual	*Sim*
	Marcar no programa os pontos de fim e *buffers* da corrida anterior	*Sim*
	Fazer em cada corrida a reprogramação de todos os projetos do *Máster* ou não	*Não, só no início*
	Nivelar em cada corrida todos os projetos do *Máster* ou não	*Não, só no início*
	Indicar a porcentagem permitida de expansão no horizonte de trabalho dos projetos do *Máster.* (ate quando pode se estender o fim do projeto)	*Sim, 100%*
	Decidir se os tempos projetados ignoram ao recurso gargalo ou não.	*Não*

8.2.
Principais Indicadores

Nos gráficos de Gantt apresentados pelo Prochain e Pipeline se mostram indicadores que permitem entender a lógica da programação:

Tabela 8.3: Indicadores

Reduction impact	Calculado quando se inserem os buffers. Disse a percentagem em que a duração do projeto é afetada se a atividade em questão é reduzida. 100 % significa um impacto direto na data final do projeto.
Check task	É uma lista de tarefas que causam uma porcentagem do consumo dos BA e BP. Pode ser considerado como a tarefa incompleta mais imediata que esta causando o consumo do buffer.
% Impact	Porcentagem do consumo do BP que é causado pela atividade em questão. Calculado com os tempos de inicio mais cedo possível.
Folga	Calculado durante o nivelamento e a atualização dos buffers. Tempo entre o inicio e fim mais cedo de cada atividade.

9
Apêndice B: Resultados da simulação

9.1.Análise das causas de atrasos

Tabela 9.1: Componentes de consumo do BP - 25% prob. Atraso

Componentes de Consumo de BP - (25% prob. Atraso)						
MEDIAS	TOTAL_P1atr	TOTAL_P1nr	TOTAL_P2atr	TOTAL_P2nr	P3A1_atr	P3A1_nr
BP50_BC00	3.1	0.0	2.4	0.9	2.6	1.8
BP50_BC15	3.2	0.0	2.7	0.6	2.3	1.0
BP50_BC25	3.1	0.0	2.5	0.4	2.3	1.1
BP50_BC35	3.1	0.0	2.4	0.1	2.2	0.4
BP50_BC45	2.8	0.0	2.5	0.0	2.3	0.2

Tabela 9.2: Atraso efetivo dos projetos - 25:% prob. de atraso

Atraso efetivo do Projeto - (25% prob. Atraso)				
	P2	P2 (BP=3)	P3	P3 (BP= 3)
MEDIAS	Consumo BP real	Delay real	Consumo BP real	Delay real
BP50_BC00	3.3	0.3	4.4	1.4
BP50_BC15	3.2	0.2	3.3	0.3
BP50_BC25	2.9	-0.2	3.4	0.4
BP50_BC35	2.5	-0.5	2.6	-0.4
BP50_BC45	2.6	-0.5	2.5	-0.5

Tabela 9.3: Componentes de consumo do BP - 20% prob. Atraso

Componentes de Consumo de BP - (20% prob. Atraso)						
MEDIAS	TOTAL_P1atr	TOTAL_P1nr	TOTAL_P2atr	TOTAL_P2nr	P3A1_atr	P3A1_nr
BP50_BC00	2.7	0.0	2.0	0.9	2.2	1.5
BP50_BC15	2.9	0.0	2.5	0.4	2.4	0.9
BP50_BC25	3.1	0.0	2.1	0.4	2.2	1.2
BP50_BC35	3.1	0.0	2.2	0.1	2.0	0.4
BP50_BC45	3.0	0.0	2.0	0.1	2.2	0.3

Tabela 9.4: Atraso efetivo dos projetos - 20:% prob. de atraso

Atraso efetivo do Projeto - (20% prob. Atraso)				
	P2	P2 (BP=3)	P3	P3 (BP= 3)
MEDIAS	Consumo BP real	Delay real	Consumo BP real	Delay real
BP50_BC00	2.9	-0.1	3.7	0.7
BP50_BC15	2.9	-0.1	3.3	0.3
BP50_BC25	2.5	-0.5	3.4	0.4
BP50_BC35	2.3	-0.7	2.4	-0.6
BP50_BC45	2.1	-0.9	2.5	-0.5

Tabela 9.5: Componentes de consumo do BP - 15% prob. Atraso

Componentes de Consumo de BP - (15% prob. Atraso)						
MEDIAS	TOTAL_P1atr	TOTAL_P1nr	TOTAL_P2atr	TOTAL_P2nr	P3A1_atr	P3A1_nr
BP50_BC00	2.8	0.0	1.9	0.7	2.0	1.5
BP50_BC15	2.7	0.0	1.8	0.4	1.9	0.7
BP50_BC25	2.4	0.0	2.2	0.2	1.8	0.7
BP50_BC35	2.8	0.0	1.7	0.1	1.7	0.4
BP50_BC45	2.7	0.0	1.9	0.0	2.0	0.2

Tabela 9.6: Atraso efetivo dos projetos - 15:% prob. de atraso

Atraso efetivo do Projeto - (15% prob. Atraso)				
	P2	P2 (BP=3)	P3	P3 (BP= 3)
MEDIAS	Consumo BP real	Delay real	Consumo BP real	Delay real
BP50_BC00	2.6	-0.4	3.4	0.4
BP50_BC15	2.2	-0.8	2.6	-0.4
BP50_BC25	2.4	-0.6	2.5	-0.5
BP50_BC35	1.8	-1.2	2.1	-0.9
BP50_BC45	1.9	-1.1	2.2	-0.8

Tabela 9.7: Componentes de consumo do BP - 10% prob. Atraso

Componentes de Consumo de BP - (10% prob. Atraso)						
MEDIAS	TOTAL_P1atr	TOTAL_P1nr	TOTAL_P2atr	TOTAL_P2nr	P3A1_atr	P3A1_nr
BP50_BC00	2.2	0.0	1.5	0.7	1.6	1.2
BP50_BC15	2.3	0.0	1.8	0.4	1.6	0.6
BP50_BC25	2.3	0.0	1.6	0.2	1.5	0.6
BP50_BC35	2.0	0.0	1.4	0.0	1.7	0.1
BP50_BC45	2.0	0.0	1.3	0.1	1.8	0.1

Tabela 9.8: Atraso efetivo dos projetos - 10:% prob. de atraso

Atraso efetivo do Projeto - (10% prob. Atraso)				
	P2	P2 (BP=3)	P3	P3 (BP= 3)
MEDIAS	Consumo BP real	Delay real	Consumo BP real	Delay real
BP50_BC00	2.2	-0.8	2.8	-0.2
BP50_BC15	2.2	-0.9	2.1	-0.9
BP50_BC25	1.8	-1.3	2.1	-0.9
BP50_BC35	1.4	-1.6	1.9	-1.2
BP50_BC45	1.4	-1.6	1.9	-1.1

9.2.
Trade-off entre *Makespan* e diminuição de atrasos por falta de recurso

Tabela 9.9: *Trade-off* (25% prob. Atraso)

Trade-off em porcentagens - (25% prob. Atraso)								
Duraçao enxuta do projeto	P2 = 5				P3 = 5			
MEDIAS	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR
BP50_BC00 (0)(0)	0	18.50	0	0.00	0	36.75	0	0.00
BP50_BC15 (1)(1)	20	11.00	-7.50	-19.26	20	20.50	-16.25	-17.84
BP50_BC25 (2)(1)	40	7.00	-11.50	-29.53	20	21.50	-15.25	-16.74
BP50_BC35 (3)(2)	60	2.25	-16.25	-41.72	40	7.75	-29.00	-31.83
BP50_BC45 (4)(3)	80	0.20	-18.30	-46.98	60	4.60	-32.15	-35.29
		38.95				**91.10**		

Tabela 9.10: *Trade-off* (20% prob. Atraso)

Trade-off em porcentagens - (20% prob. Atraso)								
Duraçao enxuta do projeto	P2 = 5				P3 = 5			
MEDIAS	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR
BP50_BC00 (0)(0)	0	17.75	0	0.00	0	30.75	0	0.00
BP50_BC15 (1)(1)	20	7.25	-10.50	-28.42	20	18.25	-12.50	-14.39
BP50_BC25 (2)(1)	40	8.50	-9.25	-25.03	20	24.50	-6.25	-7.20
BP50_BC35 (3)(2)	60	1.25	-16.50	-44.65	40	7.75	-23.00	-26.48
BP50_BC45 (4)(3)	80	2.20	-15.55	-42.08	60	5.60	-25.15	-28.96
		36.95				**86.85**		

Tabela 9.11: *Trade-off* (15% prob. Atraso)

Trade-off em porcentagens (15% prob. Atraso)								
Duraçao enxuta do projeto	P2 = 5				P3 = 5			
MEDIAS	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR
BP50_BC00 (0)(0)	0	14.00	0	0.00	0	29.50	0	0.00
BP50_BC15 (1)(1)	20	8.00	-6.00	-21.47	20	13.50	-16.00	-23.15
BP50_BC25 (2)(1)	40	4.00	-10.00	-35.78	20	13.25	-16.25	-23.52
BP50_BC35 (3)(2)	60	1.75	-12.25	-43.83	40	8.25	-21.25	-30.75
BP50_BC45 (4)(3)	80	0.20	-13.80	-49.37	60	4.60	-24.90	-36.03
		27.95				**69.10**		

Tabela 9.12: *Trade-off* (10% prob. Atraso)

Trade-off em porcentagens (10% prob. Atraso)								
Duraçao enxuta do projeto	P2 = 5				P3 = 5			
MEDIAS	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR	% Makespan	% Atraso NR	Diminuiçao do atraso NR	% Diminui o atraso NR
BP50_BC00 (0)(0)	0	13.50	0	0.00	0	24.00	0	0.00
BP50_BC15 (1)(1)	20	7.75	-5.75	-21.70	20	11.25	-12.75	-24.47
BP50_BC25 (2)(1)	40	3.75	-9.75	-36.79	20	12.00	-12.00	-23.03
BP50_BC35 (3)(2)	60	0.50	-13.00	-49.06	40	2.25	-21.75	-41.75
BP50_BC45 (4)(3)	80	1.00	-12.50	-47.17	60	2.60	-21.40	-41.07
		26.50				**52.10**		

9.3.
Efeitos dos critérios de priorização

Tabela 9.13: Atraso efetivo do projeto um

ATRASO PROJETO 1			
Prob. Atraso projeto P1	**Priori. Fixa**	**Priori.Dinâmica**	**Priori. Mista**
80%	4.5	7.0	7.0
60%	1.6	3.0	2.2
47%	1.1	1.2	1.7
25%	-0.9	-0.7	-0.4
20%	-1.0	-0.9	-1.0
15%	-1.3	-1.3	-1.2
10%	-1.5	-1.6	-1.4
5%	-2.1	-2.1	-2.0

Tabela 9.14: Atraso efetivo do projeto dois

ATRASO PROJETO 2			
Prob. Atraso projeto P2	**Priori. Fixa**	**Priori. Dinamica**	**Priori. Mista**
80%	6.5	5.4	5.0
60%	3.0	3.2	3.1
47%	2.6	2.2	2.1
25%	0.7	0.4	0.4
20%	0.2	0.0	0.1
15%	-0.1	0.0	0.0
10%	-0.6	-0.5	-0.5
5%	-1.0	-0.9	-1.0

Tabela 9.15: Atraso efetivo do projeto três

ATRASO PROJETO 3			
Prob. Atraso projeto P3	**Priori. Fixa**	**Priori. Dinamica**	**Priori. Mista**
80%	8.0	8.2	9.1
60%	5.7	5.1	5.4
47%	3.3	3.3	3.2
25%	1.4	1.4	1.5
20%	1.1	1.0	1.1
15%	0.6	0.4	0.4
10%	-0.2	-0.2	-0.2
5%	-0.7	-0.8	-0.7

Tabela 9.16: Atraso efetivo do sistema

TOTAL ATRASO SISTEMA			
Prob. Atraso	**Priori. Fixa**	**Priori. Dinamica**	**Priori. Mista**
80%	19.0	20.6	21.2
60%	10.3	11.3	10.7
47%	7.0	6.7	6.9
25%	1.1	1.0	1.5
20%	0.3	0.2	0.2
15%	-0.8	-0.9	-0.8
10%	-2.3	-2.3	-2.1
5%	-3.9	-3.7	-3.7

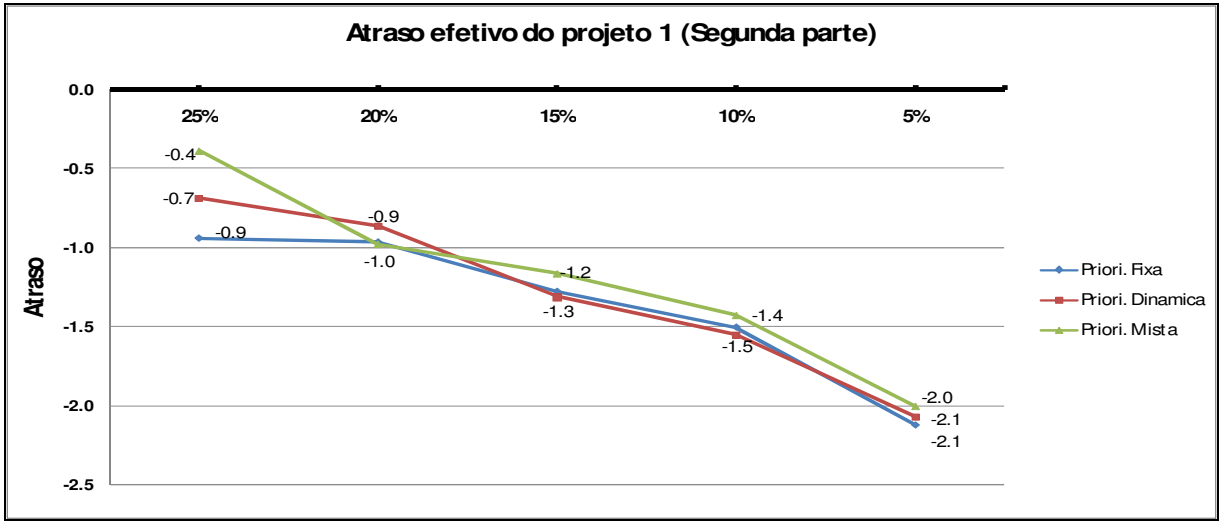

Figura 9.1: Atraso do projeto um – segunda parte

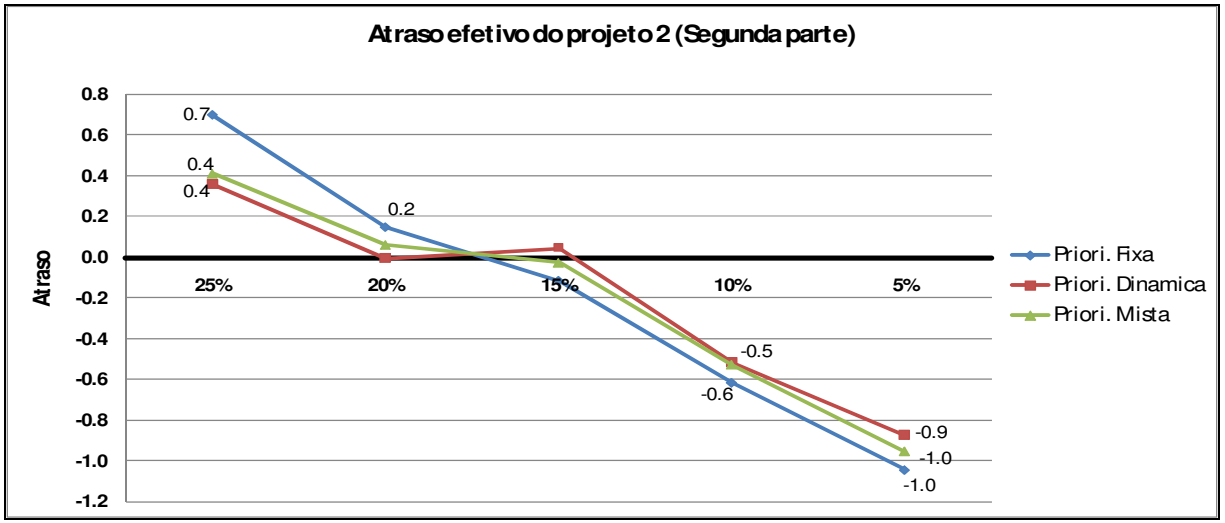

Figura 9.2: Atraso do projeto dois – segunda parte

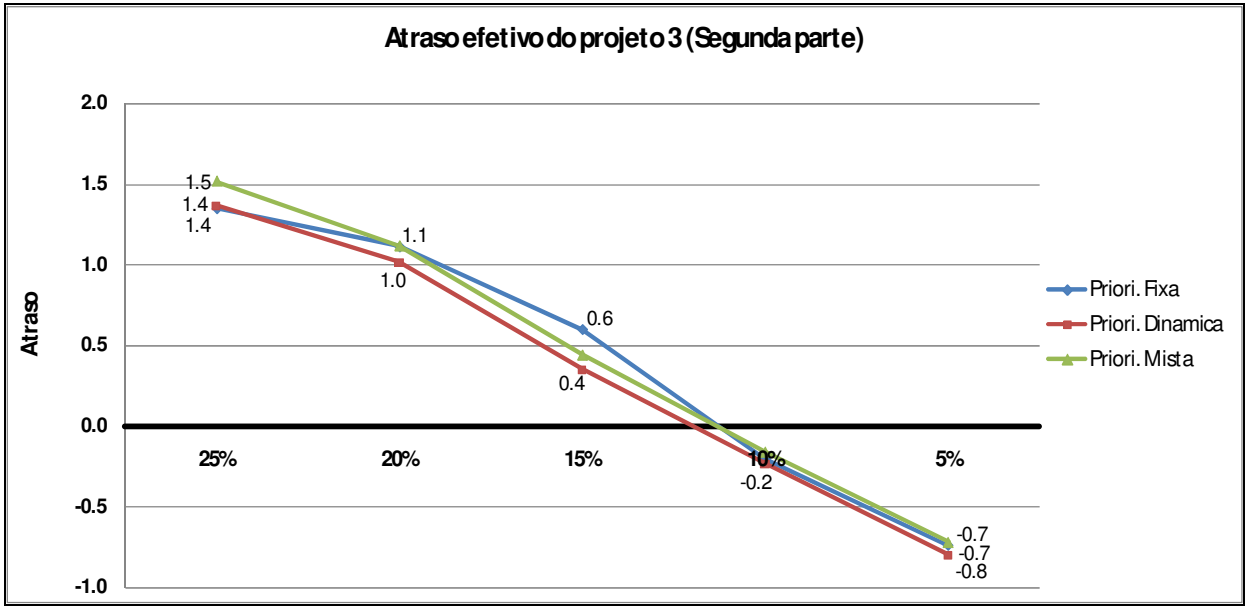

Figura 9.3: Atraso do projeto dois – segunda parte