UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO E SISTEMAS

CARACTERIZAÇÃO DE PROCEDIMENTOS HEURISTICOS EM PLANEJAMENTO E PROGRAMAÇÃO DE PROJETOS

MARIA RITA PONTES ASSUMPÇÃO ALVES

SÃO CARLOS
1988

CARACTERIZAÇÃO DE PROCEDIMENTOS HEURISTICOS EM PLANEJAMENTO E PROGRAMAÇÃO DE PROJETOS

1988?

Maria Rita Pontes Assumpção Alves

Departamento de Engenharia de Produção
e Sistemas

A P R E S E N T A Ç Ã O

Este trabalho é resultado de pesquisa sobre o problema de alocação de recursos em programação de projetos, que vem sendo desenvolvida junto à Escola de Engenharia de São Carlos, da Universidade de São Paulo.

São apresentados o problema, procedimentos heurísticos para sua resolução e categorização das principais regras, baseada em estudos comparativos desenvolvidos nestes últimos anos por pesquisadores da área.

Neste instante, está sendo elaborada uma classificação das heurísticas, relacionando sua qualidade e progresso da solução, segundo propriedades funcionais e características do problema, como por exemplo: estática ou dinâmica; representando informações locais e/ou informações globais, etc.

Esta classificação, tem como hipótese, a fundamentação proposta para o algoritmo de busca que, se sujeito a uma função avaliação sob determinadas condições, poderá orientar uma busca ótima.

O objetivo final é o desenvolvimento de uma medida, extraída da relação uso de recurso e progresso da programação, que seja eficiente para a resolução do problema, como proposto no Projeto de Tese.

SUMÁRIO

CAPÍTULO I

O PROBLEMA DE ALOCAÇÃO DE RECURSOS EM PROGRAMAÇÃO DE PROJETOS

PROJETO é um particular sistema de produção, organizado para a realização de um determinado objetivo (bem ou serviço), consubstanciado na execução de atividades específicas e interrelacionadas, geralmente requerendo a contribuição de grupos multidisciplinares, sob uma única direção (Alves, 87).

A programação do projeto consiste no estabelecimento de datas para realização das suas atividades componentes, sujeitas a restrições de precedência tecnológica e ambientais, definidas pela sua organização gerenciadora.

As restrições tecnológicas são expressas pela construção de um grafo, modelando o desenvolvimento da execução do projeto. Este grafo estabelece uma função de incidência entre um conjunto de vértices

$$V = \{ v_i, \; i = 1, \ldots, n\}$$

e um conjunto de arcos

$$A = \{ a_j, \; j = 1, \ldots, p\}$$

onde são estabelecidas as relações de precedência entre as atividades do projeto.

A representação gráfica do projeto pode ser feita por duas abordagens:

(i) Rede de atividades:

A = {atividades do projeto} $A \neq \emptyset$

V = {relações de precedência}

Nesta representação o grafo deve ser simples, orientado, conexo e acíclico.

(ii) Redes de precedência

A = {relações de precedência}

V = {atividades do projeto} $V \neq \emptyset$

As redes de precedência também não admitem ciclos e são orientadas.

As primeiras técnicas desenvolvidas para programação de projetos, PERT (Program Evaluation Review Technique) e CPM (Critical Path Method), consideram, apenas, a variável-tempo associada às atividades. A primeira oferece um tratamento probabilístico à estimativa da duração das atividades, inferindo probabilidade às datas de término do projeto, a segunda considera duração determinística às atividades do projeto e ambas, supondo disponibilidade ilimitada de recursos, necessários à execução do projeto, calculam o caminho crítico e estabelecem as datas de início e término mais cedo e mais tarde para todas as atividades.

Quando considerações sobre os recursos demandados são feitas, para a programação do projeto, faz-se necessária análise e designação dos recursos, associados às datas estabelecidas para as atividades.

Neste contexto, a programação do projeto se vale dos dados da análise do caminho crítico e o problema de alocação de recursos é estabelecido, conforme a problemática a ser resolvida, definida pelos objetivos organizacionais e o ambiente em que o projeto (ou projetos) está inserido.

A alocação de recursos em programação de projetos pode ser abordada de diferentes formas e, classicamente, se definem três grupos de problemas:

O problema trade-off tempo custo

É frequente que a performance de algumas ou todas atividades do projeto podem ser aceleradas pela alocação de maior quantidade de recursos, a expensas de maior custo direto da atividade. Quando isto ocorre, existem diferentes combinações entre as durações das atividades, gerando programações com a duração desejada. Entretanto cada programação pode estar associada a um valor diferente do custo total do projeto. Os procedimentos para resolução do problema trade-off tempo X custo são orientados para determinar a programação de menor custo para uma dada duração do projeto, normalmente sob a hipótese de recursos ilimitados. Referências de métodos para este problema são apresentados por Moder et al (83).

Nivelamento de Recursos

Este problema ocorre quando existe suficiente recurso para programar todas as atividades competindo pelos mesmos tipos de recursos, mas é desejável a utilização dos mesmos a uma taxa constante. O objetivo é, então, nivelar tanto quanto

possível os histogramas de uso dos recursos sobre o horizonte da programação, dentro da duração do projeto já estabelecida. Isto é determinado pela reprogramação das atividades dentro de suas folgas disponíveis, até alcançar um histograma aceitável.

Em alguns procedimentos para obtenção desta uniformização, um limite sobre a disponibilidade dos recursos é dada. Neste caso, há possibilidade de incluir uso "secundário" de recursos tais como, horas extras ou turnos de trabalho, ou uso de taxas alternativas de recursos. A diferença entre o problema de nivelamento com limitação de recursos e o problema de alocação de recursos limitados é quanto a expansão da duração do projeto, que no primeiro é fixada pela duração do caminho crítico.

Alocação de Recursos Limitados

Neste caso, tem-se para cada período do horizonte de programação, quantias fixadas para a disponibilidade de recursos demandados pelo projeto.

Quando este limite não é suficiente para satisfazer o requisito de todas as atividades programadas para o mesmo período, estabelecido pelo CPM-padrão, decisões sobre ressequenciamento das mesmas são necessárias, frequentemente resultando em aumento na duração do projeto, além daquela determinada pelo caminho crítico.

Este problema é o objeto deste trabalho e sua exploração é apresentada no texto. Em primeiro lugar apresentamos as várias hipóteses para tratamento do problema e então, as

duas abordagens utilizadas para sua resolução, segundo a qualidade da solução encontrada: procedimentos analíticos e heurísticos.

Apresentamos na Tabela 1.1. as principais características dos problemas acima.

Embora existam técnicas específicas a cada classe dos problemas citados, elas podem ser usadas concomitantemente numa situação de programação de projeto.

TABELA 1.1. Classificação dos Problemas de Alocação de Recursos em Programação de Projetos

Classe dos problemas		Problemas específicos	Objetivo	Técnicas para solução
Análise		Análise da alocação dos recursos	Para uma dada programação, o histograma de carga de recurso mostra a implicação no tempo pelo uso dos vários tipos de recursos no projeto. Vários parâmetros são definidos para a análise da criticalidade dos recursos, quando limites são estabelecidos.	- diagramas de carga de recursos - curva cumulativa de recursos - índices de criticalidade - localização dos picos - ociosidade.
ALOCAÇÃO DE RECURSOS	Nivelamento	Recursos ilimitados (não ocorrência de conflito no uso de recursos) Data de término do projeto pré-fixada Pode fixar uma taxa de uso de recursos por período, para o projeto. Pode considerar variações tecnológicas para execução das atividades e/ou uso secundário de recursos (horas extras, etc.)	Para uma determinada data de término do projeto, programar as atividades tal que o uso dos recursos sobre os períodos do projeto tenha uma mínima variação. (Uniformizar tanto quanto o possível, os diagramas de carga de recurso).	Baseado em decisões locais para melhoria da programação, usando a folga para deslocamento das atividades. Baseado em decisões locais para melhoria da programação, usando a folga para deslocamento das atividades.
	trade-off tempo x custo	Recursos ilimitados Considera variações tecnológicas para as atividades (duração variável)	Dada uma programação, ajustar uma combinação de duração das atividades que determine uma data mais cedo de término do projeto ou atinja uma duração pré-estabelecida, ao menor custo possível.	Algoritmos exatos
	Recursos limitados	Ocorrência de conflito entre atividades programadas para um mesmo período, devido à inflexibilidade na disponibilidade de recursos.	Resolver o conflito, tornando a programação viável	Procedimentos heurísticos Algoritmos exatos.

CAPÍTULO II

O PROBLEMA DE ALOCAÇÃO DE RECURSOS LIMITADOS EM PROGRAMAÇÃO DE PROJETOS

O problema de alocação de recursos escassos em programação de projetos é complexo, tanto na sua formulação, quanto na sua resolução.

Para sua formulação, tem-se que considerar a problemática a ser analisada, referente a:

- abordagem para definição da duração de suas atividades componentes;
- possibilidade, ou não, de interrupção das atividades;
- número, tipo e quantidade de recursos escassos a serem analisados, por atividade e por projeto;
- objetivo estabelecido para a programação;
- abordagem de projeto único ou de múltiplos projetos.

Assim sendo, levantaremos algumas questões sobre os quesitos acima mencionados.

. Duração das Atividades

Como já mencionado na introdução deste trabalho, existem duas abordagens básicas para a definição da duração das atividades: probabilística e determinística.

Na abordagem probabilística as durações das atividades são variáveis aleatórias com função distribuição suposta conhecida. Assim, parte-se de estimativas probabilísticas para determinação da duração das atividades. No PERT considera-se que os tempos de execução das atividades seguem uma distribuição tipo BETA, usando três medidas para a definição da duração esperada - estimativas: otimista, pessimista e mais provável.

No tratamento determinístico a duração da atividade é definida de modo único, na fase de planejamento do projeto.

Lee & Park (78) apresentam uma metodologia para ser usada na fase de planejamento do projeto, para estimativa do requisito de recursos, associada à determinação da duração de uma atividade. O objetivo é determinar níveis de recurso com menores custos, considerando uso de horas-extras, ou não, e analisando combinações alternativas sobre uso de equipamentos e mão de obra requerida para seu manuseio. Serve como suporte para definição da duração de uma atividade, de forma determinística, analisando-se as várias formas alternativas para sua execução, considerando as opções tecnológicas e seu requisito de recurso.

Para projetos requerendo continuidade no uso de determinados recursos, por grupos de atividades a serem executadas consecutivamente, tais como construção de estradas, de oleodutos, ou mesmo de edifícios com vários andares, as quantidades de recursos por atividade, são determinadas tal que minimize a duração do projeto. Um método proposto por Selinger (80), utiliza volume de trabalho por atividade e quantias disponíveis de recursos, como dados de entrada para a determinação da duração das atividades e minimização da duração do

projeto. Outras referências podem ser encontradas em Johnston (81).

O tratamento das várias opções tecnológicas para execução das atividades (modos de execução) é inserido na fase de programaçaõ do projeto,com as formulações apresentadas em Slowinski [(80), (81)], Weglarz (81), Talbot (82) e Reeves (82).

A consideração dos vários modos alternativos de execução das atividades pode ser feito por duas formas:

. discreto: quando as relações entre duração e requisito de recurso para uma atividade são pontuais;

. contínuo: quando se estabelece uma função, associando requisito de recurso e velocidade de execução da atividade.

A formulação do problema de alocação de recursos limitados, então, busca estabelecer uma programação onde não apenas são indicadas datas de início e término para as atividades, como também o modo de executá-la. Normalmente, o objetivo é o de buscar a duração mínima do projeto, ou a de custo mínimo. Neste último caso, o problema seria de trade-off tempo X custo com recurso limitado.

Estas formulações consideram uma categorização de tipos de recursos, de acordo com que as restrições sobre sua disponibilidade sejam por período da programação, quando são chamados renováveis; por todo o projeto ou parte da vida do projeto, sendo chamados de não-renováveis e finalmente, os recursos duplamente restritos, quando ambas as disponibilidades,tanto a periódica como a total, são limitadas.

Estas formulações são direcionadas para a construção de um modelo geral para o problema de alocação de recursos limitados em programação de projetos, onde recursos de diferentes naturezas em termos de sua divisibilidade, preemptibilidade e modos de restrição são considerados.

Normalmente, dinheiro é tratado como recurso renovável (desde que pode ser restrito a um certo nível, em cada período), ou como recurso não renovável (quando apenas seu consumo total é fixado). Na prática, entretanto, é um recurso duplamente restrito (há controle de fluxo de caixa e de gasto total). Da mesma maneira, mão de obra, tratado usualmente como recurso renovável, pode ser considerado recurso não renovável, no caso de o número de homens-hora ser restrito para todo o projeto.

Classicamente os trabalhos desenvolvidos para tratamento analítico do problema de alocação de recursos em programação de projetos consideram o caso de cada atividade ser executada por um único modo, ou seja, considera-se uma única duração com sua necessidade por recursos. Um dos primeiros procedimentos não analíticos para tratamento de modos opcionais para execução de uma atividade, desenvolvido na década de 60, foi o RAMPS [Resource Allocation and Multi-Project - (Lambourn, 63)] utilizando heurísticas na busca da solução. As versões I e II do SPAR (Scheduling Program for Allocating Resources), desenvolvido por Wiest (67), tratam também com a questão de requisitos de recursos fixos ou variáveis, mas de modo discreto.

Número de Tipos de Recursos e Quantidade Requerida e Disponível

Considerando o problema clássico de tratar um único modo de execução da atividade e valores discretos para o tratamento dos recursos, Davis (73) faz uma classificação do problema de alocação de recursos em programação de projetos, segundo o número de tipos de recursos e as quantias requeridas e disponíveis dos recursos envolvidos no projeto:

1) um único recurso comum a todas as atividades, com demanda e disponibilidade expressas em múltiplos da unidade.

2) mais que um tipo de recurso por projeto, mas unicamente um tipo de recurso por atividade, com demanda e disponibilidade limitadas a uma unidade.

3) mais que um tipo de recurso por projeto e por atividade, com demanda e disponibilidade expressas em múltiplos da unidade.

O primeiro caso pode ser análogo à modelagem do Problema de Balanceamento de Linhas de Montagem (PBL), embora este último trate da programação de operações repetitivas e grande número de produtos finais idênticos, enquanto que a programação de projetos (PPR) trata com inúmeras atividades específicas para a produção de um único "produto".

Tabela 2.2. Analogia entre PBL e PPR

BALANCEAMENTO DE LINHA	PROJETO
. elementos de trabalho	. atividades
. tempo dos elementos de trabalho	. demanda do recurso por atividade
. estações de trabalho	. dias
. tempo de ciclo	. disponibilidade do recurso

Esta analogia é estabelecida quando os problemas são expressos em um diagrama de precedência, onde, no caso de programação do projeto, as atividades são indicadas por nós, enquanto que no outro, estes representam os elementos de trabalho e, em ambos, os arcos estabelecem as relações de precedência entre as atividades (ou operações).

A relação feita entre estações de trabalho e dias ocorre se as atividades do projeto são expressas em sub-atividades com duração unitária e com continuidade na sua execução (não preemptivas).

Já no segundo caso pode-se estabelecer analogia entre o problema clássico de programação de job-shop (PJS), segundo as relações mostradas na Tabela 3.

Tabela 2.3 Analogia entre PJS e PPR

JOB - SHOP	PROJETO
tarefa	"produto"
operação	atividade

No PJS, uma tarefa é o desenvolvimento de operações específicas a serem alocadas em determinadas máquinas em uma ordenação estabelecida, para a fabricação de determinado produto.

Para esta comparação, as tarefas são de um tipo (encomenda), com um determinado roteiro para o seqüenciamento de operações, onde cada operação requer uma única unidade de um tipo de máquina,disponível em uma unidade, embora, diferentes tarefas possam usar diferentes tipos de recursos em roteiros específicos.

Na prática, a diferença básica seria a consideração da natureza contínua do fluxo de trabalho na execução das tarefas no problema de job shop, enquanto que, na programação de projetos o sequenciamento das suas atividades é específico e único para cada caso.

A hipótese de requisito unitário de recurso por atividade simplifica a formulação e o procedimento de solução, já que o número de atividades programadas para um dado instante e requerendo o mesmo tipo de recurso é equivalente à quantia requerida deste recurso.

Já o terceiro caso é um problema típico de programação de projetos, com atividades requerendo múltiplos recursos para a sua execução. É chamado Problema de Programação de Projetos com Múltiplos Recursos (PPPR).

Holloway et al (79) apresentam uma classificação dos Problemas de Programação com Recursos Limitados, apresentado na Tabela 2.4., tendo como base o número de tipos de recursos e a disponibilidade por período e requisito por atividade.

Tabela 2.4. Classificação dos problemas de programação com alocação de recursos, segundo o número e quantidade de recursos envolvidos (Holloway et al (79) - p. 862).

TIPO DE PROBLEMA	NÚMERO DE TIPOS DE RECURSOS	UNIDADES DE TIPOS DE RECURSO DISPONÍVEIS POR PERÍODO	NÚMERO DE TIPOS DE RECURSOS REQUERIDOS POR ATIVIDADE
1/1/1	1	1	1
n/1/1	n	1	1
1/n/1	1	n	1
n/n/1	n	n	1
n/1/n	n	1	n
n/n/n	n	n	n

Outra particularidade, que pode ser abordada na formulação do problema, é a fixação de datas para ocorrência de eventos no projeto, sejam intermediários ou final.

O problema de múltiplos recursos pode ser tratado, usando o artifício de desmembramento da atividade demandando múltiplos recursos, por um número de atividades de durações iguais, igual ao número de tipos de recursos demandados. Então é necessário o estabelecimento de mesmas datas início e fim para estas atividades.

. Caso Preemptivo

Outra hipótese a ser considerada é sobre a possibilidde de programação de interrupção de uma atividade já com sua data-início estabelecida, para liberação dos recursos a ela alocados, possibilitando, assim, a programação de outra atividade que, neste período estaria sendo considerada

mais urgente. Posteriormente,o término da atividade interrompida é reprogramado. Este caso, chamado de preemptivo [(Moccellin (87)] foi tratado por Wiest em seu modelo SPAR-1 (Scheduling Program for Allocating Resources) já mencionado, desenvolvido em 1960, que analisa, também, três modos alternativos para execução de uma atividade: normal, acelerado e expandido, se demanda um padrão normal de recursos, ou um nível máximo, ou nível mínimo por período de execução da atividade.

A forma mais comum de abordar o caso preemptivo é a subdivisão da atividade em tantas outras de duração unitária, quanto for o número de períodos definido para a sua duração original.

Ainda na definição do problema de programação de projetos com recursos limitados, coloca-se a questão sobre o número de projetos sob o mesmo conjunto de tipos de recursos escassos a serem alocados.

. Projeto Único ou Múltiplos-Projetos

O problema de múltiplos projetos consiste de vários projetos sendo programados conjuntamente, com disponibilidade limitada sobre recursos comuns.

A programação neste problema é a designação de datas de início para cada atividade de cada um dos projetos, tal que, as restrições de precedência e de requisito e disponibilidade de recursos sejam satisfeitas. Desde que a quantidade de recursos disponíveis por período do horizonte de programação é limitada, a data de início de algumas atividades pode ser atrasada, além da data estabelecida pela programa-

ção pautada na análise CPM-padrão (com a hipótese de recursos ilimitados). Este fato, poderá incorrer em atrasos dos projetos, além da duração do caminho crítico baseado no CPM-padrão.

Uma opção é tratar o problema como se fosse um único projeto, conectando todos os projetos numa rede única por atividades início e fim fantasmas. Caso ocorra defasagem nas datas estabelecidas para início de diferentes projetos, são estabelecidas durações para as atividades fantasmas, de forma a que a programação obedeça a estas premissas.

A maioria dos procedimentos para resolução do problema de alocação de recursos em programação de projetos tem sido tratados com esta abordagem de projeto único [Davis e Heidorn (71); Patterson (73), Patterson e Huber (74)] e tem-se encaminhado a resolução do problema de múltiplos projetos deste modo.

Outra abordagem para o problema envolvendo mais que um projeto é a programação simultânea dos mesmos, considerando-os dependentes unicamente quanto ao limite dos recursos comuns [Fendley (68), Patterson (76), Kurtulus e Davis (82); Kurtulus e Narula (85)] - abordagem de múltiplos projetos.

As duas abordagens poderão produzir programações diferentes, para o mesmo problema.

Uma questão a ser colocada para o problema de múltiplos projetos é sobre a diferenciação no tratamento dos mesmos. Se os projetos oferecem diferentes contribuições à organização gerenciadora, sejam lucrativas ou de imagem do serviço prestado, ou requerem diferentes tipos de supervisão é mais realístico ponderar diferenciadamente cada proje

to. Atribui-se, então, penalidades desiguais a cada projeto. Esta penalidade pode ser uma medida p_i relacionada à multa por dia de atraso, à significância da demanda por recursos, ao lucro proveniente da execução do projeto i, ou outra definida pela gerência.

Caso não haja necessidade desta diferenciação, assumem-se penalidades iguais.

. Objetivo da Programação

A escolha da função objetivo para resolução do problema, depende do ambiente organizacional onde o(s) projeto(s) está inserido.

A natureza do sistema de produção - tipo projeto implica no envolvimento direto do cliente com a estrutura de seu sistema administrativo, fazendo com que as organizações gerenciadoras de projetos tenham como meta a maximização da satisfação de seus clientes. (Wild, 77)

Por outro lado, cada projeto demanda recursos específicos, embora possam ocorrer demandas por recursos comuns por diferentes projetos, desenvolvidos no mesmo momento.

Um sistema de administração de projetos, portanto, tem como requisito externo a satisfação do cliente e como requisito interno a coordenação dos recursos de forma eficiente.

Para o problema de alocação de recursos os objetivos,que atendam a estes requisitos, podem ser formulados em relação ao tempo e/ou racionalidade dos recursos, bem como ao custo.

Quando o objetivo se refere ao tempo, pode-se equa

cionar de forma a ter a minimização de atrasos, relacionados ao término do projeto dada pelo comprimento do caminho crítico. Esta formulação atende a interesses da empresa,associados a evitar-se o pagamento de multas e também à preservação da imagem de seu serviço prestado.

Outra formulação do objetivo é a minimização do tempo de execução do projeto, refletindo assim o período em que a empresa estaria se ocupando com o projeto,desde o seu início até seu término. (Estrutura Organizacional por Projeto).

Pode-se ainda, minimizar o tempo de término do projeto, sem referencia a outra data, seja a determinada pelo caminho crítico ou a de início do projeto.

Pritsker et all (67) apresentam formulações precisas para os objetivos acima referidos.

Quanto à racionalidade na utilização dos recursos, o objetivo é evitar-se ociosidade, na medida em que os recursos implicam em custos estejam ou não em uso.

As atividades dos projetos programadas para serem executadas simultaneamente, concorrem por recursos escassos, que devem ser alocados com eficiência.

Uma medida de eficiência de recurso, normalmente é extraída da relação entre quantia demandada e disponibilidade do recurso.

Para agregação de todos os recursos é necessário a definição de uma unidade única, que pode ser a de unidade de tempo/recurso utilizado ou custo.

Outra questão,que se coloca para a equacionalização do objetivo de busca da eficiência na utilização de recursos, é a disponibilidade do recurso ser constante, ou variável ao longo do horizonte da programação. Neste caso a demanda deve ajustar-se a cada período.

Quando o objetivo se refere a minimização de custo, vários modos de execução da atividade estará sendo considerado e o problema é de trade-off-tempo X custo, com recursos limitados. Referências sobre este tratamento podem ser encontradas em Talbot (82).

2.2. Formulação do Problema

Em essência, o problema de programação de projetos com recursos limitados pode ser formulado como segue: dado um projeto e restrições sobre a disponibilidade dos recursos envolvidos para sua execução, estabelecer datas de início e término e designação de recursos para as atividades que satisfaçam as restrições e minimize (ou maximize) o critério estabelecido.

A característica principal deste problema é a resolução de conflitos entre atividades programadas para o mesmo período, devido a escassez de recursos disponíveis, pelo ressequenciamento destas atividades.

O problema consiste na formulação de um modelo descrevendo as condições do sistema a ser programado, (expressando restrições de procedência de recursos e datas desejadas) e a definição de uma função objetivo.

As restrições de precedência entre as atividades, ou seja, de que o término de uma atividade i é necessária para o início da atividade s, são descritas pela relação:

$$T_s \geq T_i + d_i \qquad (1)$$

onde T_s (T_i) é a data programada para início de s(i) e d_i é a duração da atividade i.

A necessidade do recurso k, requerido por cada atividade i, por período t, é denotado por r_{ikt} e $R = [R_1, R_2 \ldots . R_k]$ denotam as quantidades disponíveis para os K tipos de recursos, constantes ao longo do horizonte de programação.

Uma programação viável estabelece que, para, S_t={conjunto de atividades programadas para o período t }, então

$$\sum_{i \in S_t} r_{ikt} \leq R_k, \qquad t \geq 0 \quad e \quad k \in R \qquad (2)$$

Uma instância do problema é um particular sistema, conjunto de atividades a serem programadas, conjunto de recursos envolvidos e conjunto de restrições estabelecidas pela especificidade da situação.

Dada uma instância I, o modelo define quais soluções viáveis para I e a função objetivo associa valores a cada uma delas.

O valor associado a uma programação ótima OPT(I) é o menor (ou maior) dos valores da função objetivo das programações viáveis para a instância I, conforme o critério deva ser minimizado ou maximizado.

Um algoritmo de programação A para o problema é um procedimento que, dada uma instância I, produz pelo menos uma programação viável para I.

Fazendo A (I) denotar o valor da função objetivo, resultante de A em I, se A(I) é sempre, para qualquer I, igual a OPT(I), então A é um algoritmo de otimização.

No caso da solução ótima ser computacionalmente inviável, obtém-se uma solução aproximada.

. Métodos para Solução

Os trabalhos desenvolvidos para resolução do problema de programação de projetos com recursos escassos tem-se concentrado em duas abordagens, segundo a qualidade da solução:

1. formulação e resolução do problema como de otimização, utilizando-se técnicas usualmente de programação inteira, e recentemente, programação não linear, quando consideradas variações no tempo de execução da atividade como função não linear dos recursos demandados [Slowinsky (81), Weglarz (81)], que buscam a solução ótima para o problema.

As técnicas exatas, tais como programação linear, enumeração limitada, enumeração implícita e ramificação e avaliação (branch and bound) começam com um processo de otimização para solucionar parte do problema de programação, através da relaxação temporária sobre as restrições de recursos e geração de uma árvore de soluções parciais, que então serão consideradas sob a condição de escassez de recursos. Os métodos diferem no modo de geração desta árvore (ou seja, na ordenação das programações parciais a serem consideradas com a alocação dos recursos limitados) e na maneira de reconhecimento e avaliação, e então, rejeição de programações parciais que não levariam à otimalidade. (Balas 1970, Davis 1969, Davis e Heidorn 1971, Fisher 1973, Gorenstein 1972, Patterson e Huber 1974, Patterson e Roth 1976, Schrage 1970, Stinson 1976, 1978,Talbot 1976, 1982, Hasting 1972, 1976).

Avaliação da performance de três procedimentos, considerados como o estado da arte de cada uma destas abordagens (Enumeração Limitada - Davis e Heidorn; Ramificação e Avaliação - Stinson e Enumeração Implícita - Talbot) é apresentada por Patterson (84), que caracteriza classes específicas de instâncias do problema, para as quais um dos procedimentos melhor se adequa.

Estes métodos exatos, devido à complexidade do problema, não são viáveis para resolução de instâncias grandes, com centenas de atividades e muitos recursos com alto índice de criticalidade.

2. procedimentos heurísticos, que são métodos indutivos de construção de uma programação viável, cuja lógica básica é, sistematicamente, resolver os conflitos entre atividades programadas para o período em que competem por recursos escassos, através do uso de regras de priorização ou "heurística", na determinação de um ressequenciamento das mesmas dentro de uma alocação viável.

A escolha da heurística é fundamental para a garantia da qualidade da solução e depende do conhecimento da instância a ser tratada e das suas características quanto à configuração e tamanho da rede, às hipóteses sobre o problema, à criticalidade dos recursos e do objetivo que se pretende.

Alguns procedimentos analíticos, lançam mão de uma heurística, seja na determinação de uma solução inicial que virá a ser melhorada até a ótima, com o intuito de acelerar o processo de otimização, ou na determinação de uma solução aproximada, quando a ótima é inviável computacionalmente.

CAPÍTULO III

PROCEDIMENTOS HEURÍSTICOS PARA ALOCAÇÃO DE RECURSOS LIMITADOS

A tarefa de programar um conjunto de atividades, tais que as relações de precedência e as restrições sobre os recursos sejam satisfeitas, enquadra-se na categoria de problemas matemáticos conhecidos como problemas combinatoriais. Isto é devido a que, para uma dada instância do problema, existe um grande número de possíveis combinações das datas de início das atividades, cada qual representando uma diferente programação para o projeto. A enumeração de todas as alternativas e busca daquela representando a melhor solução, de acordo com o objetivo estabelecido, é proibitivo para o caso de situações reais, com centenas de atividades, demandando mais que um tipo de recurso limitado.

Os procedimentos heurísticos, para resolução ao problema de alocação de recursos limitados na programação de atividades foram propostos logo após o aparecimento dos modelos PERT/CPM, já que estes ofereciam programações ótimas relativas à variável tempo, porém inviáveis devido à não consideração da disponibilidade dos recursos.

As heurísticas, ou regras usadas na obtenção de solução para o problema, são esquemas para designar prioridade às atividades, orientando decisões sobre ressequenciamento, necessário para resolução de conflito pelos recursos limitados.

Baseiam-se nas datas de início e término mais cedo e mais tarde das atividades e na folga, extraídas do cálculo do

caminho crítico e consideram o horizonte de programação, como uma série de intervalos discretos no tempo (períodos), para cada qual é especificada o limite dos recursos.

Existe uma classificação geral dos procedimentos heurísticos, conforme sejam feitas as decisões sobre o ressequenciamento das atividades conflitantes para a construção da programação.

Se todas as atividades do projeto são classificadas num único grupo, segundo uma heurística e então programadas, uma por vez, enquanto houver recursos e aquelas que não puderem ser programadas na sua data de início mais cedo, são progressivamente atrasadas até que haja recursos suficientes, diz-se que o procedimento é SERIADO. Nestes procedimentos as decisões locais são tomadas, respeitando a sequência original, com as atividades agrupadas por datas mais cedo de início e ordenadas segundo a heurística. Nenhuma análise é feita durante o processo de programação.

No método PARALELO os grupos de atividades a serem ordenados são formados ao longo da construção da programação. O grupo de atividades elegíveis, formado pelas atividades que tenham satisfeitas as relações de precedência, é ordenado, segundo a heurística, após análise da situação definida até aquele período.

Hooper, segundo referência em Battersby e Carruthers (66), define este método como SERIADO PARALELO, classificando como PARALELO, aquele que, a cada atividade programada, refaz a análise sobre a situação para então ordenar as atividades elegíveis para escolha da próxima a ser programada.

Dos estudos comparativos destas rotinas, extraem-se conclusões contraditórias: Pascoe desenvolveu experimento so-

bre redes de projetos, geradas segundo características externas (relacionadas a limitação de recursos) e internas (complexidade, densidade e outras relativas à duração das atividades e ao número de tipos de recursos), com vinte a cem atividades e conclui que: métodos paralelos são superiores aos métodos seriados. Gordon (83) trabalhou sobre quarenta e sete redes, de 12 a 120 atividades, obtidas de diferentes fontes e nenhuma sendo teórica. Na seleção das redes consideradas no seu experimento, impôs que os recursos requeridos por uma atividade deveriam ser reunidos para a sua execução e codificados como de um tipo. Classifica as redes segundo 9 fatores, numa escala de 0 a 140, com a maioria situando-se no intervalo de 10 a 70. Em termos gerais, redes curtas e carregadas possuem valores baixos e redes longas e esparsas são identificadas com altos valores. Os resultados são relacionados à extensão da duração do projeto, além da definida pelo CPM-padrão, devido às restrições dos recursos. Sua experiência indicou que os métodos seriados produzem programações de duração menor para redes das categorias mais baixas, que o método paralelo, e que são mais suscetíveis à características da rede, apresentando zonas marcadas de melhor performance.

A categorização de procedimentos heurísticos em métodos paralelos e métodos seriados baseia-se na distinção metodológica entre as rotinas de ressequenciamento, que resultam em diferenças no tempo computacional para a definição das programações: a análise sobre as atividades a serem programadas, frente às atividades já programadas envolve recálculo das datas e folgas e da heurística estabelecida para a ordenação das atividades. O método paralelo, mesmo requerendo maior reforço (e tempo) computacional para ressequenciamento das atividades, é

mais amplamente usado, sendo empregado em um grande número de programas comercialmente disponíveis para programação de projetos.

Apresentamos, a seguir, alguns procedimentos heurísticos citados por Davis (73) e Herroelem (72), em seus trabalhos sobre o estado da arte sobre o problema de programação de projetos com recursos limitados.

O problema de alocação de recursos limitados definidos para atingir a duração mínima de uma programação viável, teve uma primeira proposta de solução, engendrada como de nivelamento de recursos e feita por Kelley.

Utiliza o artifício de deslocamento das atividades, dentro da folga, até baixar o nível de demanda à disponibilidade imposta. Se com isso, o limite não é atingido, então o procedimento atua sobre as atividades críticas, programando primeiro aquela que provoca maior redução na demanda por recurso, por unidade de tempo de aumento na duração do projeto.

O algorítmo proposto por ele admite decisões em série e/ou em paralelo e interrupção das atividades, se necessário, além de opções de expansão ou aceleraçaõ na duração das mesmas, com correspondente diminuição ou aumento no requisito de recurso por unidade de tempo.

O método seriado de Kelley produz resultados dependentes da ordem pelas quais as atividades são programadas.Por esta razão, sugere repetir o procedimento para diferentes ordens de ressequenciamento das atividades, para então optar pela que apresenta menor duração.

Apresenta também a possibilidade de trabalhar com um nível inicial de requisito de recurso para a atividade,que

poderá, a frente, ser modificado.

O programa de execução do algorítmo de Kelley é capaz de manusear um único grupo de recursos, com até 4 tipos requeridas por atividade e nove para todo o projeto.

Fehler propõe um método, no qual primeiro são programadas as atividades elegíveis que não requeiram nenhum recurso escasso. Então, consideram-se as atividades elegíveis competindo por recurso limitado e no caso de não se poder programá-las nas suas datas início mais cedo, constroe-se uma sequência de duas delas. Calcula-se a data mais cedo de término do projeto. A sequência é invertida e recalcula-se o término do projeto. A primeira atividade da sequência levando à maior duração do projeto é rejeitada e a segunda é combinada em nova sequência com outra atividade elegível. Repete-se o processo até que uma única elegível permaneça, que é escolhida para ser programada. Volta-se ao início e repete-se o procedimento até que todas as atividades tenham sido programadas.

Riester e Schwinn propõe um método paralelo onde as atividades elegíveis (AE) são ordenadas segundo três regras de prioridade. O procedimento pode utilizar-se de um programa de simulação de Monte-Carlo, que permite a obtenção de uma idéia sobre a eficiência de determinada ordenação proposta e continuamente, melhorar a busca das soluções.

Um algorítmo, para o caso não-preemptivo e sem opção de mudança na duração das atividades, é o algorítmo de Brooks (BAG). É um algorítmo menos flexível para a resolução do problema, já que os dados de entrada, relativos à duração da atividade, referem-se a uma única duração e não admite a

possibilidade de interrupção de atividade, depois de seu início programado. (Whitehouse, 83)

Tanto a abordagem de Kelley quanto a de Brooks são específicas para um único projeto.

Já o algorítmo utilizado em RAMPS (Resource Allocation and Multi Projects Scheduling), trata com o caso de múltiplos projetos simultaneamente e programa cada atividade, tal que as datas de término e o nível de uso de recursos seja minimizado, considerando as restrições impostas sobre os recursos.

O método divide o problema geral em vários subproblemas: em cada período de tempo o sistema examina todas as tarefas elegíveis e já programadas e analisa as várias combinações possíveis para programação, considerando atraso e interrupção de atividades já programadas, dentro das restrições de recursos. A seleção da combinação é feita considerando-se: 1) prioridades e objetivos (folga total, recursos ociosos, início e término das atividades tão cedo quanto possível, prioridade às tarefas críticas, maximizar número de atividades em processo, etc...); 2) o custo associado; 3) atraso indicado para cada projeto; 4) a importância relativa a cada projeto em questão. Como suporte a esta análise, três conjuntos de dados são requeridos para cada atividade: quantidade de recursos requerido, tempo e custo de interrupção, bem como taxas de utilização e produtividade associada. Para cada projeto é necessário ter as informações sobre data de início e de término desejada, penalidade por atraso ou outra medida de ponderação para priorização entre os projetos. O procedimento minimiza custos período-período, mas não necessariamente para toda duração do projeto.

Os objetivos da programação, disponíveis para o usuário, são colocados em termos de continuidade do trabalho, ociosidade dos recursos e conforme as ponderações estabelecidas pelo usuário, permitem a ocorrência de várias possibilidades de programação. Tem capacidade de manusear até 700 atividades, 6 projetos e 60 tipos de recursos. Os principais resultados produzidos são: uma programação para cada projeto,incluindo dados sobre custos e recursos e uma análise sobre o uso dos recursos por tipo e período.

Se não for possível o estabelecimento de uma programação viável, é indicada a restrição causadora (Lambourn, 63).

O procedimento proposto por Fendley, também sob a abordagem de múltiplos projetos, é baseado na programação-tempo, obtida pelo PERT. Devido à incerteza,associada à duração das atividades,propõe que o desenvolvimento da programação seja baseado numa análise dinâmica sobre a programação estabelecida, sempre tendo como referência o objetivo que se deseja atingir.

A sua proposta é direcionada na construção de um sistema de programação on-line, para uso de um computador digital de tamanho médio.

Partindo da hipótese que, uma organização que trabalhe com projetos não aguarda o término dos projetos em progresso,para então iniciar a programação de novos empreendimentos , sugere que a cada novo projeto, após o desenvolvimento de sua rede,seja executada a análise PERT.

Sobre as atividades dos projetos em andamento e da programação - PERT para as atividades do novo projeto, propõe, então, a construção do DCR e o cálculo do atraso máximo permitido, se o prazo de término foi estabelecido externamente e

o atraso máximo necessário, de acordo com a disponibilidade de recursos. (DCR_k é o diagrama de carga de recursos construído pelas demandas periódicas de cada recurso k da programação mais cedo). Esta previsão é estabelecida de acordo com os fatores baseados na carga total de recursos.

Esta análise dá subsídio à definição de datas de entrega mais realísticas, já que negociada com os clientes, darão retorno para identificação de prioridades diferenciadas entre os empreendimentos. Propõe, então, uma programação com alocação de recursos baseada na regra de menor folga, na ocorrência de conflitos. Sugere que o controle sobre o encaminhamento da programação possa ser mantido periodicamente (diário, semanal ou mensal), de acordo com a dinâmica da organização, e para o caso de um sistema em tempo real ele pode ser feito sem periodicidade marcada.

Esta metodologia para programação de multiplos projetos, com alocação de recursos limitados, pode ser usado para programação de produção em lote.

Outra proposta para multi-projetos é a de Mcgee e Markarian. Inicia com uma formulação trade-off-tempo X custo, com a hipótese de relação linear entre custo mínimo (duração máxima) e custo máximo (duração mínima). Mão de obra é especificada por classe e restrições são impostas por cada intervalo de tempo.

Parte-se da programação CPM-padrão, usando a duração acelerada para as atividades do projeto e, se ocorrer excesso às restrições impostas aos recursos, as atividades são expandidas por sua folga, buscando a viabilidade da programação.

Um dos mais compreensivos procedimentos para alocação de recursos limitados em programação de projetos foi desenvolvido por Wiest: Os programas I e II do sistema SPAR (Scheduling Program for Allocation Resource), com uma versão chamada SPARTAN (Scheduling Program for Allocation Resources to Alternativa Networks) desenvolvida especialmente para aplicação ao problema de avaliação de alternativas para o sistema de armamento do governo americano.

O SPAR serve à programação de múltiplos projetos e segue a metodologia de rotinas paralelas para decisões sobre ressequenciamento, análogo ao RAMPS. As atividades são ordenadas por sua folga total, em cada período da programação. Uma característica peculiar a este sistema é um procedimento probabilístico para seleçaõ da atividade a ser programada, dentre aquelas elegíveis e ordenadas para o período. Desta forma, a sequencia da programação irá variar aleatoriamente, e, mesmo para mesma instância, se processada mais de uma vez, poderá indicar resultados diferentes que serão avaliados,segundo um critério específico.

A cada atividade é designado três níveis de recursos: normal; máximo e mínimo, correspondendo às durações normal; acelerada e expandida. A alocação de recursos é especificada,automaticamente pelo programa, de acordo com a criticalidade da atividade e disponibilidade de recurso.

O programa interage a cada dia, selecionando e ordenando atividades elegíveis por folga. Se a tarefa selecionada não é crítica, a primeira tentativa é feita para programá-la em sua duração normal. Se não for possível, a opção de expandi-la é considerada, antes de ser necessário atrasá-la para consideração no próximo período. Se a tarefa é crítica, tenta

tiva é feita para programá-la acelerada e se não for possível, tentar programá-la na sua duração normal. Assim, o sistema força a atividade crítica, para seus modos acelerada e normal, prioritariamente. Caso nenhum modo de executá-la é possível, aciona-se uma sub-rotina que interrompe atividades ativas (em programação) para liberação de recursos, desde que esta interrupção não atrase o projeto (ou seja, há folga). Usa subrotina para reprogramação destas atividades até sua data de início mais tarde. Caso ainda não seja possível programar a atividade crítica, nas durações acelerada ou normal, é tentado programá-la com o nível mínimo de recursos, antes da decisão de atrasá-la.

Após todas as tarefas serem programadas no período, aciona-se a sub-rotina de aproveitamento de recursos ociosos. A lista de atividades ativas no período, que não estão programadas de forma acelerada, é ordenada pela maior folga e os níveis de recursos são aumentados, até que se tenha chegado nos limites impostos para o período.

Quando passa à iteração seguinte, as atividades críticas ativas que não tiveram os níveis máximos designados, são analisadas, frente à disponibilidade dos recursos e se possível, programadas, agora, de forma acelerada.

A cada interação são recalculadas as datas pela análise do caminho crítico, posto que as ações sobre as atividades críticas podem ter consequências sobre as atividades não programadas da rede.

Quando todas as tarefas tiverem sido programadas são calculados os custos associados à programação proposta, quais sejam: despesas com horas extras, multas e/ou gratificações, custos de recursos para os níveis impostos. Conforme os resulta-

dos, comparado com outros processados, termina-se o problema definindo-se uma solução, ou novo processamento é executado. O SPAR-1 foi programado em FORTRAN para o IBM-7094 e o CONTROL-DATA G-20 e é dimensionado para manusear um projeto com 1200 atividades com recursos únicos, 500 eventos e 12 tipos de recursos sobre um horizonte de programação de 300 dias. A solução, para tratar atividades com múltiplos recursos, é criar pseudo-atividade para cada recurso e restrições sobre datas iguais de início e mesmo nível de consumo dos recursos para estas atividades. Wiest desenvolve, posteriormente, uma adaptação para o SPAR opera em sistema on-line, utilizando display gráfica, visando a interação homem-máquina na tomada de decisão sobre programação de recursos.

A primeira tentativa de desenvolvimento de procedimento heurístico para tratar com mais de um recurso por atividade, sem necessidade de desagregação da mesma, foi desenvolvida por Butler, em 1961, no MIT como dissertação de Mestrado, empregando uma função heurística de prioridade.

A definição de função heurística para nortear o desenvolvimento de solução para problemas combinatoriais, atualmente é objeto do escopo da Inteligência Artificial, para construção de Sistemas Especialistas.

Brand, Meyer e Shafer propõe o método de Programação de Recursos (RSM - Resource Scheduling Method), resolvendo o conflito entre duas atividades e selecionando o ressequenciamento entre elas, de forma a obter-se o menor aumento na duração do projeto, análogo ao proposto por Fehler. Para a identificação da melhor sequência, é incorporada à rede uma atividade fictícia indicando esta relação adicional de precedência. Foi implementado no PCS (Project Control System)

para o IBM 1130. Suporta a programação de projetos com até 200 atividades, com cada tarefa requerendo até seis diferentes tipos de recursos.

Um primeiro sistema interativo para o problema é apresentado como sendo o de Leahy, no qual mensagens de ocorrência de conflito são apresentadas on-line, assim como especificadas alternativas para resolução do mesmo. Este sistema é capaz de manusear até 200 atividades, requerendo cada uma até 10 diferentes tipos de recursos. Fica ao usuário a opção de escolha da alternativa, podendo inclusive simular várias para comparar os resultados.

Este esquema é o encontrado no software "Super--Project-V-1.0" para micros-PC, 16 bytes. A versão 2.0 deste software, apresenta a opção de nivelar automaticamente os recursos para obediência ao limite imposto, segundo três regras de prioridade, consideradas na ordem: folga disponível, data de início mais cedo e prioridade na alocação de recurso, ou na ordem: prioridade na alocação de recursos; folga disponível e data de início mais cedo.

Thesen (78), apresenta o programa WASP (Winsconsin Activity Schedulin Program), escrito em FORTRAN IV e possível de ser implementado com requisitos baixos de memória. Utiliza um processo de escolha das candidatas elegíveis, a cada período, pela resolução de um problema de mochila, definindo um fator de urgência para ponderação dsa atividades. Estes fatores podem ser ajustados pelo usuário que, assim, terá facilidade sobre o controle da programação completa a ser obtida.

É apresentado a seguir, algoritmo base para a maioria dos procedimentos apresentados acima, inclusive os utilizados nos programas comerciais.

O Problema

O problema a ser considerado é o problema padrão de programação de projetos com restrição de recursos. Considera um conjunto P de projetos ordenados pelo comprimento do caminho crítico, em ordem ascendente, tal que CP_p = MAXCPL. Cada projeto é um conjunto A_i parcialmente ordenado de N_i atividades- A cada atividade j é associado: uma duraçaõ inteira não negativa d_{ij}; um requisito não-negativo por r_{ikt} unidades do recurso k em cada período t, no qual a atividade está em progresso, para cada um dos k tipos distintos de recursos usados no projeto i, e demandado pela atividade; um conjunto de atividades predecessoras imediatas P_{ij} e uma data de início s_{ij} para ser determinada.

Desde que A_i é parcialmente ordenado, $G(A_i, p)$ é um grafo direcionado acíclico e define para cada atividade j, um conjunto P^*_{ij} de todas as atividades predecessoras, um conjunto S^*_{ij} de todas as atividades sucessoras, e um conjunto S_{ij} de todas as sucessoras imediatas de j.

Hipóteses

Para determinação de s_{ij} é necessário satisfazer as restrições:

a) Precedência - Para todas as tarefas j em A_i,

$$s_{ij} \geq 0, \text{ se } P_{ij} = 0$$

$$s_{ij} \geq s_{ij} + d_{il}, \forall\, l \in P_{ij}, \text{ se } P_{ij} \neq 0$$

b) Limite de Recursos. A demanda por recursos em qualquer período não deve exceder a disponibilidade . Assim:

$$\sum_i \sum_j x_{ijt}\ r_{ijkt} \leq R_{kt} \quad \text{para todo } k \text{ e } \forall\ t$$

onde R_{kt} é a quantia disponível do recurso k em t

H_{ij} é o horizonte de programação de j

$$x_{ijt} = \begin{cases} 1 \text{ se a atividade } j \text{ é ativa no período } t \\ 0 \text{ caso contrário} \end{cases}$$

c) Consideraremos, ainda, a hipotese de tratarmos com o caso não preemptivo, para o qual a atividade, uma vez iniciada, não pode ser interrompida. Assim, se o início da atividade j é programado para s_{ij}, então

$$x_{ijt} = 1 \quad \text{para } t \in \{s_{ij},\ s_{ij}+1,\ \ldots,\ s_{ij}+d_{ij}-1\}$$

Assim, agregamos à hipótese (b) a restrição para todo $t \in H = \{s_{ij},\ s_{ij}+d_{ij}-1\}$

Nesta formulação estão incluída as hipóteses dos recursos serem discretos e renováveis. Quando a limitação do recurso for constante para todos os períodos, então $R_{kt} = R_k$, $\forall$ t.

Objetivos

Para o caso de múltiplos projetos, os objetivos a serem minimizados podem ser definidos como:

. Atraso Total - soma dos atrasos verificados em cada projeto

$$\sum_i (H_i - CP_i) \quad \text{onde } H_i = \max_j \{ s_{ij}+d_{ij} \}$$

. Atraso Ponderado - uma ponderação do atraso pode ser forçada, se houver interesse em priorizar a programação de algum projeto. Sendo PEN_i penalidade associada ao projeto i, teremos

$$\sum_{i} PEN_i \ (H_i - CP_i)$$

Definições

Uma programação parcial T^c, no período t, é a especificação de datas de início para um subconjunto AP^t de atividades programadas, tal que se a atividade y ε AP^t, então P_y C AP^t, de forma que nenhuma restrição tenha sido violada.

Um subconjunto AA^t C AP^t é constituído das atividades em progresso, ou seja, x_{ijt} = 1 para o período t.

Para aumentar o tamanho de T^c, devemos determinar data de início s_{ij}, satisfazendo as restrições. Esta atividade candidata a integrar T^c, não pertence a AP^t, mas P_{ij} C AP^t.

Assim, definimos o conjunto AE^t, das atividades x elegíveis (ou candidatas), tal que

$$AE^t = \{ x \mid x \notin AP^t \wedge P_x \ C \ AP^t \}$$

Para a programação parcial T^c, define-se τ, de forma a não violar nenhuma das restrições. O controle sobre a disponibilidade dos recursos é feito por:

$$\overline{R}_{kt} = R_{kt} - \sum_{AA^t} (r_{ilt} + r_x) \text{ para todo k e l } \varepsilon \ AA^t$$

$$t \ \varepsilon \ [t, \ \max_{AA^t U \{x\}} (s_{il} + d_{il} - 1)$$

Desta forma, para alguma atividade x ε AE^t e satisfazendo as restrições, define-se

$$s_x = \tau$$

$$AA^\tau = AA^t U \{x\}$$

$$T^c_\tau = T^c_t U \{x\}$$

Algoritmo Base

O método paralelo cria uma programação completa, através de uma série de programações parciais.

Seja para um período t, uma programação T^c como definida anteriormente. A essência do procedimento heurístico é a escolha da atividade x, entre as elegíveis, sujeito a algum critério de seleção.

Sob este critério define-se o conjunto ordenado AO^t, formado pelas atividades candidatas.

Definimos a função heurística, segundo f = g + h.

ALGORITMO 1 - Método Paralelo

---

início: t = 0; AP ≠ 0

IT1: t; Crio AE^t. Se $AE^t = 0 \rightarrow$ FIM

Caso contrário VÁ para IT2

IT2: Crio $AO^t = f\ (AE^t)$

IT3: Defino AP^t e $t = \tau$. VÁ para IT1

ALGORITMO 2 - Método Seriado-Paralelo

---

início: t = 0; AP = 0; Crio AE^0 e AO^0

$\overline{IT1}$: t; Crio AE^t. Se $AE^t = 0 \rightarrow$ FIM

Caso contrário VÁ para IT2

$\overline{IT2}$: Crio $AO^t = f\ (AE^t)$

$\overline{IT3}$: Escolho x de AO^t, otimizando f(x) = g(x) + h(x)

$\overline{IT4}$: Agrego x a T^c. Defino $t = \tau$. VÁ para $\overline{IT1}$

Algoritmo Seriado

A essência do algoritmo seriado é esgotar todas as atividades relacionadas por ordem de precedência.

ALGORITMO SERIADO

IT1: t = 0

AE^t = (lista das atividades por ordem de precedência e de menor folga)

IT2: Programa as atividades de AE^t com folga igual a zero. Atualiza t pela viabilidade de recurso.

IT3: t; AE^t é ordenada por função heurística gerando o conjunto AO^t.

IT4: Programar as atividades do conjunto AO^t em ordem sequencial. As atividades que não puderem ser programadas, são inseridas no final da lista. Atualiza t, por disponibilidade de recurso.

IT5: Se $AO^t = 0 \rightarrow$ FIM.

Caso contrário vá para IT4.

A proposta de Thesen (76), define o "fator de urgência" usado na iteração 2, (IT2), para resolução do problema da mochila, apresentado como:

$$\text{Max } z = \sum_{AE^t} c_i \, x_{it}$$

$$\text{s.a. } \sum_i r_{ik} \, x_{it} + \bar{R}_k^t \leq R_k$$

$$\text{e } x_{it} = \begin{cases} 1 \text{ se } i \in AA \\ 0 \text{ caso contrário} \end{cases}$$

onde $\bar{R}_k^t$ - demanda por recurso k, no período t

c_i - fator urgência da atividade $i \in AO$

A diferença básica, desta proposição, frente ao algoritmo base é a combinação de atividades para seleção conjunta, para integração na programação parcial, e não o sequenciamento das atividades pelos seus valores individuais.

Cooper (76) apresenta outra variante do procedimento base, priorizando as atividades de AA, em $\overline{IT3}$, de forma aleatória, com o seguinte método: a prioridade da tarefa i em AO pertencendo ao conjunto AA é dada por $\frac{P(i)}{\sum_{a \in AO} P(a)}$, ou seja, pondera o valor da heurística da atividade i, pela somatória de seus valores para todas as atividades em AO.

Em experimento, comparando este procedimento e um método paralelo, verifica-se que o método amostral produz programações com duração ao menos 7% menores que as produzidas pelo método paralelo.

Holloway et al apresentam um procedimento de múltiplos estágios baseados na decomposição do problema de programação de projetos sob recurso limitado, do tipo 1/1/1 e n/1/1 para o caso não preemptivo, podendo ser aplicado para os problemas 1/n/1 e n/n/1, quando cada atividade pode ser alocada à capacidade disponível do recurso requerido. Tem como objetivo a obediência à data estabelecida. Para o caso de minimizar a data de término o programa iterativamente procura acelerar o projeto. Ou seja, com uma data T definica pela heurística de morfologia, processa o projeto buscando a viabilidade para T-1. E assim, iterativamente até chegar a inviabilidade no período τ.

A decomposição é feita em subproblemas definidos por recurso e a estratégia de controle é pela definição de "force signals" criados pela violação de restrição, estruturadas segundo uma hierarquia.

Tabela III-2. Estrutura de classe de Violação de Restrições (Holloway et al, p. 864).

CLASSE	TIPO DE VIOLAÇÃO PERMITIDA	TIPO DE RESTRIÇÃO IMPOSTA PARA FORÇAR UMA OPERAÇÃO
1	Nenhuma	-
2	IT	IC
3	IC	ITP-P
4	ITP-P	ICP-P
5	ICP-P	ITP-P

Restrições de Capacidade de Recurso e restrições de datas estabelecidas para o projeto são expressas pelas datas de início mais cedo possível (ICP) e início mais tarde possível (ITP) para cada atividade. Estas restrições são chamadas de "hard" e não podem ser violadas no processo de programação. As Restrições de Precedência são refletidas pe-

las datas de início mais cedo (IC) e datas de início mais tarde (IT) para cada atividade. São as restrições "soft" que poderão ser violadas, se necessário. Se violadas é necessário reprogramar as atividades adjacentes. Atenção especial é dada a estas violações pelo "sinal de força", transmitido por meio de pseudo-restrições (ICP-P ou ITP-P). O procedimento trata as pseudo restrições como intermediárias, em termos de rigidez, entre as do tipo "hard" e as do tipo "soft".

O uso destes "sinais de força" oferece um mecanismo para o programa trabalhando com um dado sub-problema (recurso), ajustar-se e eliminar violações que tenham sido resultantes da programação de outros subproblemas.

A hierarquia para violação das restrições, correspondendo a nunca violar restrições tipo "hard" (classe 1), violação de restrições tipo "soft" (classes 2 e 3), e violação das pseudo-restrições (classes 4 e 5) é estabelecida.

Na resolução de um dado subproblema as atividades que podem ser programadas na classe 1, são programadas primeiro.

Se todas as atividades requerendo este recurso podem ser programadas na classe 1, a programação começa pelo próximo recurso. Caso contrário, uma escolha é feita, usando regras de prioridade, entre: (1) reivsão da programação parcial corrente e tentando não violar quaiquer restrições, ou

(2) permitindo que atividades das classes superiores sejam programadas, envolvendo, então, violação de restrições.

Uma hierarquia dos modos de programação de cada atividade, paralelamente à estrutura de classes das violações às restrições, é estabelecido.

Combina a hierarquia dos modos de programação das atividades à hierarquia das restrições, de modo progressivo e dentro de cada modo o conjunto de restrições preferido (classe da hierarquia) é mantido. Antes de mudar de um modo para o próximo, uma rotina backtracking é acionada e oferece um meio de tentar reprogramar os recursos dentro da programação corrente.

Um experimento envolvendo 23 redes de projetos únicos, cada um dos quais com 30 a 60 atividades foi projetado para comparar sua efetividade (tempo computacional) e eficiência (qualidade da solução) com o método de Branch and Bound (Schrage e Stinson), e com o SPAR e a conclusão apresentada foi de que: para minimização do tempo de término do projeto, os resultados do PSP são mais rápidos e efetivos, quando comparados com o método de B.B. e superiores em termos de qualidade da solução, que o SPAR.

A apresentação deste procedimento é feita para ilustrar outra metodologia utilizada para desenvolvimento de uma programação, baseada em heurística, e tratando com modos alternativos de execução da atividade, que não o método paralelo. Baseia-se na decomposição do problema e para cada subproblema define estrutura de classes para as restrições, combinada com hierarquização do modo de execução das atividades, sendo processado em múltiplos estágios.

O uso de heurísticas pode estar associado a algum procedimento analítico, para aceleração da busca de solução ótima. São apresentados abaixo, alguns destes procedimentos:

Algoritmo de Enumeração Limitada

[DAVIS & HEIDORN (71)]

n/n/n

Problema de um único projeto.

Abordando restrições de: múltiplos recursos, datas designadas, substituição de recursos. Caso não preemptivo. Atividade designada por d_j. Atividades de duração unitária, com restrição que devem ser executadas em seguida. Grafo de Busca e Árvore de Busca. O Grafo de Busca expressa todas as programações parciais possíveis (definidas pelas relações de precedência) e define subconjuntos viáveis definidas pelas restrições de recursos. A enumeração é limitada pelo uso de uma heurística, avaliando a potencialidade da programação viável no traçado da Árvore de Busca.

Algoritmo de Limite Mínimo

[PATTERSON & HUBER (74)]

n/n/n

Projeto único.

1. Estabelece um limite inferior para término do projeto T, maior inteiro positivo definido por

$$W = \max \{ CP, \max_k [\frac{\Sigma r_{jk} d_j}{R_k}] \}$$

2. Busca programação viável pelo algoritmo 0 - 1 [Bowman's] . Se conseguir viabilidade para T → FIM.
3. Caso contrário. Substitui T por T+1 e retorna à etapa 2.

Algoritmo de Limite Máximo

[PATTERSON & HUBER (74)]

n/n/n

Projeto único.

1. Determina programação viável por alguma heurística. T é associado ao limite superior.
2. T = T+1. Busque programação viável pelo algoritmo de programação 0 - 1. Se não existir, a heurística deu solução ótima → FIM.

   Caso contrário, faça T = T', valor da programação obtida, e vá para 1. Repita até que a inviabilidade seja atestada pelo algoritmo 0 - 1.

Algoritmo de Pesquisa Binária

[PATTERSON & HUBER (74)]

Combinação dos algoritmos de limite máximo e limite mínimo e execução de pesquisa binária no intervalo entre os dois valores.

CAPÍTULO IV

MEDIDAS PARA CARACTERIZAÇÃO DO PROBLEMA

Fatores de caracterização de uma rede de atividades são usados para analisar a dificuldade do problema e estimar o tempo computacional para resolvê-lo através de determinada técnica de solução e/ou validar comparações entre heurísticas alternativas e/ou prever "durações" para o projeto sobre os dados padrões da rede (durações das atividades e lógica - relações de precedência) e os requisitos e disponibilidades de recursos.

Herroelen e Elmaghraby (80) discutem o propósito de se construir uma medida para auxílio à resolução do problema questionando: se a sua necessidade deve ser voltada para medir a "qualidade" da rede ou do procedimento usado na análise da mesma. Chamam a atenção, ainda, para a necessidade de apontar o objetivo para análise da rede, para então, definir os fatores para sua caracterização.

A programação de atividades, sob relações de precedência e limitação de vários recursos é o objetivo para caracterização da rede neste trabalho. Segundo os autores acima citados "este é, talvez, o problema central nos estudos de Rede de Atividades, e um dos mais difíceis de resolver". Na visão deles, "a resolução do mesmo deveria esperar a realização de aberturas de caminho no campo da otimização combinatorial". Por isso acham prematuro tentar medir a complexidade da rede

para este problema, sob número e quantidade arbitrárias de recursos tratados.

Embora isso, sugerem que uma boa medida para análise do problema de programação com recursos escassos, seja construída de forma a relacionar características da estrutura da rede e da disponibilidade de recursos já que consideram insuficiente a análise sobre a estrutura da rede.

Apresenta um gráfico que fundamenta sua sugestão e a seguinte argumentação:

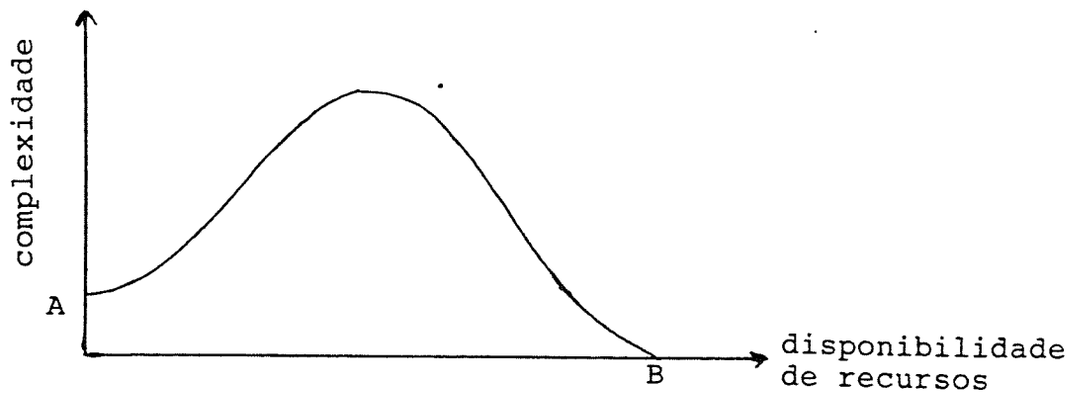

FIGURA 4.1 - Complexidade da rede versus disponibilidade de recursos [Elmaghraby e Herroelen (80), pg.231]

"Se os recursos são disponíveis em quantias extremamente pequenas, existirá pouca liberdade em programar as atividades (por exemplo, as atividades deverão ser sequenciadas em série e a resultante duração do projeto será igual à soma das durações das atividades), daí a complexidade deverá ser pequena (Ponto A). Se, por outro lado, recursos são amplamente disponíveis, as atividades poderão ser programadas em paralelo, e a duração resultante do projeto será igual ao compri-

mento do caminho crítico. Daí a complexidade seria igual a zero (Ponto B)". [Elmaghraby & Herroelen (1980), pg.230].

Os procedimentos básicos PERT/CPM, que produzem programações detalhadas para as atividades do projeto, são limitados na medida em que não se considera restrições impostas aos recursos. Estas abordagens básicas assumem unicamente a lógica (estrutura de precedência) e as durações das atividades, como determinantes para a programação do projeto.

A aplicação destes procedimentos tem como resultado datas de início e término mais cedo e maistarde para as atividades e dados sobre as folgas existentes.

Quando o problema de restrição de recursos é colocado, estes elementos dos procedimentos básicos do CPM-padrão são afetados de modo significativo e, a necessidade de ressequenciamento de atividades conflitantes no uso de recursos escassos, leva à existência de diferentes opções sobre datas de início mais cedo (IC_j) para atividades do projeto.

O processo de decisão sobre designação destas datas, levando ao ressequenciamento das atividades é o cerne do problema de alocação de recursos. Um procedimento heurístico tenta otimizar, em termos computacionais, este processo de busca de soluções viáveis. Para isso recorre a elementos extraídos da análise dos recursos sobre a instância tratada.

A análise sobre a alocação de recursos requeridos para execução das atividades na programação de projetos é baseada na relação entre demanda por recursos em cada período do projeto, segundo uma programação inicial (normalmente usa-se a programação mais cedo) e a disponibilidade dos recursos. Po

de ocorrer que os limites periódicos impostos aos recursos sejam diferentes ou constantes ao longo do horizonte de programação.

A demanda, por cada recurso, pode ser mostrada pelo DIAGRAMA DE RECURSO (DCR_k). Este diagrama é plotado a partir da soma das demandas por recurso k, das atividades programadas para um dado período.

Seja t um período de programação, $t = \overline{1, CP_m}$ onde

CP_m = MAX CPL (comprimento do mais longo projeto)

S_t = conjunto de atividades programadas para este período

$D_{kt} = \sum_{j \in S_t} r_{jkt}$ = demanda total periódica por recurso k no período t

O DCR_k é o histograma demonstrativo das demandas totais, período a período, de uso do recurso k, para os projetos.

A partir do DCR_k constrói-se a CURVA CUMULATIVA DE RECURSOS (CCR_k) pela acumulação das demandas totais até cada período.

Assim teremos para o período T,

$A_{kt} = \sum_{t=1}^{T} D_{kt}$, representando a demanda ocorrida até a data T, pelo uso do recurso k, por todas as atividades programadas (incluindo as completadas e as em progresso).

Estas curvas oferecem a visualização do uso dos recursos para uma dada programação: localizam-se os períodos de

pico de demanda, os períodos de excesso ao limite imposto, os de ociosidade, relaciona-se a demanda média e a taxa de utilização programada (comparação entre a configuração da CCR_k e a sua bissetriz).

As experiências sobre o problema de alocação de recursos limitados em programação de projetos mostra claramente que a efetividade dos procedimentos para sua solução são dependentes das características particulares da instância tratada.

Estudos tem sido desenvolvidos para extração de parâmetros para análise da estrutura do problema em si, ou seja, complexidade, densidade, tamanho, configuração das redes do projeto e sobre a criticalidade dos recursos. Não há ainda, uma metodologia amplamente aceita para categorização dos problemas, embora a sua necessidade é reconhecida desde os primeiros desenvolvimentos de procedimentos para solução do problema e muitos parâmetros são usados para teste das proposições existentes. Estes testes dizem respeito a efetividade do procedimento analítico. (tempo computacional requerido) ou previsão, no caso de heurísticos, de qual delas apresenta melhor resultado, segundo características da instância a ser tratada.

Não há, ainda, um consenso sobre a relação entre características do problema e eficiência de uma particular técnica de solução, embora as experiências obtidas tendem a convergir sobre alguns resultados.

Pascoe[1], em 65, foi o primeiro a propor um fator pa

[1] A referência neste capítulo, de autores não citados na bibliografia podem ser buscadas em Elmaghraby e Herroelen (80) e Davis (75).

ra especificar o interrelacionamento das atividades e características associadas à variável tempo e aos recursos de redes de projetos. Usou estes critérios para controlar características de rede artificialmente geradas para testar diferentes heurísticas, e relacionar suas performances com as especificidades da rede.

Johnson, em 67, propôs medidas similares às de Pascoe e usou-as para tentar desenvolver relações com o tempo computacional e o seu algoritmo de Branch and Bound para o probrema de programação com restrição de recursos. Concluiu que das medidas testadas nenhuma mostrou sigficância na previsão do tempo computacional do seu procedimento.

Fendley(68), partindo da hipótese que o tempo de programação de um projeto, depende principalmente da quantia de recursos necessários à execução dos múltiplos projetos desenvolvidos conjuntamente, relacionados à sua disponibilidade, desenvolveu uma metodologia de análise de recursos e previsão de datas de término, quando a programação básica é feita pelo PERT.

Da programação esperada desenvolvida para cada projeto, extrai-se o DCR_k para cada projeto e para a combinação de projetos, para cada recurso a ser analisado.

Definem-se os pontos de término do projeto, como pontos chaves, já que para todos os recursos, espera-se nesta data, decréscimo na demanda, devido a ocorrência de menor número de atividades competindo pelo uso dos recursos.

Para definir a demanda de recursos entre estes pontos, as durações - PERT dos projetos são arranjadas em ordem

crescente e os seguintes segmentos são criados:

$$S_1 = CP_1^e \; ; \; S_2 = CP_2^e - CP_1^e \; ; \; \ldots \; ; \; S_M = CP_M^e - CP_{M-1}^e$$

onde CP_i^e = duração esperada do projeto i.

Fazendo $a_1 = CP_{1-1}^e + 1$ e $b_1 = CP_1^e$, calcula a demanda média por período, no segmento 1 e relacionando com a disponibilidade R_k, obtém-se um indicador de criticalidade do recurso k, neste intervalo:

$$LF_{1k} = \frac{\sum_{t=a_1}^{b_1} D_{kt}}{(b_1 - a_1)\, R_k} = \frac{1}{R_k} \cdot \frac{\sum_{t=a_1}^{b_1} D_{kt}}{(b_1 - a_1)}$$

Para a obtenção de um fator de carga total, agregando todos os recursos e sobre o horizonte de programação de todos os projetos sugere três medidas:

$$F_1 = \frac{\left[\sum_{1} \{ \max_{k} (LF_{1k}) \cdot (b_1 - a_1) \} \right]}{b_M}$$

F_1 é o fator de carga total, considerando a criticalidade máxima dos recursos em cada segmento 1.

$$F_2 = \frac{\left[\sum_{1} \{ \text{mediana}_{k} (LF_{1k}) \cdot (b_1 - a_1) \} \right]}{b_M}$$

F_2 é o fator de carga total, atribuído a criticalidade mediana dos recursos em cada segmento 1.

$$F_3 = \frac{\left[\sum_{1} \{ \min_{k} (LF_{1k}) \cdot (b_1 - a_1) \} \right]}{b_M}$$

F_3 é o fator de carga total, considerando os meno-

res valores para o índice de criticalidade dos recursos, calculados por cada segmento.

Fendley sugere que os fatores de carga total F_1, F_2, F_3 podem ser usadas como variáveis independentes para previsão do atraso médio e atraso máximo, sobre todos os projetos, utilizando a heurística de menor folga para programação - PERT e alocação de recursos das atividades, com a abordagem de múltiplos projetos.

O seu meio experimental para análise da performance do seu sistema consistiu-se de oito projetos simulados, com cada um contendo até 20 atividades e cada atividade requerendo até três unidades, de cada um dos três tipos de recursos escassos. Da seleção de 3 combinações dos oito projetos e variação dos níveis de disponibilidade de 6 e 12 unidades em diferentes combinações, sugere um fator de atraso médio calculado como:

$$MF = 1.73\ F_1^2 - 1.90\ F_1 + 0.25\ F_2 + 0.006\ F_3 + 0.50$$

é um fator de atraso máximo, na seguinte forma:

$$MXF = 2.23\ F_1^2 - 2.97\ F_1 + 0.35\ F_2 + 0.08\ F_3 + 1.51$$

para: $0.70 \leq F_1 \leq 2.60$

a sua experiência, mostra que o fator de atraso médio é mais consistente que o atraso máximo. Porém, considerando que a previsão de atraso máximo é mais útil para empresas de múltiplos projetos, sua experiência contêm, ainda, testes sobre a equação de MXF para previsão de datas de término, sobre uma combinação de seus projetos testes. A acuracidade dos seus re

sultados varia entre 98 a 100%, a menos de um caso específico, cujo valor é de 13%.

O estudo de Fendley foi inovador no sentido de se propor a desenvolver um sistema de gerenciamento de múltiplos projetos, prevendo atrasos devido à limitação de recursos.

Prideaux e Cullingford, em 1969, propõe uma medida sobre a "quantidade" de tomadas de decisão sobre uma rede para programação sob recursos escasso.

Klovstad, em 1969, apresenta alguns conceitos envolvendo características tecnológicas da rede, no sentido de analisar a ordenação das atividades. Utiliza o conceito de "rank" da atividade, como sendo o número de atividades predecessoras desde o evento início da rede que teria rank zero . Construindo uma árvore, explicitando as relações de precedência, no qual cada nó pai, gera todas as atividades imediatamente sucessoras: o rank do nó pai sendo n, o rank das atividades filhas seria n+1. Utilizando este conceito, define-se uma medida de comprimento da rede, como sendo o maior rank mais um e a amplitude como sendo o número de atividades associadas ao rank apresentando maior número de atividades. Um índice para medir configuração da rede seria a razão entre as duas medidas (comprimento/amplitude).

Davis e Heidorn (71), seguindo a abordagem de Pascoe, usaram medidas para controlar as características de projetos simulados para testar seu procedimento de enumeração limitada para o problema de programação com restrição de recursos, com o objetivo de controlar a variabilidade no tempo de computação para o seu algoritmo. Com resultados análogos aos

de Johnson, concluiram ainda que, o experimento não era suficiente para rejeitar a hipótese de que nenhuma relação existe entre características da rede e o problema analisado.

Thesen, assim como Elmaghraby e Herroelen (80) buscam uma medida para refletir a quantidade de sequências viáveis que podem ser geradas de uma rede, considerado para o problema de programação sujeito à restrição de um único recurso, com uma única disponibilidade:

$S = \text{Perm } (M^C)$

onde M^C é uma matriz de posição, onde as linhas denotam as atividades e as colunas o rank da sequência representando uma programação parcial . Assim:

$M^{(c)} = [\ m_{jr}\ ]$ tal que

se j tem $s \geq 0$ sucessoras e $p \geq 0$ predecessoras, então, $m_{jr} = 1$ para $r = p+1, p+2, \ldots, N-s$.

Perm (M^C) é calculado com o Det (M^C), a menos da troca de sinais.

Davies (83) e Kaimann, similarmente a Pascoe, definem medidas de complexidade da rede, relacionando número de atividades e número de relações de precedência da rede.

A seguir apresentamos uma coletânea de medidas para caracterização do problema de programação de projetos sob recursos escassos. As Tabelas 4.1, 4.2 e 4.3 tomam como base os parâmetros apresentados por Patterson (76).

NOTAÇÃO	DESCRIÇÃO
N_i	Número de atividades do Projeto i
P	Número de Projetos
R	Conjunto das categorias dos k recursos
d_{ij}	Duração da atividade j do projeto i
r_{ijk}	Demanda Periódica do Recurso k pela atividade j do projeto i
R_k	Disponibilidade do Recurso k em cada período do horizonte da programação
CP_i	Comprimento do caminho crítico do projeto i
FF_{ij}	Folga Livre da atividade j do projeto i
FT_{ij}	Folga Total da atividade j do projeto i
s_{ij}	Data de início a ser programada a atividade j do projeto i
P_{ij}	Conjunto predecessores imediatas da atividade j do projeto i
S_{ij}	Conjunto sucessoras imediatas da atividade j do projeto i
P^*_{ij}	Conjunto dos predecessores de j em i
S^*_{ij}	Conjunto das sucessoras de j em i
r_{ilkt}	Requisito de recurso k, pela atividade l do projeto i no período t
H_{ij}	Horizonte de programação da atividade j do projeto i
H_i	Duração do projeto i sob limitação do recurso
PEN_i	Penalidade associada ao projeto i
T^c	Programação Parcial no período t
AP^t	Atividades programadas no período t
AA^t	Atividades ativas no período t (em progresso)
AE^t	Atividades elegíveis, possíveis de serem programadas
AO^t	Atividades elegíveis definidas e/ou ordenadas pela heurística

Tabela 4.1 - Parâmetros associados ao tamanho, configuração e tempo antes da análise do caminho crítico

LENGHT (Kovstad)	Profundidade
WIDTH (Kovstad)	Amplitude: número de atividades no rank possuindo maior número de atividades.
SHAPE	Razão entre profundidade e amplitude $\frac{\text{LENGHT}}{\text{WIDTH}}$
NPROJ	Número de projetos a serem programados $\sum_{P} i$
NNODE	Número de atividade a serem programadas $\sum_{P} \sum_{N_i} j$
NARC	Número de relações de precedência
NDUMMY	Número de atividades fantasmas
DUM (Davies)	Porcentagem de atividades fantasmas (duração e demanda por recurso iguais a zero) sobre o número total de atividades $\frac{\text{NDUMMY}}{100.\text{NNODES}}$
T-DENSITY (Johnson)	Densidade total de atividade $\sum_{N_i} \max \{0,$ número de atividades predecessoras - número de atividades sucessoras $\}$
$\bar{X}$ DENSITY	Densidade média de atividade $\frac{\text{T-DENSITY}}{\text{NNODE}}$
COMPLEXITY (Pascoe)	Complexidade do projeto. Relação entre número de atividades (N) e número de relações de precedência (E) N / E
COMPLEXIDADE (Davies)	$2 (N-E+1) / (E-1) (E-2)$
COMPLEXIDADE (Kaimann)	N^2 / E
S s (Elmaghraby e Herroelen)	Número de sequências viáveis $S = \text{Perm } (M^C)$
$\sum$ DUR	Soma da durações das atividades $\sum_{P} \sum_{N_i} d_{ij}$
$\bar{X}$ DUR	Duração média das atividades $\frac{\sum \text{DUR}}{\text{NNODE}}$
VA-DUR	Variância na duração $\frac{\sum_{P} \sum_{N_i} (d_{ij} - \bar{X}\ \text{DUR})^2}{\text{NNODE} - 1}$

Tabela 4.2 - Parâmetros baseados no tempo e configuração da rede, após análise do caminho crítico*

ΣCPL	Soma das durações dos caminhos críticos $\sum_{P} CP_i$
XCPL	Duração média de caminho crítico $\frac{\Sigma CPL}{NPROJ}$
VA-CPL	Variância da duração do caminho crítico $\frac{\sum_{P} (CP_i - \bar{X}CPL)^2}{NPROJ - 1}$
MAXCPL	Duração do mais longo caminho crítico $\max_{P} \{CP_i\}$
ΣSLACK	Soma das folgas total das atividades $\sum_{N} TF_{ij}$
NSLACK	Número de atividades possuindo folga total positiva $\sum_{N} \begin{cases} 1 & \text{se } TF_{ij} > 0 \\ 0 & \text{se } TF_{ij} = 0 \end{cases}$
PCTSLACK	Percentagem das atividades com folga total positiva $\frac{NSLACK}{NNODE}$
$\bar{X}$SLACK	Folga total média $\frac{\Sigma SLACK}{NNODE}$
TOTSLACK-R	Proporção de folga total sobre o caminho crítico mais longo $\frac{\Sigma SLACK}{MAXCPL}$
$\bar{X}$SLACK-R	Proporção da folga média sobre o caminho crítico mais longo $\frac{\bar{X}SLACK}{MAXCPL}$
PDENSITY-T	Densidade do projeto sobre a folga total $\frac{\Sigma DUR}{\Sigma DUR + \Sigma SLACK}$
ΣFREESLK (Johnson)	Folga livre de todas as atividades $\sum_{N} FF_{ij}$
NFREESLK	Número de atividades possuindo folga livre positiva $\sum_{N} \begin{cases} 1 & \text{se } FF_{ij} > 0 \\ 0 & \text{se } FF_{ij} = 0 \end{cases}$
PCTFREESLK	Porcentagem de atividades com folga livre positiva $\frac{NFREESLK}{NNODE}$
$\bar{X}$FREESLK	Folga livre média por atividade $\frac{\Sigma FREESLK}{NNODE}$
PDENSITY-F (Pascoe)	Densidade do projeto sobre a folga livre $\frac{\Sigma DUR}{\Sigma DUR + \Sigma FREESLK}$

* Estes parâmetros procuram identificar o grau de liberdade da programação.

Parâmetros baseados no uso de recursos

Podemos agrupar estes parâmetros conforme reflitam uma medida de demanda ou uma relação entre demanda e disponibilidade. Medidas do primeiro grupo são calculadas separadamente para cada tipo de recurso e relacionam a demanda com elementos da rede e do caminho crítico. Tais medidas incluem: demanda média por atividade; demanda média por período;variança sobre a demanda média por período e o nível máximo requerido (pico de utilização de recurso). Uma medida de demanda total para o projeto é o "conteúdo total de trabalho, igual ao valor para CCR_k no término do projeto.

Uma medida usada para indicar a localização da predominância dos requisitos, no horizonte da programação é a chamada Momento Produto (indica o centro de massa para o DCR_k) . Outra medida, simples, porém, importante é o número de diferentes tipos de recursos.

Medidas de segundo grupo, também chamadas de Estatísticas sobre a Restrição de Recursos, tem forte influência no aumento da duração do projeto além do comprimento do caminho crítico, bastante comum na programação sob recursos limitados; daí serem normalmente relacionadas aos elementos do CPM-padrão.

Em geral, a magnitude da restrição do recurso é o excesso de demanda, gerado para períodos do DCR_k, nos quais o requisito é maior que a disponibilidade. Tais excessos podem ser expressos por várias formas: número de períodos em excesso; maior número de períodos consecutivos com excesso. Estas

medidas podem refletir a flexibilidade da programação sob recursos limitados, indicando situações, nas quais períodos de excesso são intermediados por períodos de ociosidade (demanda menor que a disponibilidade) ou situações em que o excesso ocorre em períodos sequenciais. Outra estatística sobre restrição de recurso mede a obstrução que determinada programação pode sofrer, devido à proporção entre excesso e conteúdo de trabalho (OFACT e LC-FAC). Duas outras medidas relacionando o excesso ajustado ao número de períodos e ao número de diferentes tipos de recursos são os chamados Fatores de Obstrução Ajustados: ADOPES e ADOPLS, calculados pelas programações mais cedo e mais tarde respectivamente. (Davis, 75).

Medidas análogas são definidas, considerando-se, ao invés do excesso, a ociosidade de recursos. Uma medida, análogo a usada por Fendley, é o fator de utilização de recurso, que é a razão entre o conteúdo total de trabalho e o conteúdo total disponível, considerado como o produto entre a duração dada pelo CPM-padrão e o limite de disponibilidade.

A seguir listamos os fatores de caracterização associados com o uso de recursos:

Tabela 4.3 - Parâmetros baseados no uso de recurso

NRES	Número de tipos de recursos
NAS_k	Número de atividades demandando recurso k $\sum_{P} \sum_{N_j} x$ onde $x = 0$ se $r_{ijk} = 0$, $x = 1$ se $r_{ijk} = 1$
$MAXNAS_k$	Número máximo de atividades em paralelo, demandando mesmo recurso k $\max_{maxCPL} \{ NAS_{kt} \}$ para todo k
$PCTR_k$	Percentagem das atividades demandando recurso k $\frac{NAS_k}{NNODE}$ para todo k
MIN%DEMAND	Percentagem mínima de demanda por recurso $\min_{R} \{ PCTR_k \}$
$\bar{X}$%DEMAND	Percentagem média de demanda por recurso $\frac{\sum_{R} PCTR_K}{NRES}$
MAX%DEMAND	Percentagem máxima de demanda por recurso $\max_{R} \{ PCTR_k \}$
$UTIL_k$ (Davis)	Utilização do recurso k (medida sobre o mais longo caminho) $\frac{\sum_{P} \sum_{N_i} r_{ijk} d_{ij}}{R_k \; MAXCPL}$ para todo $k \in R$
MINUTIL	Menor utilização dos recursos $\min_{R} \{ UTIL_K \}$
$\bar{X}$UTIL	Utilização média dos recursos $\frac{\sum_{R} UTIL_k}{NRES}$
MAXUTIL	Maior utilização dos recursos $\max_{R} \{ UTIL_k \}$
$DMND_k$	Demanda média de k por atividade $\frac{\sum_{P} \sum_{N_i} r_{ijk}}{NASK}$ para todo k
$PRES_k$ (Davis)	Proporção entre demanda máxima esperada e demanda disponível $\frac{DMND_k \; MAX \; NAS_k}{R_k}$ para todo k
$\bar{X}$DMND	Demanda média sobre os recurso $\frac{\sum_{R} DMND_k}{NRES}$
$CONSTR_k$	Indice de criticalidade do recurso k $\frac{DMND_k}{R_k}$ para todo $k \in R$
MINCON	Menor criticalidade sobre os recurso $\min_{R} \{ CONSTR_k \}$
$\bar{X}$CON	Criticalidade média dos recursos $\frac{\sum_{R} CONSTR_k}{NRES}$
MAXCON	Maior criticalidade sobre os recursos $\max_{R} \{ CONSTR_k \}$
VA-CON	Variança na criticalidade sobre os recursos $\frac{\sum_{R} (CONSTR_k - \bar{X}CON)^2}{NRES - 1}$

Tabela 4.3 - Continuação

TCON	Criticalidade de k sobre o tempo $\frac{\sum_{P} \sum_{N_i} r_{ijk}\, d_{ij}}{NAS_k\, R_k\, MAXCPL}$ para todo k
MINCON-TM	Menor criticalidade dos recursos sobre o tempo $\min_{R} \{ TCON_k \}$
$\bar{X}$CON-TM	Média de criticalidade dos recursos sobre o tempo $\frac{\sum_{R} TCON_k}{NRES}$
MAXCON-TM	Maior criticalidade dos recursos sobre o tempo $\max_{R} \{ TCON_k \}$
VA-CON-TM	Variança da criticalidade dos recursos sobre o tempo $\frac{\sum_{R} (TCON_k - \bar{X}CON\text{-}TM)^2}{NRES - 1}$
$ACON_k$	Criticalidade do recurso k, sobre todas as atividades $\frac{\sum_{R} r_{ijk}}{NNODE\ R_k}$ para todo $k \in R$
MINCON-ALL	Menor criticalidade de recurso, sobre todas as atividades $\min_{R} \{ ACON_k \}$
$\bar{X}$CON	Criticalidade média de recurso sobre todas as atividades $\frac{\sum_{R} ACON_k}{NRES}$
MAXCON-ALL	Maior criticalidade de recursos sobre todas as atividades $\max_{R} \{ ACON_k \}$
VA-CON-ALL	Variância na criticalidade de recurso sobre todas as atividades $\frac{\sum_{R} (ACON_k - \bar{X}CON\text{-}ALL)^2}{NRES-1}$
W_k	Conteúdo de trabalho do recurso k $\sum_{P} \sum_{N_i} r_{ijk}\, d_{ij}$
$OFACT_k$ (Davis, Pascoe)	Fator de obstrução do recurso k $\frac{\sum_{MAXCPL} \max [0, D_{kt} - R_k]}{W_k}$ D_{kt} = demanda de k no período t, para programaç mais cedo
TOTOFACT	Fator total de obstrução $\sum_{R} OFACT_k$
MINOFACT	Menor fator de obstrução $\min_{R} \{ OFACT_k \}$
MAXOFACT	Maior fator de obstrução $\max_{R} \{ OFACT_k \}$
$LC\text{-}FACT_k$ (Davis)	Fator de obstrução do recurso k para programação mais tarde
TOT-LC-FACT	Fator total de obstrução para programação mais tarde
MIN-LC-FACT	Menor fator de obstrução para programação mais tarde
MAX-LC-FACT	Maior fator de obstrução para programação mais tarde
$UFACT_k$	Fator de ociosidade do recurso k $\frac{\sum_{MAXCPL} \max [0, R_k - D_{kt}]}{W_k}$ D_{kt} = para programação mais cedo
TOTUFACT	Fator total de ociosidade $\sum_{R} UFACT_k$

Tabela 4.3 - Continuação

MINUFACT	Menor fator de ociosidade $\min_{R} \{ UFACT_k \}$
MAXUFACT	Maior fator de ociosidade $\max_{R} \{ UFACT_k \}$
$ADOPES_k$ (Davis)	Fator de obstrução ajustado $\Sigma \left[\frac{\sum_{MAXCPL} \max [0; D_{kt} - R_k]}{NRES \; MAXCPL} \right]$
$ADOPLS_k$ (Davis)	Fator de obstrução ajustado, calculado para programação mais tarde
$ADAOP_k$ (Davis)	Fator de obstrução combinado $\frac{ADOPES_k + ADOPLS_k}{2}$
$NOVER_k$	Número de períodos do horizonte de programação, que ocorre excesso de demanda de k sobre a disponibilidade do recurso k $\sum_{MAXCPL} \begin{cases} 1 \text{ se } demanda_{kt} > R_k \\ 2 \text{ se } demanda_{kt} \leq R_k \end{cases}$ para todo k
$\overline{X}OVER$	Média dos números de períodos com excesso de demanda $\frac{\Sigma_R \; NOVER}{NRES}$
MINOVER	Menor número de períodos com excesso de demanda $\min_{R} \{ NOVER_k \}$
MAXOVER	Maior número de períodos com excesso de demanda $\max_{R} \{ NOVER_k \}$
$NUNDER_k$	Número de períodos do horizonte de programação. com ociosidade ou plena utilização $\sum_{MAXCPL} \begin{cases} 1 \text{ se } R_k \geq demanda_{kt} \\ 2 \text{ se } R_k < demanda_{kt} \end{cases}$ para todo k
$\overline{X}UNDER$	Média do tempo de ociosidade ou plena capacidade dos recursos $\Sigma_R \frac{NUNDER_k}{NRES}$
MINUNDER	Menor tempo de ociosidade ou plena capacidade $\min_{R} \{ NUNDER_k \}$
MAXNUNDER	Maior tempo de ociosidade ou plena capacidade $\max_{R} \{ NUNDER_k \}$

Análise do uso de recursos: medidas para caracterização do problema de multirecursos sob abordagem de múltiplos-projetos
(Propostas por Kurtulus)

As medidas de caracterização do problema de programação sob recursos escassos, apresentadas por Kurtulus e Davis (82) e Kurtulus e Narula (85), são adequadas para a abordagem de multi-projetos.

Localização de picos de utilização: $ARLF_k$ e ALF_k

Estas propostas são feitas para identificação dos picos de utilização de recursos, através dos chamados Fatores de Carga de Recurso, pela ponderação da localização da ocorrência dos picos de demanda, sobre o horizonte de programação, considerados os Diagramas de Carga de Recurso (DCR_k) para as programações mais cedo dos projetos.

No Fator de Carga Média ($ARLF_k$) as ponderações feitas para ocorrência de pico são iguais a um, porém negativa se é na primeira metade da duração do caminho crítico e positiva, caso contrário, sobre o DCR_k para cada projeto.

O Fator de Carga Média para o problema é calculado pela média sobre todos os projetos.

Definindo

$$z_{ijt} = \begin{cases} 1 \text{ se } t \geq CP_i/2 \\ -1 \text{ caso contrário} \end{cases}$$

$$x_{ijt} = \begin{cases} 1 \text{ se a atividade j do projeto i é ativa em t} \\ 0 \text{ caso contrário} \end{cases}$$

onde CP_i é a duração do projeto i pelo CPM - padrão tem-se que o FATOR DE CARGA MÉDIA para o projeto i, $ARLF_i$ é igual a:

$$\frac{1}{CP_i} \sum_{CP_i} \frac{1}{NRES_i} \sum_{NRES_i} \sum_{N_i} z_{ijt}\, x_{ijt}\, r_{ijt}$$

O valor do FATOR DE CARGA MÉDIA para o problema é a média sobre os valores dos fatores dos projetos

$$ARLF = \frac{1}{P} \sum_{P} ARLF_i \qquad \text{para cada recurso k}$$

Se os recursos puderem ser expressos por uma unidade agregativa, pode-se usar esta medida para localizar picos de trabalho do conjunto de projetos.

A segunda medida proposta, na tentativa de se localizar os picos de demanda, é calculada por recurso k, sobre o DCR_k construído sobre a demanda de todas as atividades desenvolvidas no problema, programadas para suas datas de início mais cedo.

As ponderações associadas aos picos de utilização são diferenciadas para quartis do horizonte da programação. Os períodos de pico no primeiro e no quarto quartil tomam peso igual a dois, sendo negativo para o primeiro e positivo para o quarto. Aqueles localizados no segundo e no terceiro, recebem peso 1, com sinal negativo e positivo, respectivamente.

Definindo:

T = MAXCPL

Conteúdo de trabalho para o período t, demandando o recurso k, como:

$$W_{kt} = \sum_{P} \sum_{N_i} r_{ijkt} \, x_{ijt} \qquad e$$

$$W'_k = \max_{T} \{ W_{kt} \} \; e$$

$t^{*\S}$ o período correspondente à ocorrência de W'_k

As ponderações a $t^{*\S}$ sendo feitas por:

$$z_{\S} = \begin{cases} -2 \text{ se } 0 < t^{*\S}/T \leq 0.25 \\ -1 \text{ se } 0.25 < t^{*\S}/T \leq 0.5 \\ 1 \text{ se } 0.5 < t^{*\S}/T \leq 0.75 \\ 2 \text{ se } 0.75 < t^{*\S}/T \leq 1.0 \end{cases}$$

Então o Fator de Carga Máxima (MLF) para o recurso k é dado por:

$$MLF_k = \frac{1}{n} \sum_{\S = 1}^{n} W'_k \, z_{\S}$$

para o caso de n períodos de picos de demanda pelo recurso k.

Propõe o FATOR DE CARGA MÉDIA (ALF) dividindo o Fator de Carga Máxima pelo número total de atividades do problema.

$$ALF_k = \frac{1}{NNODES} \; MLF_k$$

Para o problema de múltiplos recursos, consideram-se os vetores

$$\vec{ARLF} = [\ ARLF_1, \ldots, ARLF_k[$$

$$\vec{ALF} = [\ AUF_1, \ldots, AUF_k]$$

Criticalidade dos recursos

As medidas apresentadas, análogas às de Fendley (68), indicam a criticalidade de cada recurso.

São criados segmentos S_L, através da ordenação das durações dos caminhos críticos dos projetos, da seguinte forma:

$$S_1 = CP_1;\ S_2 = CP_2 - CP_1;\ \ldots;\ S_M = MAXCPL - CP_{M-1}$$

Fazendo $a = CP_{L-1} + 1$ e $b = CP_L$, define conteúdo de trabalho para cada segmento L

$$W_{S_1,k} = \sum_{t=a}^{b} W_{kt} \quad e$$

O Fator Médio de Utilização (AUF)

$$AUF = \frac{1}{M} \sum_{L=1} (W_{S_1,k}) / (R_k \cdot S_L)$$

Para o caso de múltiplos recursos

$$\vec{AUF} = [\ AUF_1, \ldots, AUF_k]$$

Medidas usadas por COOPER (1976)

Em seu experimento comparando sua proposição de procedimento heurístico, chamado de método amostral e o método paralelo, usa os seguintes parâmetros:

$OS = \frac{T}{U}$ - ordem de densidade - expressando a relação entre número de relações de precedência no projeto (T) e número de relações de precedência possíveis

$[\ U = \frac{N\ (N-1)}{2}\ ]$

[rede de atividades nos nós]

$$P - Density\text{-}F = \left(\sum_{i \varepsilon a} d_i \right) / \left(\sum_{i \varepsilon a} (d_i + ffree_i \right)$$

$$\text{Fator de recurso } rf = \frac{\sum_{kj} \sum_{j}}{K}\ , \text{ tal que } r_{jk} \neq 0$$

$$\text{Potência de recurso k} \quad rs_k = W_k \ / \ \left(\sum_t R_{kt} \ / \ NAS \right)$$

Medidas usadas por Thesen (1976)

U (Thesen) Utilização real de recurso

$$U = \left(\frac{\sum \sum d_j \ r_{jk}}{R_k} \right) \ / \ K,\ CP$$

I (Thesen) $0 \leq I \leq 1$ Conteúdo de informação da rede (grau de restrição que a rede impões ao número de sequências viáveis para o projeto)

Estimativa de I dada por $\hat{I}$

$$\hat{I} = \frac{L_N}{L_D - L_N}$$

L_N = número de atividades não redundantes (existe uma ligação não redundante entre dois nós, se nenhuma cadeia ocorre entre os dois nós)

L_D = conjunto de arcos não disjuntivos (um arco não disjuntivo é um arco imaginário ligando duas atividades, sem relação de precedência).

Função Penalidade

Como exposto no Capítulo II, o problema de multiprojetos pode ser tratado com diferenciação entre os projetos.

Kurtulus & Narula (85) apresentam funções penalidades, de acordo com: demanda por recursos e duração do caminho crítico.

A penalidade associada a demanda por recursos, indica que atrasos podem ocorrer em projetos que requerem maior quantidade total de recursos. Assim, penalisando os projetos pela demanda total dos recursos, estar-se-ia priorizando o projeto mais complexo, devido à reunião de vários recursos.

A função penalidade seria:

$$p_i = [\ TWK_i / (\min_i TWK_i)\ ] \times 100$$

onde

$$TWK_i = \sum_{j=1}^{N_i} \sum_{k=1}^{K} d_{ij}\ r_{ijk}$$

Um raciocínio inverso é usado, no caso de ter-se a necessidade de geração de fundos para a empresa gerenciadora dos projetos: é priorizar o término do projeto requerendo menos recurso. Se terá retorno mais rapidamente, por um esforço menor. Assim, uma função penalidade priorizante do projeto demandando menor quantia por recursos, é definida:

$$p_i = [\ (\min_i TWK_i)/TWK_i\ ] \times 100$$

A priorização por duração do caminho crítico, dada àquele mais longo, é baseada no crédito que, assim, estar-se-

ia evitando maiores atrasos:

$$p_i = [\ CP_i\ /\ (\min_i CP_i)\ [\ x\ 100$$

O inverso pode ser considerado: priorizar o projeto de menor duração para liberação de recursos. Assim,

$$p_i = [\ (\min CP_i)\ /\ CP_i\ ]\ x\ 100$$

Critérios

$$\text{Atraso total} = \sum_i (CR_i - CP_i)$$

$$\text{Atraso ponderado} = \sum_i \sum r_{ik}\ (CR_i - CP_i)$$

Eficácia na utilização (total de tempo de ociosidade dos recurso) = $\sum_t [\max\ (0;\ R_k - \sum r_{ijk})]$ para $R_k > \sum_t r_{ijk}$

$$OVDUR = \frac{CR}{CP}$$

$$A\%C^* = \frac{CR}{CEL}$$ onde CEL é a duração da programação ótima fornecida pelo algoritmo de enumeração limitada de Davis.

MAXCPL = CR - MAXCPL (abordagem de projeto único)

$$A\%CP = \frac{CR-CP}{CP}$$ - aumento percentual relativo à duração do caminho crítico (abordagem de projeto único)

CAPÍTULO 5

Caracterização de Procedimentos Heurísticos em Programação de Projetos

A natureza das heurísticas faz com que sua performance esteja relacionada a características do problema e a estratégia de aplicação.

A investigação sobre o assunto tem sido pautada em experimentos, com testes sobre problemas, a maioria composto por projetos simulados. O primeiro trabalho sistemático de comparação da performance das regras, relativa a um resultado ótimo é apresentada por Davis & Patterson (75). Neste, os autores fazem uma revisão dos experimentos desenvolvidos anteriormente, apresentando a seguinte revisão: "... em Kelley, em um dos primeiros trabalhos realizados sobre o assunto, tem uma excelente discussão do uso de diferentes heurísticas para programação de projetos únicos. Ele conclui que existe pouca base, a priori, para se fazer uma escolha sobre diferentes procedimentos e que o melhor teste sobre a qualidade de uma particular abordagem era se produzisse prgoramações "razoáveis" para projetos reais, Outros pesquisadores, tem frequentemente concordado com a conclusão geral de Kelley, com base em evidências limitadas Verhines, por exemplo, advoga o uso irrestrito para a regra LFT, aparentemente com base na habilidade em produzir durações mais curtas, que outras regras testadas para poucos problemas selecionados".(2)

---

(2) Se refere a análise sobre testes de aplicações a heurísticas específicas, remetendo aos textos:

Obrien, J.J. "Resources scheduling", Scheduling Handbook , Capítulo 6, Prentice Hall, 1969.

Apresenta uma tabela, usada na montagem da Tabela 5.6, deste trabalho, listando testes comparando performance de diferentes heurística, em ordem cronológica de realização, aplicados a problemas de projeto único e de múltiplos projetos, examinados para verificação do efeito na duração do projeto e atraso dos projetos, respectivamente, confirmando a LTF como a melhor, na maioria dos casos [LTF ≃ LST].

Cita resultados do trabalho desempenhado por Mueller-Merhback[3], comparando soluções heurísticas com soluções ótimas, sobre teste em 42 redes de projeto único sob condições de job-shop, no qual as heurísticas dão resultados ótimos em 40% dos casos, e soluções com duração 5% maior que a ótima, em 28% dos problemas. Não especifica as regras usadas nesta experiência.

A maioria dos experimentos, para teste de heurísticas, são realizados sobre um conjunto de projetos simulados, onde determinadas características são controladas e sob varia

---

Pascoe (65) "An experimental comparison of heuristic methods for allocating resources", Tese de Doutorado não publicada, Departament of Engineering Cambridge University, 1965.

Walton, H. "Resource allocation-workshop application" in Project Planning by Network Analysis, North-Holland Press, 1969.

Woodgate, H.S. "The planning network as a basis for resource allocation, cost planning and project profitability assesment", Proceedings, Fall Joint Computer Conference, 1967.

(3) Mueller-Merhback, H. "Ein verfahren zur planung des optimalen betrieksmittelainsatzes bei der terminierung von grossprojekten", AWF - Mitteilung Handbook, Capítulo 6, Prentice Hall, 1967.

ção de outras, estabelece-se a análise da variabilidade do resultado obtido, com a aplicação da heurística para geração de uma programação viável.

As experiências diferem no enfoque dado a abordagem para a programação de vários projetos: abordagem de projeto único, quando todos os projetos são ligados numa rede única e programados como um único projeto e abordagem de múltiplos projetos, quando os projetos são conectados pela alocação dos recursos em comum, com disponibilidade limitada.

Neste capítulo resumimos experimentos encontrados na literatura, sobre estudos comparativos da performance de diferentes regras de priorização.

Outras considerações são tomadas para referenciar as análises sobre as heurísticasm tais como: tipo de problema , segundo número de tipos de recursos, quantidade disponível , quantidade requerida por atividade [Holloway (1979)], método de sequenciamento (paralelo/seriado), critério-objetivo, regras analisadas, metodologia de análise, características do(s) problemas-teste.

Experimento de Davies (73)

(n/n/n)

Análise de fatores significantes para alocação de recursos limitados.

Projeto único; único recurso por atividade.

Caso preemptivo e não preemptivo (Método Paralelo).

Método Paralelo. Desempate com MOF.

Controle sobre duração obtida relativa a duração do caminho crítico (OVDUR = CR/CP).

Características dos problemas

648 redes; $25 \leq Ni \leq 68$; $0.1 \leq CC$ (Davies) ≤ 0.5

$0 \leq$ NDUMMY ≤ 20

$0.5 \leq$ DENSITY-F ≤ 0.9

$0.1 \leq$ PRES ≤ 0.5

Experimento

Considerou o problema dividido em duas partes: a primeira foi analisada sob a condição de poder interromper as atividades, quando outra mais urgente estava para ser programada. O restante da atividade no próximo período era a primeira colocada. A segunda: caso preemptivo.

Conclusão

Conclui que o fator de maior significância sobre o resultado da programação foi o parâmetro PRES, listando por ordem de importância os demais: DEN; CC; NDUMMY; Regra. Embora as regras não tenham re-

sultado efetivo para sua análise, avalia suas performances , posto que representam o único fator, passível de controle. Ordena as de melhor performance: LTF, LST, maior custo por unidade de tempo, MOF, LFT.

Apresenta um modelo de previsão de OVDUR, priorizando as atividades por maior custo por unidade de tempo, relacionado os fatores NDUMMY, CC (Davies), DEN, PRES, com as seguintes ponderações: 0.02; 2.05; 3.46; -7.29.

As outras regras analisadas foram: EFT, EST, GRD , GRU, FREE, SIO, sendo GRD e GRU, ordenadas em ordem crescente e decrescente, e RAN.

Experiência de Patterson (73)

(n/n/n)

Um problema de multiprojetos. Multi-recursos. Recursos restritos. Considera as regras LTF, GTRD, GRRD, SIO, GRU, RAN, GRES.

Critérios: atraso total, atraso ponderado, tempo de ociosidade.

Abordagem de único projeto.

Características do problema

Geração: variação das datas de início de cada projeto, de forma aleatória sobre o horizonte de 180 dias. Ordenação dos projetos por datas de início.

Metodologia de análise

Define 3 tratamentos e 20 destes tratamentos são combinados:

A - LTF, GTRD, GRRD, SIO, GRU

B - LTF/RAN, GTRD/RAN, GRRD/RAN, SIO/RAN, GRU/RAN

C - LTF/RES, GTRD/RES, GRRD/RES, SIO/RES, GRU/RES

D - Combinação dos três tratamentos

Amostragem: 43 projetos ligados por uma rede e com datas de início defasadas. Análise de variança.

$0.75 \leq CC \text{ (Pascoe)} \leq 0.99$

$\overline{X}CPL = 90.11$

$0.26 \leq UTIL_k \leq 0.81$

$1.03 \leq \overline{X}\%DEM \leq 2.28$

considera no máximo 13 tipos de recurso por problema

Conclusões

É econômicamente prático usar todas as regras, em paralelo, e então pelos resultados obtidos, escolhero melhor.

As regras que produzem menor atraso total, o fazem causando maior ineficiência no uso dos recursos.

As regras que atendem à eficiência de recursos, causam grandes atrasos aos términos do projeto.

Os procedimentos heruísticos devem ser considerados devido a sua utilidade e não por sua otimalidade.

Experimento de Davis (75)

(n/n/n)

Comparação do método de enumeração limitado com e sem critério de seleção.

Regras: LFT; GRD; GRU; SIO; MJP; RAN; RSM; LTF.

Método Paralelo.

O experimento de Davis foi um teste sobre 83 problemas de múltiplos projetos, montados sobre 57 redes hipotéticas, para comparação de seu algoritmo de enumeração limitada aplicado exaustivamente até a solução ótima e aplicado usando como critério de seleção, para escolha de candidatas potenciais, as oito heurísticas: LTF, RSM, LFT, GRD, GRU, MJP, SIO e RAN, as regras LTF, SIO, RAN e GRD necessitam requisito computacional k_t, semelhantes para sua implementação; a heurística RSM requer igual espaço de memória que as anteriores, porém mais tempo computacional, para processamento das comparações efetuadas a cada período; as regras MJP e GRU ocupam duas vezes mais espaço de memória e aproximadamente cinco vezes mais tempo de CPU, devido a necessidade de processamento da programação 0-1, em cada período.

Características do problema

No máximo 3 tipos de recursos por atividade e por projeto. Disponibilidade do Recurso e requisito por atividade constantes.

$1.0 \leq C \leq 3.0$

$0.58 \leq UTIL_k \leq 1.50$

$1 \leq d_{ij} \leq 9$

Resultados

Análise do aumento percentual obtido sobre a solução ótima definida pela busca exaustiva pelo método de enumeração limitada (%AD)

Heurística Usada	% AD			N. de vezes em que a duração obtida foi				
	$\bar{X}$	S_x	MAX	Ótima	A Menor	Única Menor	A Maior	Única Maior
LTF	5.6	6.1	23.68	24	50	15	2	1
LFT	6.7	5.7	29.62	17	38	5	4	0
RSM	6.8	5.1	20.0	12	28	5	2	2
RAN	11.4	7.9	33.31	4	13	5	14	8
GRU	13.1	7.0	36.36	2	6	0	14	1
GRD	13.1	9.5	40.74	11	15	4	20	11
SIO	15.3	8.3	36.36	16	3	1	33	18
MJP	16.0	8.1	36.36	2	4	1	30	14

Conclusões

Nenhuma das regras testadas foi consistemente mais efetiva, para todos os problemas. Destaca a LTF, prioridade às atividades pela menor folga, por ter produzido mais vezes a duração ótima e o menor aumento percentual médio, relacionado à duração ótima.

A regra RSM apresenta menor desvio padrão, associado com a obtenção da duração máxima obtida, frente ao de todas as regras.

Destaca, ainda, a regra LFT por ter resultados in-

termediários às duas acima.

Comparando seus resultados com o de experiências efetuadas para projetos únicos, destaca a diferenciação que pode correr entre resultados na programação de um único projeto, e na de mais de um projeto, sob a abordagem de único projeto. Destaca as regras GRU e SIO, como efetivas na programação de mais de um projeto, devido ao número de vezes em que o ótimo foi obtido.

Coloca a questão sobre se estes resultados podem ser estendidos para redes maiores, como afirmado por Pascoe. Deixa a questão em aberto.

Conclui, afirmando a relação entre características do problema, sobre a estrutura da rede e a criticalidade dos recursos, e o efeito na performance das heurísticas.

Experimento de Cooper (76)

(n/n/n)

Analisa as regras heurísticas usadas na comparação entre o método amostral, que propõe, e o método paralelo usual.

Características do programa

Usa dois grupos de projetos - Para o método paralelo as características das redes são:

OS = { 0.1; 0.25; 0.5; 0.75 }

P-DENSITY-F = { 0.5; 0.75 }

K = 5

r_f = { 0.5; 1.0 }

$r_{jk} \leq [1,10]$

Para o método amostras as redes obedecem a:

OS = { 0.25; 0.5 }

P-DENSITY-F = { 0.5; 0.75 }

r_f = { 0.5; 1.0 } e a análise é sobre as regras.

FCFS; LST; EST; LFT; SIO; LONG; MSAI; MST; RAN ; MSA; MINA; MONA; RANK.

O método paralelo é aplicado as regras acima e mais: MST; PROD; RED; CUMRED; PRANK; RRANK; MSAP ; MISA; FSFS; LTFI; LEVEL; LFTP.

Faz comparações usando regressão múltipla e análise de variança sobre os problemas testes.

Critério: duração do projeto.

Conclusões

Os resultados do método amostral, quando considera da mesma heurística, são melhores que os do método paralelo , produzindo soluções 7% melhores,embora demande mais tempo computacional.

A escolha da regra de prioridade é menos crítica quando as restrições de recurso são altas. O método amostral apresenta grande variabilidade de performance, conforme a heurística usada. A melhor regra para o método amostral, indicada pelo experimento, foi a combinação das regras LONG, MSA, MONA, MST, RANK, pelo produto dos seus valores (produziu a

sequência amostral de menor duração).

As melhores regras para o método paralelo foram: FSFS, CUMRED, RANK, MSA, LTFI.

Experimento de Patterson (76)

Análise entre relação das características do problema e função objetivo sobre a performance das heurísticas:LTF, GRD, GRRD, RSM, SIO, GRU, LFT, MJP, RAN. Funções objetivo analisadas: Atraso total, Atraso ponderado; Atraso no mais longo caminho crítico (aumento porcentual sobre a duração do caminho crítico, sob a abordagem de único projeto).

Metodologia

Regressão múltipla, correlação múltipla e regressão step-wise.

Amostragem

Usou dois grupos de problemas de multi-projetos:

G1: 60 problemas sob abordagem de multi-projetos, montados com a combinação de 6 a 10 projetos,cada um dos quais contendo entre 20 a 40 atividades.

G2: 83 problemas do Experimento de Davis. Os projetos ligados numa única rede. (Abordagem de um único projeto).

Considerou 13 categorias diferentes de recursos por problema, onde cada atividade demanda quantia fixa de no máximo 13 tipos.

Considera estes projetos, com características dos encontrados na prática, embora tenham sido todos simulados

respeitando, em média, os seguintes valores:

Sumário Estatístico para o Experimento para a abordagem de multi-projetos [60 problemas testes de multiprojetos]*

Parâmetro	Média	σ^2	Parâmetro	Média	σ^2	Parâmetro	Média	σ^2
Σ DUR	1,395.43	621.23	PDENSITY-T	0.47	0.07	VA-CON	0.08	0.02
$\bar{X}$DUR	7.49	0.40	PCTFREESLK	0.09	0.02	MAXCON-TM	0.04	0.01
NSLACK	70.07	31.19	PDENSITY-F	0.82	0.05	MAXOFACT	11.07	8.23
PCTSLACK	0.38	0.03	$\bar{X}$%DEMAND	0.18	0.02	MAXUFACT	0.58	0.16
TOTSLACK-R	8.62	4.39	MINUTIL	0.11	0.06	MAXCON-ALL	0.24	0.13

* Extraída de Patterson (76), pag.110.

Conclusões

A conclusão principal do trabalho de Patterson é a demonstração da relação estatística entre o objetivo da programação e características do problema, para análise do desempenho das heurísticas na geração das soluções, resultado este sugerido por Davis (75).

Apresenta resultados sobre análise de regressão para indicação da regra mais efetiva, sob específica configuração do problema. Ainda, assim, aponta situações em que a escolha da regra não é importante, como por exemplo quando o fator de obstrução inicial (programação mais cedo) é baixo, em que todas produzem soluções próximas.

Através de correlação múltipla a regressão step-wi-

se entre os parâmetros característicos do problema e regras , chama atenção para o fato que, em algumas situações, há necessidade de desvendar mais o problema para orientar sobre o uso das regras, posto que grande número delas, são sujeitas igualmente a estes parâmetros, quais sejam: obstrução inicial alta; grande quantidade de folga livre e número alto de atividades com folga livre e algumas medidas de criticalidade dos recursos:

ANÁLISE ENVIESADA*

Parâmetros significantes para mais que quatro heurísticas			
Abordagem Multi-Projetos			Único-Projeto
A%MAXCPL	Atraso Ponderado	Atraso Total	A%CP
MAXUTIL (8)	VA-DUR [(+)5]	NSLK [(+)6]	MINUTIL [(+8)]
$\overline{X}$CON-TM (5)	ΣFREE [(+)7]	$\overline{X}$CON [(+)5]	MAXOFACT [(+7)]
	$\overline{X}$FREE [(-)7]	MAXCON [(+)7]	VA-CON-ALL [(+)5]
	$\overline{X}$%DEM [(+)6]	$\overline{X}$OVER [(+)7]	
	MAXCON [(+)6]		
	VA-CON [(-)7]		
	$\overline{X}$-CON-ALL [(-)5]		
	TOTOFACT [(+)5)]		
	MINOFACT [(-)6]		
	TOTUFACT [(+)7]		
	$\overline{X}$OVER [(+)7]		

* Baseado nos resultados de Patterson (76).

Embora a significância de parâmetros do problema relacionados ao critério desejado, algumas regras tem desempenho garantidamente efetivo independente da abordagem utilizadas, se com a de multiprojetos ou de projeto único: as regras LTF e LFT são bastante efetivas para ambos os tratamentos, no caso de minimização do tempo de ocorrência (makespan), particularmente quando a variabilidade na duração das atividades e a média de folga livre por atividade são altas.

SENSIBILIDADE DAS REGRAS*

Múltiprojetos - atraso no mais longo caminho crítico (makespan)

Parâmetro	Alto	Baixo	Regra Sujeita	Rank	
				Média	σ^2
NSLACK	(-)		MJP	8º	8ª
		(+)	LTF	2ª	2ª
XSLACK-R	(-)		GRD	5ª	5ª
ΣFREESLK		(+)	RAN	4ª	3ª
$\overline{X}$FREESLK	(-)		RAN	4ª	3ª
$\overline{X}$CON	(-)		GRD	5ª	5ª
MINCON-ALL	(-)		RAN	4ª	3º
VA-CON-ALL	(-)		GRD	5ª	5ª
MAXOFACT	(-)		GRD	5ª	5ª
TOTUFACT	(-)		LTF	2ª	2ª
MAXUFACT	(-)		MJP	8ª	8ª
MAXOVER		(+)	SIO	6ª	7ª
MAXUNDER	(-)		SIO	6ª	7ª
		(+)	MJP	8ª	8ª

* Baseado nos resultados de Patterson (76), pag.114 e 115.

Critério: atraso no mais longo caminho crítico

Melhor regra: LFT

2ª melhor: LTF

3ª melhor: GRRD

SENSIBILIDADE DAS REGRAS*

Projeto único - Aumento % na Duração do Caminho Crítico

Parâmetro	Alto	Baixo	Regra Sujeita	Rank	
				Média	σ^2
VA-DUR	(-)		RAN	4ª	7ª
COMPLEXITY	(-)		SIO	7ª	1ª
$\overline{X}$CPL	(-)		LTF	1ª	6ª
NSLACK		(+)	LFT	2ª	3ª
PCTFREESLK	(-)		LFT	2ª	3ª
MINOFACT		(+)	GRD	6ª	8ª
MINUFACT		(+)	RSM	3ª	4ª
MAXUFACT	(-)		MJP	8ª	5ª
MINOVER		(+)	MJP	8ª	5ª

Regra melhor: LTF

2ª melhor: LFT

3ª melhor: RSM

* Baseado em resultados de Patterson (76), pag.118.

Sobre o efeito no Atraso total e Atraso ponderado, verifica-se no geral, que a regra SIO tem melhor desempenho . Importante observar, porém, que, quando a variança na duração das atividades é alta, qualquer outra regra dá melhores resul

tados que ela. Sugere, assim, que a heurística SIO seja usada com o objetivo de minimizar atraso total, desde que a variança na duração das atividades seja pequena, porém não nula.Neste caso, sugere a MJP.

A possibilidade de trabalhar com maior número de atividades,pode resultar em que as folgas de recursos, sujeitas a uma atividade longa, sejam usados para execução de outras de duração menor. Assim fazendo, combina as duas melhores regras apontadas para o caso.

SENSIBILIDADE DAS REGRAS*

(multiprojetos - atraso total)				Rank sobre a média	
Parâmetro	Alta	Baixa	Regra Sujeita	Média	σ^2
VA-DUR		(+)	SIO	1ª	1ª
PCTFREE	(-)		MJP	2ª	2ª
MAXUTIL		(+)	LFT	6ª	7ª
	(-)		GRD	5ª	4ª
MINCON		(+)	GRU	3ª	5ª
VA-CON-ALL		(+)	LTF	8ª	8ª
MINUFACT	(-)		GRU	3ª	5ª
		(+)	GRD	5ª	4ª
MINOVER	(-)		LFT	6ª	7ª
MAXOVER	(-)		LTF	8ª	8ª

Melhor regra: SIO
2ª melhor: MJP
3ª melhor: GRU

* Baseado em resultados de Patterson (76), pag.112 e 113.

SENSIBILIDADE DAS REGRAS*					
Multiprojetos - atraso ponderado				Rank sobre a média	
Parâmetros	Alto	Baixo	Regra Sujeita	Média	σ^2
NDUMMY	(-)		SIO	1ª	3ª
COMPLEXITY	(-)		LFT	6ª	6ª
VA-CPL	(-)		GRU	5ª	5ª
NFREESLK		(+)	GRU	5ª	5ª
		(+)	GRD	3ª	4ª
NFREE		(+)	GRU	5ª	5ª
MAXUTIL	(-)		GRD	3ª	4ª
VA-CON-TM	(-)		LFT	6ª	6ª
VA-CON-ALL		(+)	GRD	3ª	4ª
MAXUFACT	(-)		GRD	3ª	4ª
MINOVER		(+)	GRU	3ª	5ª
	(-)		LFT	6ª	6ª
MAXOVER	(-)		GRD	3ª	4ª
		(+)	MJP	2ª	2ª

Melhor regra: SIO
2ª melhor: MJP
3ª melhor: GRD

* Baseado nos resultados de Patterson (76), pag.112 e 113.

Para afirmar com precisão sob quais valores do parâmetro, as regras por ele influenciadas seria mais efetiva, propõe uma metodologia que viria, mais tarde, a ser usada no experimento de Kurtulus & Davis [Procedimento de lay-out de duas entradas - Hollander e Wolfe, 1973].

Mesmo com esta restrição, seu modelo de regressão oferece explicação, em geral, para não menos que 82% da variabilidade dos resultados obtidos usando as heurísticas. Apresenta resultados estatísticos sobre a eficácia do seu procedimento usando análise de regressão (uso da heurística indicada = procedimento da heurística projetada). Para o caso de minimizar atraso total e ponderado é significativo, devido a evitar-se o "esforço" em se experimentar todas as regras e escolher a melhor solução obtida. Em média, as soluções propostas pela heurística projetada são melhores.

Já para a previsão do atraso do mais longo caminho crítico, a eficácia do seu procedimento não é significativa.

Questiona, porém, a validade da sua proposta, devido a que o aumento crescente na proposição de heurísticas novas, faz com que a vantagem relativa de seu modelo diminua,já que é restrito a estas testadas.

TABELA 3.1 - Resultados da regressão múltipla sobre atraso total*

[60 problemas simulados de programação multi-projetos]

HEURÍSTICAS \ CARACTERIZAÇÃO DO PROBLEMA	Baseado no tempo e na rede antes da análise do caminho crítico					Baseado no tempo e na rede após análise do caminho crítico										
	NDUMMY	X̄DUR	VA-DUR	X̄DENSITY	COMPLEXITY	X̄CPL	VA-CPL	NSLACK	PCTLACK	X̄SLACK	TOTSLACK-R	X̄SLACK-R	ΣFREESLK	NFREESLK	PCTREESLK	X̄FREESLK
LFT		+	+			-	+	+			-	+	+			-
GRD						-							+			
GRRD								+								
SIO			-			-		+		-		+		-		
GRU	+	+	+			-	-	+			-	+	+			-
LFT		+	+		-	-				-		+	+			-
MJP						-	-	+							+	
RAN			+					+	-							

HEURÍSTICAS	Baseado no uso de recursos																												
	MINZDEMAND	X̄ZDEMAND	MAXZDEMAND	MINUTIL	X̄UTIL	MAXUTIL	XDMND	MINCON	XCON	MAXCON	VA-CON	X̄CON-TM	MAXCON-TM	VA-CON-TM	MINCON-ALL	X̄CON-ALL	VA-CON-ALL	TOTOFACT	MINOFACT	MAXOFACT	TOTUFACT	MINUFACT	MAXUFACT	X̄OVER	MINOVER	MAXOVER	X̄UNDER	MINUNDER	MAXUNDER
LFT	+	+			+			-	+	+	-	-	+		-	-	+		-					+		-			
GRD					+	-				+						-						+		+					+
GRRD		+								+	-										+			+					
SIO							+	-	+	+				-				+		-				+			+		
GRU	+	+	-		+			+	+	+	-	-				-		+	-	-		-		+					+
LFT	+	+	-			+			+	+	-	-	+	-	-	-		+	-	-				+	-		+		
MJP	+						+		+				+	-				+		-	-		+			+	+		
RAN										+											+			+					

HEURÍSTICAS	Estatísticas e Resultados da Regressão			
	Estatísticas		Resultados da Regressão	
	Média do atraso total	Desvio padrão	Estimativa do erro padrão	R^2
LFT	390.82	232.05	64.08	0.96
GRD	359.60	211.81	67.64	0.91
GRRD	358.93	204.89	61.46	0.92
SIO	326.73	200.52	66.77	0.92
GRU	356.30	213.22	45.16	0.97
LFT	372.95	224.28	55.76	0.96
MJP	337.70	203.58	74.21	0.90
RAN	373.60	222.53	76.32	0.89

(*) Nível de significância dos coeficientes da regressão até 95%.

(+) Coeficiente de regressão positivo.

(-) Coeficiente de regressão negativo.

(**) Patterson [76] - páginas 112 e 113.

TABELA 5.2 - Resultados da regressão múltipla para atraso ponderado*
[60 problemas simulados de programação multi-projetos]

CARACTERIZAÇÃO DO PROBLEMA	Baseado no tempo e na rede antes de análise do caminho crítico					Baseado no tempo e na rede após análise do caminho crítico										
HEURÍSTICAS	NDUMMY	X̄DUR	VA-DUR	X̄DENSITY	COMPLEXITY	X̄CPL	VA-CPL	NSLACK	PCTSLACK	X̄SLACK	TOTSLACK-R	X̄SLACK-R	ΣFREESLK	NFREESLK	PCTFREESLK	X̄FREESLK
LFT			+							-		+	+			-
GRD		+	+			-	+				-	+	+			-
GRRD		+	+			-				-		+	+			-
SIO	-		+							-		+	+			-
GRU			+			-	-					+	+	+		-
LFT		+	+		-	-				-		+	+			-
MJP																
RAM													+			-

CARACTERIZAÇÃO DO PROBLEMA	Baseado no uso de recursos																												
HEURÍSTICAS	MIN%DEMAND	X%DEMAND	MAX%DEMAND	MINUTIL	X̄UTIL	MAXUTIL	X̄DMND	MINCON	X̄CON	MAXCON	VA-CON	X̄CON-TM	MAXCON-TM	VA-CON-TM	MINCON-ALL	X̄CON-ALL**	VA-CON-ALL	TOTOFACT	MINOTACT	MAXOFACT	TOTUFACT	MINUFACT	MAXUFACT	X̄OVER	MINOVER	MAXOVER	X̄UNDER	MINUNDER	MAXUNDER
LFT									+	+	-								-		+			+					
GRD	+	+	-	+	+	-		+	+	+	-	-	+		-	-	+	+	-	-	+	+	-	+		-			+
GRRD	+	+	-			+			+	+	-	-	+		-	-		+	-	-		-		+					
SIO		+						+	+	+	-					-			-		+			+					
GRU	+	+	-		+				+	+	-					-		+	-	-	+	-		+	+				+
LFT	+	+	-	+		+				+	-	-	+	-	-	-		+	-	-	+			+	-				+
MJP	+					+												+	-	-	+					+			
RAM		+								+	-										+			+					

CARACTERIZAÇÃO DO PROBLEMA	Estatísticas e Resultados da Regressão			
	Estatísticas		Resultados da Regressão	
HEURÍSTICAS	Média do atraso Ponderado	Desvio padrão	Estimativa do erro padrão	R^2
LFT	317.962	206.664	71.183	0.90
GRD	287.495	183.392	36.005	0.98
GRRD	287.970	177.508	35.611	0.97
SIO	271.129	181.779	62.720	0.91
GRU	289.576	183.628	40.940	0.97
LFT	302.709	194.513	42.184	0.97
MJP	275.561	178.562	81.843	0.82
RAM	304.636	194.579	72.888	0.88

(*) Nível de significância dos coeficientes da regressão até 95%.

(+) Coeficiente de regressão positivo.

(-) Coeficiente de regressão negativo.

(**) Patterson [76] - páginas 112 e 113.

TABELA 5.3 - Resultados da regressão múltipla sobre aumento na duração do caminho crítico mais longo* [60 problemas simulados de programação de projetos]

CARACTERIZAÇÃO DO PROBLEMA		Baseado no tempo e na rede antes da análise do caminho crítico					Baseado no tempo e na rede após análise do caminho crítico									Baseado no uso de recursos																				Estatísticas e Resultados da regressão			
																																				Estatísticas		Resultados da regressão	
		NDUMMY	$\bar{X}$DUR	VA-DUR	$\bar{X}$DENSITY	COMPLEXITY	$\bar{X}$CPL	VA-CPL	NSLACK	PCTSLACK	$\bar{X}$SLACK	TOTSLACK-R	$\bar{X}$SLACH-R	ΣFREESLK	NFREESLK	PCTREESLK	$\bar{X}$FREESLK	MIN%DEMAND	$\bar{X}$%DEMAND	MAX%DEMAND	MINUTIL	$\bar{X}$UTIL	MAXUTIL	$\bar{X}$DMND	MINCON	$\bar{X}$CON	MAXCON	VA-CON	XCON-TM	VA-CON-TM	MINCON-ALL	$\bar{X}$CON-ALL	VA-CON-ALL	TOTFACT	MINOFACT	MAXOFACT	TOTUFACT	MINUFACT	MAXUFACT	$\bar{X}$OVER	MINOVER	MAXOVER	$\bar{X}$UNDER	MINUNDER	MAXUNDER	Média do atraso no mais longo caminho crítico	Desvio Padrão	Estimativa do erro padrão	R^2
HEURÍSTICAS	LFT							+			-								-	+	+		+	+	-			-			+		+		-	-		-		+				197.23	75.46	8.27	0.99
	GRD									+		-						+	-			-	+		-						+	-		-				-						210.88	79.30	13.84	0.98
	GRRD						+	+														-	+	+									+			-		-						207.29	78.73	14.79	0.97
	SIO				+							-	+		+	-		+					+		-		-														+		-	211.04	82.84	16.41	0.96
	GRU		+	+														+		-			+					-	-															212.69	81.84	16.99	0.96
	LFT										+	-							-	+			+		-				-		+		-	+					-	+			196.45	74.93	8.27	0.99
	MJP		+	+				+	-		-		+		+	-				-			+		-	+			-									-		+			+	223.55	88.70	15.74	0.98
	RAN						+	+				-	+	+			-	+			+		+			+		-	-		-								-			-			209.68	78.70	13.72	0.98

(*) Cada coeficiente de regressão com nível de significância de 95%

(+) Coeficiente de regressão positivo

(-) Coeficiente de regressão negativo

(**) Patterson [76] - páginas 114 e 115.

Tabela 5.5 - Análise sensibilidade da performance das regras *

		ATRASO TOTAL	ATRADO PONDERADO	AUMENTO % NO LONGO CAMINHO CRÍTICO	AUMENTO % NA DURAÇÃO DO CAMINHO CRÍTICO
LTF	(+)	$\bar{X}$DUR VA-DUR VA-CPL ΣFREE $\bar{X}$%DEM $\bar{X}$UTIL $\bar{X}$CON MAXCON-TM VA-CON-ALL $\bar{X}$OVER $\bar{X}$SLK-R MIN%DEM NSLK MAXCON	VA-DUR $\bar{X}$SLK-R ΣFREE $\bar{X}$CON MAXCON TOTUFACT $\bar{X}$OVER	NSLK MAX%DEM MINUTIL MAXUTIL $\bar{X}$DMND $\bar{X}$CON-ALL MINOFACT $\bar{X}$UNDER	ΣDUR PDENSITY-F MINUTIL VA-CON-TM VA-CON-ALL MAXOFACT MAXUFACT
	(-)	$\bar{X}$CPL TOTSLK-R $\bar{X}$FREE MINCON-ALL $\bar{X}$CON-ALL MINOFACT MAXOVER MINCON VA-CON $\bar{X}$CONT-TM	$\bar{X}$SLACK $\bar{X}$FREE VA-CON MINOFACT	TOTSLK-R $\bar{X}$%DEM MINCON $\bar{X}$CON-TM TOTUFACT MINUFACT $\bar{X}$OVER	$\bar{X}$CPL TOTSLK-R VA-CON MINCON-TM TOTUFACT
GRD	(+)	ΣFREE $\bar{X}$UTIL MAXCON MINUFACT $\bar{X}$OVER MAXUNDER	$\bar{X}$DUR VA-DUR VA-CPL $\bar{X}$SLK-R MAXCON-TM VA-CON-ALL TOTOFACT TOTUFACT ΣFREE MIN%DEM $\bar{X}$%DEM MINUTIL FREE MAXUNDER MINUFACT $\bar{X}$UTIL MINCON $\bar{X}$CON MAXCON $\bar{X}$OVER	$\bar{X}$SLK MIN%DEM MAXUTIL $\bar{X}$CON-ALL	MINUTIL MAXCON-TM VA-CON-ALL MINOFACT
	(-)	$\bar{X}$CPL MAXUTIL $\bar{X}$CON-ALL	$\bar{X}$CPL TOTSLK-R $\bar{X}$FREE MAX%DEM MAXUTIL VA-CON $\bar{X}$CON-TM MINCON-ALL $\bar{X}$CON-ALL MINOFACT MAXOFACT MAXUFACT MAXOVER	$\bar{X}$SLK-R $\bar{X}$%DEM $\bar{X}$UTIL $\bar{X}$CON VA-CON-ALL MAXOFACT $\bar{X}$OVER	$\bar{X}$%DEM
GRRD	(+)	NSLK $\bar{X}$%DEM MAXCON TOTUFACT $\bar{X}$OVER	$\bar{X}$DUR VA-DUR $\bar{X}$SLK-R ΣFREE $\bar{X}$%DEM MAXUTIL $\bar{X}$CON MAXCON MAXCON-TM TOTOFACT $\bar{X}$OVER MIN%DEM	$\bar{X}$CPL VA-CPL MAXUTIL $\bar{X}$DMND MINOFACT	PDENSITY-F MINUTIL MAXCON-TM VA-COM-TM VA-CON-ALL MAXOFACT MINUFACT
	(-)	VA-CON	$\bar{X}$CPL $\bar{X}$SLK $\bar{X}$FREE MAX%DEM VA-CON XCON-TM MINCON-ALL $\bar{X}$CON-ALL MINOFACT MAXOFACT MINUFACT	$\bar{X}$UTIL MINUFACT $\bar{X}$OVER	VA-CON $\bar{X}$%DEM (RSM)
SIO	(+)	NSLK $\bar{X}$SLK-R $\bar{X}$DMND $\bar{X}$CON TOTOFACT $\bar{X}$OVER $\bar{X}$UNDER MAXCON	VA-DUR $\bar{X}$SLACK-R ΣFREE $\bar{X}$%DEM MINCON $\bar{X}$CON MAXCON TOTUFACT $\bar{X}$OVER	$\bar{X}$DENSITY $\bar{X}$SLK-R NFREE MIN%DEM MAXUTIL MAXOVER	ΣDUR PCTSLK MINUTIL MAXCON-TM MAXOFACT
	(-)	VA-DUR $\bar{X}$CPL $\bar{X}$SLK NFREE MINCON VA-CON-TM MAXOFACT	NDUMMY $\bar{X}$SLK $\bar{X}$FREE VA-CON $\bar{X}$CON-ALL MINOFACT	TOTSLK-R PCTFREE MINCON VA-CON MAXUNDER	COMPLEXTY NSLK VA-CON MINUNDER
GRU	(+)	NDUMMY $\bar{X}$DUR VA-DUR NSLK $\bar{X}$SLK-R FREE MIN%DEM $\bar{X}$%DEM $\bar{X}$UTIL MINCON $\bar{X}$CON MAXCON TOTOFACT MAXUNDER $\bar{X}$OVER	VA-DUR $\bar{X}$SLK-R ΣFREE NFREE MIN%DEM $\bar{X}$%DEM $\bar{X}$UTIL $\bar{X}$CON MAXCON TOTOFACT TOTUFACT $\bar{X}$OVER MINOVER MAXUNDER	$\bar{X}$DUR VA-DUR MIN%DEM MAXUTIL	ΣDUR MINUTIL MAXOFACT MAXUFACT
	(-)	$\bar{X}$CPL VA-CPL TOTSLK-R $\bar{X}$FREE MAX%DEM VA-CON $\bar{X}$CON-TM $\bar{X}$CON-ALL MINOFACT MAXOFACT MINUFACT	$\bar{X}$CPL VA-CPL $\bar{X}$FREE MAX%DEM VA-CON $\bar{X}$CON-ALL MINOFACT MAXOFACT MINUFACT	MAX%DEM VA-CON $\bar{X}$CON-TM	TOTSLK TOTUFACT

Tabela 5.5 - Continuação

		ATRASO TOTAL	ATRASO PONDERADO	AUMENTO % NO LONGO CAMINHO CRÍTICO	AUMENTO % NA DURAÇÃO DO CAMINHO CRÍTICO
LFT	(+)	$\bar{X}$DUR VA-DUR $\bar{X}$SLK-R ΣFREE MIN%DEM $\bar{X}$%DEM MAXUTIL $\bar{X}$CON MAXCON MAXCON-TM TOTOFACT $\bar{X}$OVER $\bar{X}$UNDER	$\bar{X}$DUR VA-DUR $\bar{X}$SLACK ΣFREE MIN%DEM $\bar{X}$%DEM MINUTIL MAXUTIL MAXCON MAX-CON-TM TOTOFACT TOTUFACT $\bar{X}$OVER MAXUNDER	$\bar{X}$SLK MAX%DEM MAXUTIL $\bar{X}$-CON-ALL MINOFACT $\bar{X}$UNDER	$\bar{X}$SLK MINUTIL MAXCON-TM VA-CON-ALL MAXOFACT MAXUFACT
	(-)	COMPLEXTY $\bar{X}$CPL $\bar{X}$SLK $\bar{X}$FREE MAX%DEM VA-CON $\bar{X}$CON-TM VA-CON-TM MINCON-ALL $\bar{X}$CON-ALL MINOFACT MAXOFACT MINOVER	COMPLEXITY $\bar{X}$CPL $\bar{X}$SLK ΣFREE MAX%DEM VA-CON $\bar{X}$CONT-TM VA-CON-TM MINCON-TM X-CON-ALL MINOFACT MAXOFACT MINOVER	TOTSLK-R $\bar{X}$%DEM MINCON $\bar{X}$-CON-TM TOTOFACT MAXOVER	PCTSLK TOTSLK-R VA-CON TOTUFACT
MJP	(+)	NSLK PCTFREE MIN%DEM $\bar{X}$DMND $\bar{X}$CON MAXCON-TM TOTOFACT MAXUFACT MAXOVER $\bar{X}$UNDER	MIN%DEM MAXUTIL TOTOFACT TOTUFACT MAXOVER	$\bar{X}$DUR VA-DUR VA-CPL $\bar{X}$SLK-R NFREE MAXUTIL $\bar{X}$CON MINOVER MAXUNDER	MINUTIL MAXCO-TM VA-CON-ALL MAXOFACT MINOVER
	(-)	$\bar{X}$CPL VA-CPL VA-CON-TM MAXOFACT TOTUFACT	MINOFACT MAXOFACT	NSLK $\bar{X}$SLK PCTFREE MAX%DEM $\bar{X}$eON-TM MAXUFACT MINCON	MAXUFACT MAXOVER $\bar{X}$%DEM
RAN	(+)	VA-DUR NSLK MAXCON TOTUFACT $\bar{X}$OVER	ΣFREE $\bar{X}$%DEM MAXCON TOTUFACT $\bar{X}$OVER	$\bar{X}$CPL VA-CPL $\bar{X}$SLK-R ΣFREE MIN%DEM MINUTIL MAXUTIL $\bar{X}$CON	ΣDUR PCTSLK MINUTIL MAXCON-TM MAXOFACT
	(-)	PCTSLK	$\bar{X}$FREE VA-CON	TOTSLK-R $\bar{X}$FREE VA-CON $\bar{X}$CON-TM MINCON-ALL MINUFACT MAXOVER	VA-DUR $\bar{X}$CPL NSLK $\bar{X}$%DEM MINCON-TM

* Resumo das tabelas 5.1 a 5.4 - experimento de Patterson (76).

Experimento de Thesen (1976)

(n/n/n)

Comparação entre procedimentos utilizando diferentes fatores de urgência, aplicados com a rotina de Knapsack e com o método paralelo.

Características do Problema

. Metodologia usada

Usando parâmetros de número de atividades, duração de atividades, número de restrições de recursos disponíveis, disponibilidade de recursos, configuração da demanda por cada recurso foram gerados problemas com 10 a 200 atividades; 1,5 e 10 recursos sendo limitados, demanda por recurso igualmente distribuída entre um limite superior de 50 a 100 unidades

$$(0,5 < UTIL_k < 1,0)$$

e um limite inferior de 0 a 50 unidades. $(0 < UTIL_k < 0,5)$. Apresenta o fator de conteúdo de informação na rede (I) relacionando relações de precedência existentes (arcos não redundantes) e arcos possíveis entre atividades, porém não existentes (arcos disjuntivos). Os problemas gerados atendem aos valores de I = { 0, 0.2, 0.4, 0.6, 0.8, 1.0).

I é estimado por $\hat{I} = \frac{L_N}{L_N + L_D}$

onde L_N = número de arcos não redundantes;

L_D = número de arcos disjuntivos.

Cada problema foi resolvido por dez modos diferentes, considerando cinco fatores urgência (MINSLK, duração da atividade, fator de atraso, fator de recurso e C_i, aplicados pelo algoritmo de Kanapsack para combinação do conjunto de atividades que deverá ser escolhido para programação e pelo método paralelo, quando a ordenação é sobre o valor individual para cada atividade.

Fator urgência c_j (Thesen)

$$c_j = \frac{d_j \, r_{jk}}{(R_k - \bar{R}_k)} + \frac{a \, . \, b \, (T - SLK_j)}{(b - 2a) \, SLK_j} + T \, . \, b$$

onde:

$$SLK_j = LCT_j - d_j - t_{NOW}$$

(folga existente no estágio da programação)

$T = d_j$ (limite superior para solução do problema)

a = constante fixando o fator urgência de uma atividade com folga zero (crítica)

b = constante fixando o fator urgência de uma atividade com o maior atraso.

A segunda parcela de c_j é chamada D_j e reflete o impacto na duração do projeto, pelo atraso de uma atividade (fator de atraso) e o fator de recurso reflete a contribuição da atividade pelo uso do recurso disponível no período (fator de recurso).

A análise sobre os procedimentos é feita considerando o resultado segundo: eficiência computacional (espaço e tempo computacional requerido), e eficácia do procedimento (qualidade da solução) oferecendo os seguintes resultados:

TEMPO COMPUTACIONAL (Redes com I = 0.2; NNDDES = 5.0; k = 10)

Resultado usando o Método da Mochila (Knapsack)					
	Fatores de Urgência				
Criticalidade	99999-LFT	D_i (EQ4.3)	R_i (EQ4.2)	d_i	C_i (EQ4.1)
0	0,33	0,33	0,33	0,33	0,33
0,29	0,33	0,33	0,33	0,33	0,33
0,50	0,45	0,44	0,44	0,38	0,39
0,71	0,70	0,54	0,56	0,48	0,47
1,03	0,85	0,67	0,56	0,49	0,48
1,24	0,96	0,72	0,59	0,50	0,49

Thesen (1976) - página 420.

Eficiência Analítica dos Algoritmos com Diferentes Fatores Urgência

Fator Urgência	Melhor Solução		Qualidade de Solução*	
	Knapsack	Sequenciamento (Método Paralelo)	Knapsack	Sequenciament (Método Paralel
	%	%	%	%
Duração	75	25	18	12
Requisito de recurso	80	20	31	15
MINSLK	58	42	69	81
L F T	100	0	0	0
e $\frac{\text{Slack}}{\text{Maxslack}}$	100	0	0	0
log (Slack)	55	45	11	22
Constante	72	28	18	18
C_j com a = 10 b = 700	81	19	73	33

(*) Percentagem de soluções próximas em 5% da melhor achada.

Extraída de Thesen (1976), página 421.

Conclusões

A eficiência computacional para redes com conte de informação (I) iguais, aumenta linearmente com o aumento no nho do problema, número de recursos limitados e criticalidade recursos; a taxa de crescimento do tempo computacional com recu críticos é proporcional a I, ou seja, a eficiência computaciona pende estreitamente das características do problema.

Quanto à qualidade da solução verificou-se que cedimento com rotina de resolução por Knapsack é mais eficiente se usando os fatores de urgência para sequenciamento das ativi

individualmente. Confirma a performance da heurística MINSLK, como a de melhores resultados.

A análise sobre a variabilidade das soluções encontradas, relacionando estrutura da rede (I) e criticalidade dos recursos, verifica-se que há dependência do conteúdo de informação (I), embora para cada I, não há diferença substancial entre os resultados gerados pelos fatores de urgência.

Para restrições de alta criticalidade e mesmo conteúdo de informações a amplitude dos resultados obtidos é grande.

Experimento de Elsayed (82)

(n/n/1)

Análise das heurísticas propostas: ROT, ROT-ACTIM e ROT-ACTRES, comparando-as com GENRES, ACTIM, ACTRES e TIMRES, sob o critério de minimização da duração do projeto, programado sob limitação dos recursos (método seriado).

Características do Problema e Metodologia de Análise

Utiliza 16 projetos, com número de atividades variando de 6 a 38 e número de nós variando de 5 a 22, com requisito de recursos proporcional à disponibilidade, para cada atividade, variando entre 0,1 e 1,0 de um único recurso.

Apresenta os resultados obtidos pelo processamento de cada rede, pelo método seriado, para diferentes níveis de recursos, sob todas as regras.

Regra	Número de vezes em que apresentou a menor duração
GENRES	19
ROT/ACTIM	18
ROT/ACTRES	14
ACTIM	10
TIMRES	10
ACTRES	6
ROT	5

Conclusão

Nenhuma destas regras de priorização garante a duração mínima em qualquer dos casos analisados (mesma rede sob diferentes níveis de recursos).

Experimento de Kurtulus & Davis (82)

(n/n/n)

Comparação entre as heurísticas conhecidas: SOF (SIO), MINSLK (LTF), FCFS e seis outras desenvolvidas especialmente para o problema de multiprojetos (abordagem de multiprojetos): SASP, LALP, MOF, MAXSLK, MINTWK, MAXTWK, aplicadas pelo método paralelo, no problema de alocação de multirecursos, na busca da solução do menor atraso total. Propõe 2 medidas sumárias, próprias à análise do problema de multiprojeto:

ARLF - Fator de carga média de recurso por problema (indicando o centro de massa do DCR_k, para sua programação mais cedo).

AUF - Fator médio de utilização do recurso (indicando a rigidez das restrições do recurdo k).

Características dos problemas

O número total de atividades por problema variou entre 34 e 63. Pela combinação de 7 níveis de ARLF (-3, -2, -1, 0, 1, 2, 3) e 11 níveis de AUF (0.6, 1.6), 77 problemas foram tratados pelas heurísticas e analisados sob o critério de mínimo atraso total.

Metodologia de Análise

Considera que: não existe nenhum indicador de pro-

jetos reais, reunidos pelo uso comum de recursos limitados , tenham seus atrasos seguindo uma distribuição normal; os erros de previsão da utilização das regras, sobre cada problema são independentes.

Sob estas hipóteses utiliza o procedimento não-paramétrico de duas entradas para testar o atraso que adviria da aplicação da regra, sob condições de AUF e ARLF.

Para usar o modelo de duas entradas, para o qual é possível testar o efeito de tratamento do problema pelas diferentes heurísticas, relaciona cada nível de ARLF com os valores médios sobre AUF e vice-versa (two-way layout). O resultado obtido é analisado pela localização (Rank) frente aos demais. São transcritas as tabelas obtidas.

Ordenação das oito melhores regras obtidas dos resultados gerais

Baseado no atraso médio[a]

1	SASP	71.05
2	MAXTWK	74.39
3	FCFS	87.60
4	MINSLK	87.90
5	LALP	88.60
6	MOF	90.56
7	MAXSLK	91.40
8	SOF	92.38

Baseado no Rank médio [b]

1	SASP	1.81
2	MAXTWK	2.87
3	MINSLK	3.92
4	FCFS	4.27
5	MAXSLK	4.53
6	MOF	4.80
7	SOF	4.96
8	LALP	4.99

Baseada no número de vezes em que foi a única no primeiro lugar[c]

1	SASP	43
2	MAXTWK	18
3	MINSLK	13
4	MAXSLK	6
5	MOF	5
6	FCFS	3
7	SOF	1

a. dados obtidos pela mediação do atraso total sobre todos os valores de AUF e ARLF (i.é, todos os problemas testados)

b. dados obtidos mediando o número de vezes em que a regra, sobre todos os valores AUF e ARLF, obteve o melhor resultado, sobre todos os problemas.

c. nenhuma outraregra foi ordenada em primeiro.

Extraída de Kurtulus & Davis (82).

Tabela 3 - Categorização de acordo com ARLF e AUF*

	ARLF						
AUF	-3	-2	-1	0	1	2	3
0.6	MINSLK[a]	MAXSLK	SASP	MINSLK	MAXSLK[d]	MINSLK	e
0.7	MINSLK	SASP	SASP	MINSLK	MAXTWK	MINSLK	e
0.8	MINSLK	MAXTWK	SASP	MINSLK	SASP	MINSLK	MINSLK[f]
0.9	SASP	MAXTWK[b]	SASP	MINSLK	SASP	MAXTWK	MINSLK
1.0	SASP	SASP	MAXTWK	MAXSLK	MAXTWK[d]	MOF	MINSLK[b]
1.1	SASP	SASP	MAXTWK[b]	MAXSLK	SASP	SASP	SASP
1.2	SASP	SASP	MAXTWK	SOF[c]	SASP	SASP	SASP
1.3	MAXSLK	MAXTWK	MAXTWK	SASP	SASP	MOF	SASP
1.4	SASP	SASP	MAXTWK	SASP	SASP	MAXTWK	SASP
1.5	SASP	SASP	MAXTWK	MAXTWK	SASP	SASP	SASP
1.6	MAXTWK	SASP	MAXTWK	SASP	SASP	MOF[b]	SASP

* Esta tabela contém a melhor regra sob cada valor de ARLF e AUF, baseada na média do atraso ideal.

[a] empate com MOF e MAXTWK.

[b] empate com SASP.

[c] empate com MAXSLK e SASP.

[d] empate com FCFS.

[e] omitida devido a atraso zero serem obtidas pela combinação de AUF e ARLF.

[f] empate com MOF, MAXTWK, FCFS, SASP.

Extraída de Kurtulus & Davis (82), pag.167.

A categorização das regras foi obtida, comparando-se para cada nível de uma medida sumário, para todos os níveis da outra, a melhor regra firmada pelos resultados do modelo de duas entradas, com a seguinte melhor regra. Se o resultado era reafirmado, o procedimento era descontinuado, se não o teste prosseguia, até que uma melhor regra era escolhi-

da da mesma forma, ou a lista das regras era exaurida. No caso de empate de duas regras abria-se um "conjunto de decisão" e testava-se uma terceira regra (Teste par a par pelo procedimento WSRT - Wilcoxon Signed Ranks Test) as regras, então, foram ordenadas pelo número de vezes em que foi a primeira do Rank, pela média do atraso total obtido no nível e, finalmente, pelo Rank médio obtido.

Tabela 4 - Sumário de performance das regras sob categorização de ARLF*

	Rank	Atraso total médio	Ordenado no atraso total	Número de vezes em 1º
ARLF=-3.0	1a.	SASP	SASP	SASP
	2a.	MAXTWK	MAXSLK	MINSLK
	3a.	MAXSLK	MAXTWK	MAXTWK
	4a.	MINTWK	MINTWK	MAXSLK[a]
ARLF=-2.0	1a.	SASP	SASP	SASP
	2a.	MAXTWK	MAXTWK	MAXTWK
	3a.	LALP	LALP	MAXSLK
	4a.	MINTWK	MINSLK	e
ARLF=-1.0	1a.	MAXTWK	MAXTWK	MAXTWK
	2a.	SASP	SASP	SASP
	3a.	LALP	LALP	e
	4a.	FCFS	FCFS	
ARLF=-0.0	1a.	MAXTWK	MAXTWK	MINSLK[b]
	2a.	SASP	SASP	MAXSLK
	3a.	MAXSLK	MAXSLK	SOF[c]
	4a.	MINSLK	MINSLK	e
ARLF= 1.0	1a.	SASP	SASP	SASP
	2a.	FCFS	FCFS	FCFS[c]
	3a.	MAXTWK	MAXTWK	MINSLK[d]
	4a.	LALP	SOF	e
ARLF= 2.0	1a.	MOF	MOF	SASP
	2a.	SASP	SASP	MINSLK[a]
	3a.	MAXTWK	MINSLK	MAXTWK
	4a.	MINSLK	MAXTWL	e
ARLF= 3.0	1a.	SASP	SASP	SASP
	2a.	MINSLK	MINSLK	MINSLK
	3a.	FCFS	FCFS	FCFS
	4a.	MINTWK	MINTWK	MOF[c]

[a] empate com MOF.

[b] empate com SASP.

[c] empate com MAXTWK.

[d] empate com MAXSLK.

[e] nenhuma regra foi qualificada para esta posição.

Extraída de Kurtulus & Davis (82), pag.169.

Conclusões

Sugere que a abordagem de multiprojetos é mais eficiente que a de projeto único, já que as regras específicas para a primeira (SASP e MAXTWK) são as mais eficientes [1].

Confirma a significância dos parâmetros ARLF e AUF sobre a performance das regras, embora para ARLF = 3 (obstrução final alta) e ARLF = 0 (centro de massa centralizado em DCR_k) os resultados obtidos sejam muito próximos. Verifica que a significância de ARLF, relacionada aos vários níveis de AUF, é inversamente proporcional ao seu crescimento.

A análise sobre AUF categoriza as regras mais efetivas do seguinte modo:

Tabela 6 - Resultados*: As duas melhores regras sob a categorização de AUF

AUF	Rank	Baseado no atraso	Baseado no Rank	Baseado no número de vezes que teve 1º Rank
0.6	1.	MINSLK	MINSLK	MINSLK
	2.	FCFS	FCFS	MAXSLK
0.7	1.	MINSLK	MINSLK	MINSLK
	2.	SASP	SASP	SASP
0.8	1.	SASP	SASP	MINSLK
	2.	MINSLK	MINSLK	SASP
0.9	1.	SASP	SASP	SASP
	2.	MAXTWK	MINSLK	MINSLK
1.0	1.	MAXTWK	SASP	SASP
	2.	SASP	MAXTWK	MAXTWK
1.1	1.	SASP	SASP	SASP
	2.	MAXTWK	MAXTWK	MAXTWK

* Kurtulus & Davis (82), pag.

(1) Esta conclusão é passível de crítica, na medida em que, para comparação das duas abordagens, é necessário análise sobre critérios equivalentes como feito por Patterson (76).

Usando os dados de Patterson (76), verifica-se que as melhores regras, listadas em ordem foram: SASP, SOF (SIO), MINSLK e MAXTWK. Justifica a performance da SIO e MINSLK, devido a que os problemas têm variação de ARLF entre 0.01 e 0.6 (não restritivas).

Orienta, finalmente, a aplicação das heurísticas segundo os valores das medidas sumárias, pelo seguinte quadro:

Tabela 7 - Sugestões**: Melhor regra para várias categorias de ARLF e AUF

	AUF	
ARLF	0.6 a 0.8	0.9 a 1.6
-3.5 a -2.5	MINSLK	SASP
-2.5 a -1.5	MAXTWK	SASP
-1.5 a -0.5	SASP	MAXTWK
-0.5 a 0.5	MINSLK	SASP, SOF, MAXTWK*
0.5 a 1.5	SASP	SASP
1.5 a 2.5	MINSLK	MOF, SASP
2.5 a 3.5	MINSLK	SASP

* mais que uma regra é recomendada. A escolha depende do segundo critério.

** Kurtulus & Davis (82), pag.171.

Experimento de Whitehouse (83)

(n/n/1)

Comparação das heurísticas GENRES, ACTRES, ACTIM e TIMBRES, aplicadas com o algoritmo CONSOAL. Redes únicas com atividades demandando um único tipo de recurso.

Característica do Problema e Procedimentos Analisados

Teste comparativo sobre um projeto (rede com 35 atividades nos arcos e 22 nós) sob diferentes níveis de disponibilidade, utilizando o algoritmo BAG, sequenciando as tarefas pelo valor obtido através das regras e programando tantas quantas forem possíveis. (Método seriado) e pelo algoritmo CONSOAL, quando o valor da heurística para cada atividade é ponderado sobre a soma dos valores de todas as atividades concorrentes.

No modelo de busca GENRES vários sequenciamentos são obtidos, através da combinação linear convexa das regras ACTIM e ACTRES, para diferentes pesos W:

GENRES = W (ACTRES) + (1 - W) (ACTIM)

e a escolha é sobre aquela de menor duração.

Conclusões:

Os resultados obtidos com a aplicação do procedimento COMSOAL são melhores do que os procedimentos heurísticos tradicionais (sem ponderação do valor da regra).

Apresenta outras cinco estratégias para aplicação com o COMSOAL.

Experimento de Gordon (83)

(n/n/1)

Projeto único, requisito único, vários tipos de recursos. Comparação entre método paralelo e método seriado.

Gordon (83) apresenta os resultados de seu experimento, utilizando projetos reais, categorizados, de acordo com 9 parâmetros que definem a localização do projeto numa escala variando de zero a 120, com a maioria das redes com valores entre 10 e 70. Os parâmetros foram escolhidos dentre 18, explorados em pesquisa anterior a este experimento, a qual ainda não tive acesso e "tendem a localizar redes curtas e carregadas aos menores valores e redes longas, esparsas e sequenciais a valores maiores". [Gordon (1983), pag.165]. O número de atividades por rede, variando de 12 a 180 atividades, repressenta uma diferenciação acentuada entre os 47 elementos da amostra, com uma média de 56 atividades por rede. A condição imposta as atividades: necessidade de reunir os recursos necessários, durante a execução, nos leva a aventar a hipótese de tratamento do problema como de atividades requerendo único recurso. Embora esta restrição, cada rede requer em média 5 e no máximo 15 tipos de recursos. A disponibilidade foi taxada, no nível de 70% a 50% do pico de utilização fornecido pelo DCR para programação mais cedo. (A demanda por recurso de uma atividade foi trocada, caso a original excedesse à disponibilidade). Isto reflete uma situação de criticalidade para todos os recursos (RA = 70%; RB= 50%) a cada experimento. Os resultados dos efeitos das regras sobre o aumento percentual na duração do caminho crítico, considerada a variação na escala, são plotados nas curvas das

figuras 5.1 (a) e (b) e 5.2 (a) e (b). A análise serve à comparação entre a aplicação das regras nos métodos seriado e paralelo.

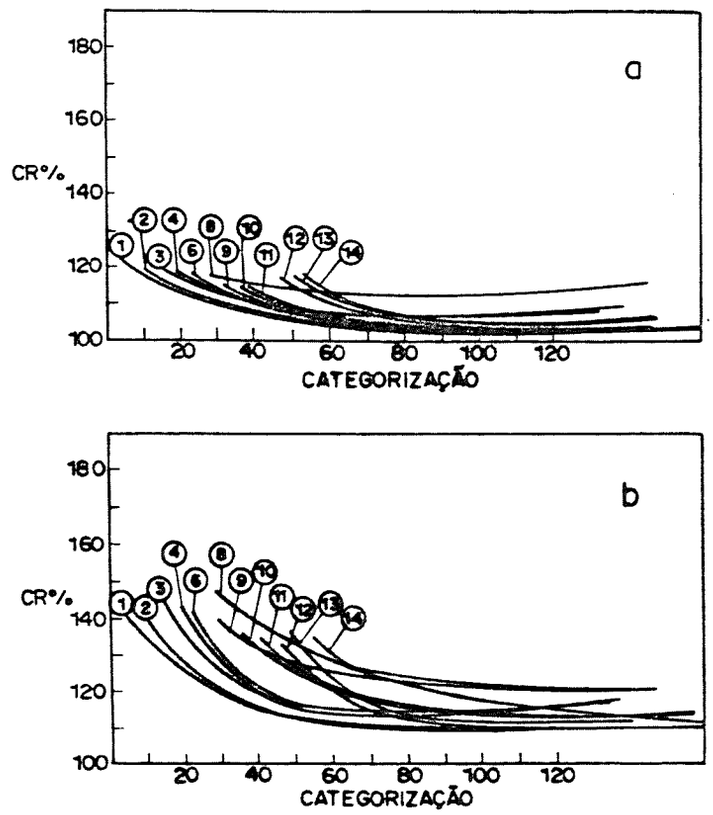

Figura - Método paralelo: a) RA=70%, b) RB=50%.

(*) os dados circulados identificam as regras.

(**) RA e RB significam o limite calculado como percentual do pico de utilização do recurso identificado pela programação mais cedo da análise do caminho crítico pelo CPM -padrão.

(***) extraídos de Gordon (83), páginas 165 e 166.

A análise foi orientada para conhecer a performance de duas heurísticas combinadas em ordem sequencial, a segunda sendo usada para o caso de necessidade de desempate. As regras testadas são identificadas, segundo a numeração abaixo:

1- LFT/LTF	8- SIO/LST
2- LST/LTF	9- MOF/LST
3- EFT/LTF	10- ROT/LST
4- EST/LTF	11- MOT/LST
5- LTF/SIO	12- ROF/LST
6- DUT/LST	13- MOF/LST
7- SUF/LST	14- NI/NF

O efeito da combinação das regras serviu para testar que no método paralelo, a LTF e LST são equivalentes [cuja prova é apresentada em Davis & Patterson (75), página 953] e a regra para desempate não influi significamente na definição dos resultados.

Dos resultados apresentados o que mais chama atenção é que para o método paralelo há uma melhoria crescente , geral para todas as regras, conforme as redes aumentam seus valores na escala de categorização, e a partir do valor 70, praticamente, não é apresentada nenhuma diferenciação entre as regras, a menos da SIO (com pior performance). Mesmo para os valores menores, as regras, no geral, tem performance uniformemente crescente com a escala proposta. Podemos inferir que a busca em amplitude das programações parciais é mais eficiente em redes longas para o uso de todas as regras, a menos das LFT e LST (LTF), do que no método de busca em profundidade (seriado), para o caso de menor criticalidade nos recursos.

Já o método seriado, apresenta zonas marcantes de sensibilidade no uso das regras, sendo as regras LFT e LST , as que apresentam menos oscilação em função da categorização dos projetos, e no geral os melhores resultados. Apresentam , ainda, no geral, melhor performance para redes à direita na escala de categorização que no método paralelo.

O efeito não uniforme das regras devido a escala de categorização, quando aplicados no método seriado, pode justificar os resultados contraditórios apresentados por outras experiências.

A conclusão definitiva é que nenhum dos dois métodos e nenhuma das heurísticas dá o melhor resultado para todas as categorias de redes.

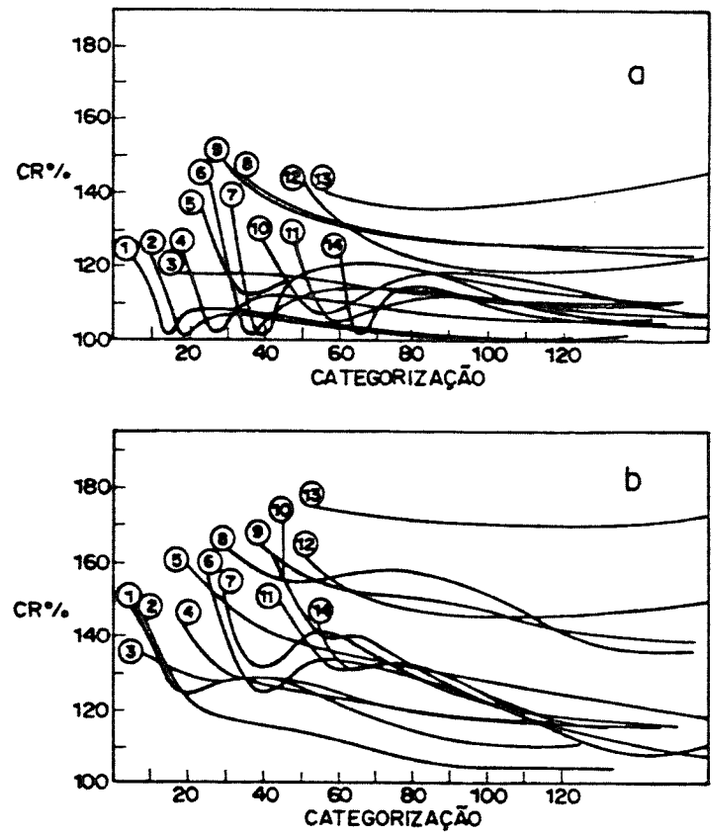

Figura - Método seriado: a) RA=70%, b) RB=50%

Experimento de Kurtulus & Narula (85)

Análise da performance das regras MAXTWK, MINSLK , MINTWK, SOF e regras definidas para o experimento: SLKPEN/MAXSLK; SASP/MAXPEN (SASP1), SASP/FCFS, MAXTOP/GRES (LALP), MAXPEN/GRES (LALP), MAXPEN/FCFS (FCFS); DURPEN/GRES (LOF).

A particularidade deste experimento é a consideração de penalidades desiguais para os projetos. A abordagem é de multiprojetos e o método paralelo.

Características dos problemas

Os problemas são caracterizados por 5 medidas: fator caga média por recurso (ALF); fator médio de utilização (AUF); complexidade (Pascoe); função penalidade (PEN); tamanho do problema (NAS).

A medida ALF (índice de localização do centro de massa de cada um dos diagramas de carga de recurso), difere da ATLF, por ponderar a localização dos picos de utilização por quartas partes ao horizonte fornecido pelo caminho crítico (programação mais cedo). A combinação de projetos nos problemas gera oito níveis de ALF ={-3.0, -2.0, -1.2, -0.8, 0.8 , 1.5, 2.4, 3.0}.

Para os problemas com valor de ALF = -3.0 ou +3.0 (obstrução inicial e final altas, respectivamente) é obtida um valor mínimo, aproximadamente igual a 1.3, para a complexidade, considerando o número de arcos necessários para conectar cada rede e um valor máximo aproximada a 1.75. Considerou,

ainda, um terceiro valor para a complexidade (1.54).

A priorização dos projetos é definida por 6 funções: maior demanda total por recursos; menor demanda total por recursos; maior duração do caminho crítico; menor duração do caminho crítico; igual para todos e aleatória. Esta última é uma forma de referenciar a importância em se definir penalidades específicas aos projetos.

Metodologia de análise

A mesma metodologia usada no experimento de Kurtulus e Davis foi estabelecida para o experimento, desde a geração dos projetos testes, até a análise estatística sobre os resultados das regras (método não paramétrico, layout de 2 entradas considerando a relação de variação dos níveis de um fator, pela média dos demais).

Assim, foram obtidos os seguintes resultados por Rank, relacionados à penalidade:

Resultados*: Número de vezes que cada regra teve o primeiro Rank

PENALIDADES DESIGUAIS	
Regra	f
MAXPEN	717
MAXTOP	469
SASP2	459
SASP1	445
MINSLK	422
MAXTWK	420
DURPEN	241
SOF	147
MINTWK	76
SLKPEN	71

PENALIDADES IGUAIS	
Regra	f
SASP1	210
SASP2	207
MINSLK	125
MAXTWK	102
RCRS	64
SOF	59
LOF	41
MINTWK	34
LALP	25
MAXSLK	16

* Kurtulus & Narula (85), pag.63.

Considerando-se penalidades diferentes, as melhores performances obtidas foram das regras MAXPEN, MAXTOP, o que era de se esperar, na medida em que as duas estão associadas à penalidade e a terceira melhor foi a SASP2. Este último resultado é um pouco surpreendente, por ela ter sido melhor classificada que a SASP1, regra de maior eficiência para o caso de considerar-se penalidades iguais. [Resultado coincidente com o de Kurtulus e Davis (82)]. Ou seja, o fato de priorizar a seleção de atividades pela menor duração e em caso de desempate, a maior penalidade, é menos eficiente que se a decisão sobre o empate, for feita pela primeira da fila já ordenada.

Na seleção das melhores regras por tamanho de problema, a MAXTWK foi mais efetiva para pequenos problemas com penalidades iguais e a MAXPEN para o caso de penalidades diferentes. (Vide Tabela).

Para grandes problemas, as três melhores regras, sob penalidades diferenciadas, foram MAXPEN, SASP2 e SASP1 e, não diferenciando os projetos, foram SASP1/SASP2 e MINSLK, confirmando os resultados de Kurtulus e Davis (82).

Sob a categorização da AUF os resultados obtidos para penalidades iguais, não são totalmente concordantes com os apresentados por Kurtulus e Davis:

AUF	Regra melhor
0.9	SASP
1.3	MAXSLK
1.6	MAXTWK

Resultados*: Classificação de regras por tamanho do problema

PENALIDADES DESIGUAIS

Pequenos		Grandes	
Regra	f	Regra	f
MAXPEN	380	MAXPEN	337
MAXTWK	324	SASP2	312
MAXTOP	248	SASP1	278
MINSLK	196	MINSLK	226
SASP1	167	MAXTOP	221
SASP2	147	DURPEN	131

PENALIDADES IGUAIS

Pequenos		Grandes	
Regra	f	Regra	f
MAXTWK	88	SASP1,2	133
SASP1	77	MINSLK	69
SASP2	74	SOF	42
MINSLK	56	MINTWK	23
FCFS	45	FCFS	19
LOF	29	MAXTWK	14

* Kurtulus & Narula (85).

Resultados*: As duas melhores regras para cada valor AUF**

PENALIDADES IGUAIS

0.6		0.7		0.8		0.9		1.0		1.1	
Regra	f	Regra	f	Regra	f	Regra	f	Regra	f	Regra	f
MINSLK	36	MINSLK	28	MINSLK	15	MINSLK	14	SASP1,2	17	SASP1	20
MAXTWK	18	FCFS	10	SASP1,2	14	MAXTWK		MAXTWK	12	SASP2	19
						SASP1,2	12			MAXTWK	10

1.2		1.3		1.4		1.5		1.6	
Regra	f	Regra	f	Regra	f	Regra	f	Regra	f
SASP1,2	20	SASP1	29	SASP1	27	SASP1	28	SASP1,1	28
MAXTWK	11	SASP2	28	SOF	9	SASP2	27	SOF	9
		MAXTWK	7			SOF	11		

PENALIDADES DESIGUAIS

0.6		0.7		0.8		0.9		1.0		1.1	
Regra	f	Regra	f	Regra	f	Regra	f	Regra	f	Regra	f
MINSLK	131	MINSLK	91	MINSLK	57	MAXPEN	56	MAXPEN	52	MAXPEN	64
MAXTOP	102	MAXTOP	85	MAXTOP	52	MAXTWK	53	MINSLK	39	SASP2	48

1.2		1.3		1.4		1.5		1.6	
Regra	f	Regra	f	Regra	f	Regra	f	Regra	f
MAXPEN	58	MAXPEN	75	MAXPEN	88	MAXPEN	76	SASP2	74
SASP2	51	SASP1	60	SASP2	57	SASP1	65	MAXPEN	69
								SASP1	

* Kurtukus & Narula (85) - pag.65.

** Baseado no número de vezes que obteve o primeiro Rank. A terceira regra é indicada, quan

embora confirme que para alta obstrução inicial, a melhor regra é a MINSLK, seja diferenciando ou não os projetos. A desempenho das regras, sob os níveis de ALF, refletem a ordenação pelo Rank das mesmas, a menos de ALF = 1.20, quando a melhor regra é MINSLK.

Sob penalidades desiguais, a MAXPEN é a melhor para 6 dos 8 valores de ALF. Apenas para ALF = -1.2 e ALF = 3.0 , situa-se em terceiro lugar, após MAXTOP e MAXTWK.

Quanto à complexidade, as melhores regras são: SASP1 e MAXPEN para penalidades iguais e diferentes, respectivamente.

As sugestões para implementação das regras é sumarizada, como segue, e são, no geral, concordantes com a de Kurtulus & Davis (82), apenas agregando diferenciação pelo tamanho do problema.

Sugestões para implementação do procedimento*

PENALIDADES IGUAIS		
	(0.6 ≤ AUF ≤ 0.8)	(0.9 ≤ AUF ≤ 1.6)
pequenos	MINSLK	MAXTWK
grandes	MINSLK	SASP

PENALIDADES DESIGUAIS		
	(0.6 ≤ AUF ≤ 0.8)	(0.9 ≤ AUF ≤ 1.6)
pequenos	MINSLK	MAXPEN
grandes	MINSLK	MAXPEN

* Kurtulus & Narula (85).

PESQUISAS ORIENTADAS PARA ANÁLISE DA PERFORMANCE DAS HEURÍSTICAS UTILIZADAS NO PROBLEMA DE ALOCAÇÃO DE RECURSOS LIMITADOS EM PROGRAMAÇÃO DE PROJETOS

PESQUISADOR	ABRODAGEM E CARACTERÍSTICAS GERAIS DO PROBLEMA	METODOLOGIA USADA	REGRAS ANALISADAS	MELHORES REGRAS E RESULTADOS
Brand, Meyer & Shaffer (1964)	Projeto único Multi recursos Critério: duração	Comparação entre Softwares "MPS"; "CPMS"; "RSM"		RSM Software "MPS"
Mize (1964) n/1/1	Multiprojetos, job-shop Critério: atraso	Experimento sobre 8 problemas simulados contendo 3, 4 e 5 projetos, o maior com 133 atividades. Considera 20 recursos	12 regras: 3 simples e 9 complexas baseadas na menor folga total	LFT (projeto), FCFS (job-shop) . diferença entre job-shop e programação de projetos. . relação entre projetos, quando há conflito
Pascoe (1965)	Único projeto, Multi recursos	32 redes com 20 atividades. 3 tipos de recursos por atividade. Análise de variança sobre a performance das regras e parâmetros para classificação do problema: complexidade, densidade, fator de obstrução.	10 regras LTF, LFT, LST	não contradiz a hipótese nula que não há diferença entre as regras para a maioria das medidas propostas. Sugere LFT e LST.
Knight (1966)	Multi projetos. Redes lineares. Multi recusros Critério: Makespan	6 problemas de 10 ou 11 projetos com 2 a 7 atividades. 4 recursos.	SIO, LONG, GRU	Regras baseadas no uso de recursos são superiores àquelas baseadas unicamente no tempo.
Mueller-Mehrback (1967) 1/1/1	Projetos com condições de job-shop Critério: comparação com a ótima	42 redes. 1 recurso. Teste comparativo		LST e LFT
Crowston (1968)	Projeto único. Multi recursos			Min LST, ajustado a cada iteração
FENDLEY (1968)	Multi projeto. Multi recursos. (Abrodagem de multi projetos). PERT. Critério 1: atraso total e ponderado. Makespan Critério 2: eficiência na utilização dos recursos Critério 3: tamanho das filas de expectativa (número médio e máximo de atividades, volume de trabalho)	Teste sobre 2 problemas contendo 3 e 4 projetos simulados. Vários níveis de disponibilidade. Obstrução inicial. Proposta de sistema para previsão de atraso, relacionado à regra MSF, e fator de carga de recurso	SOF (SIO); MAR; MSA; MMSA; MSF (LFT); MCA; FIFO; MMSF	Resultado relacionado à carga de recurso (demanda/disponibilidade) e à regra aplicada. Não existe única melhor regra para todos os problemas e todos os critérios, mas MSF teve melhor performance para C.1 e C.2. A SOF e a MMSF forma melhores para C.3. Previsão no atraso depende: dos projetos em progresso (requisito e disponibilidade). Propõe medida sumária baseada na carga de recurso para programação mais cedo
Gonguet (1969)	Único projeto; multi recursos			LFT

Performance das regras para programação de multiprojetos

Os experimentos analisadospara comparação entre heurísticas usadas para programação de projetos sob recursos limitados, para problemas reunindo mais que um projeto, seguem uma trajetória de busca e de características dos problemas para sugestão da regra a ser usada (heurística projetada).

A intenção destes trabalhos é montar um padrão para projetar uma heurística, dada determinada configuração do problema. Porém, como Patterson notou, este padrão de sugestão pode ser rapidamente desatualizado, devido a descoberta de novas heurísticas até então não consideradas. Assim foi com o experimento de Davis e Kurtulus, quando as regras mais eficientes foram específicas para a abordagem de multiprojetos e após, isso, o experimento de Kurtulus e Narula,considerando penalidades para os projetos. Mesmo para estes dois experimentos, realizados segundo mesma metodologia, quando comparadas as suas sugestões, as heurísticas projetadas para condições iguais, são diferentes.

Os resultados analisados apresentam algumas discordâncias, que podem ser explicadas pelo uso de diferente abordagem para agrupamento dos projetos, consideração de critérios não semelhantes como objetivo da prgoramação e amostras desiguais tomadas em cada experimento.

A regra de menor folga se encontra com maior constância, nas listas dsa mais eficientes, sendo indicada como a de melhor performance quando usada na abordagem de projeto único. [Davis e Patterson (76), Davis (75)], considerando a mi

nimização do atraso sobre a duração do caminho crítico. Prioriza as atividades do caminho crítico, prevenindo o atraso no projeto. Esta regra é considerada como secundária para a abordagem de multiprojetos, quando a criticalidade dos recursos é alta, sendo indicada, para esta abordagem, apenas quando há recursos necessários para atender a demanda, dentro da duração do caminho crítico (AUF < 0.8).

Para a abordagem de multiprojetos as regras associadas à liberação mais rápida de frentes de trabalho, seja pela priorização de atividades mais curtas (SIO - Patterson (76), SASP - Kurtulus e Davis (82), SASP1 - Kurtulus e Narula (85)] ou pela priorização de atividades com maior utilização de recursos [MJP - Patterson (75); MAXTWK - Kurtulus e Davis (82)], Kurtulus e Narula (85)] são as indicadas.

Depreende-se destes resultados que o problema de alocação de recursos escassos que tem seu ponto crítico, na obstrução para programação das atividades, quando tratado de forma a ter mais rapidamente, condições de fluidez para o progresso da sua programação, poderá ser tratado com maior eficiência.

No caso de priorizar as menores atividades, abrem-se frentes de trabalho: as atividades sucessoras, que poderão estar se combinando no uso dos recursos, conforme fossem liberados (folgas de recursos mais eficientemente alocados).

No caso de priorizar atividades de forma a se ter menor ociosidade, garante que o conteúdo de trabalho para todos os projetos estará mais rapidametne decrescendo, o que flexibiliza a programação para os períodos finais.

Desta forma, uma heurística eficiente deve considerar o trabalho já realizado, controlando a ociosidade, e o trabalho que está por vir, controlando a duração do caminho crítico, ou do horizonte realizável frente às restrições dos recursos.

Assim sendo, propomos uma categorização analítica para as heurísticas até hoje sugeridas, de forma a que tenham mais informações relativas a uma ordem (controle da ociosidade do trabalho realizado) e a outra ordem (controle sobre o horizonte realizável).

Esta classificação tem sua base teórica nos fundamentos de busca da trajetória ótima, pelo algoritmo A^* para resolução dos sistemas de produção, como definido por Nilsson (80) e apresentados a seguir.

Proposição de uma abordagem analítica para análise das regras

SOLUÇÃO POR SISTEMA DE PRODUÇÃO EM IA

Partimos da hipótese que um procedimento heurístico é um sistema de produção em IA, como definido por Nilsson(82), cuja base de dados global é o conjunto das programações parciais geradas em cada estado do Espaço do problema de programação de multiprojetos com limitação sobre os recursos.

A base procedural consiste das regras de produção (heurísticas) e de uma estratégia de controle sobre a busca da solução (inferência sobre o uso das heurísticas na geração das programações parciais).

O grafo do projeto, expressando as relações de precedência e a análise sobre o uso de recursos, alimentam a construção do grafo explicito , G, grafo de busca e um subconjunto T, de G chamado de árvore de busca. Cada nó (exceto o inicial) em G tem um ponteiro ligando-o a seus nós pais em G, e um único ponteiro identificando seu nó pai em T.

O grafo de busca forma uma ordenação parcial porque nenhum nó em G é seu próprio antecessor. Cada trajetória para um nó descoberto pelo algorítmo é preservado explicitamente em G; existe uma única trajetória para aquele nó em T.

O processo de geração da árvore de busca, e portanto de construção da programação completa do problema, poderá seguir o procedimento de busca em grafo, descrito por Nilsson (82), página 64, onde OPEN = conjunto de atividades elegíveis (AE) e CLOSED = conjunto de atividades ordenadas (CLOSED).

O uso de uma função de avaliação para ordenar os nós candidatos (em OPEN), reduz o esforço de construção da árvore de busca, sem sacrificar a garantia de percorrer uma trajetó-

ria mais eficiente para se chegar à solução.

É interesse, no nosso caso, minimizar a combinação do custo de uma trajetória e o custo da pesquisa para se obter esta trajetória.

Desta forma, estaremos interessados em classificar as heurísticas, produzindo trajetórias de busca da solução completa, de forma a minimizar esta combinação.

Se uma combinação ponderada de custo de um procedimento heurístico 2 é menor que o de outro PH1, então o método 2 é dito usar um HEURÍSTICA MAIS PODEROSA que o método 1.

É dito que o Método 2 é mais informado que o Método 1.

A informação heurística faz com que a expansão das programações parciais se dê pelo meio mais promissor. Um método para se calcular a "promessa" do nó candidato é a definição da função avaliação.

Várias tentativas tem sido feitas para se definir, tal função, para o caso da programação de projetos com recursos limitados.

Cooper ao definir seu método amostral, no qual a escolha da candidata a incorporar-se à programação é feita pela ponderação do seu valor pela heurística, pela soma dos valores das candidatas concorrentes define um método probabilístico para a escolha da melhor trajetória. As mais frequentes são as distâncias "métricas" o nó início e o nó candidato (baseadas nos elementos do caminho crítico). As construídas sobre análise do uso de recursos oferecem informações relativas à configuração do estado do problema.

Usaremos a notação f(n), para o valor da função no nó n, programação parcial candidata a pertencer à busca da programação completa. Esta poderá ser pensada como uma programação incorporando uma única atividade, seja pela ordenação de todas as candidatas (ALGORITMO 2), seja por comparação par a par com todas as candidatas (RSM), seja pela ponderação do seu valor com os valores das demais candidatas, e então escolha de uma única, ou pela, integração de um conjunto de candidatas, seja escolhendo tantas quantas forem possíveis, da lista ordenada, ou por um processo de otimização pelo uso de recursos, quando é usado um algoritmo de programação inteira (Thesen).

Apresentamos os resultados sobre função de avaliação, como proposto por Nilsson para fundamentar nossa metodologia de categorização das heurísticas encontradas na literatura sobre procedimentos para programação de projetos com restrição de recursos.

Função de avaliação

Define-se f, função de avaliação, tal que, seu valor, f(n), para qualquer nó n estima a soma do custo da trajetória de mínimo custo do nó início s, para o nó n, mais o custo da trajetória de mínimo custo do nó n para um nó meta (uma programação completa). Ou seja, f(n) força que passe por n, a trajetória de menor custo e representa o valor deste esforço.

Notação

A função $k(n_i, n_j)$ dá o custo real da trajetória de menor custo entre dois nós arbitrários n_i e n_j (programações parciais pertencentes à mesma trajetória de busca).

A função k é indefinida para nós não tendo nenhuma trajetória entre elas. O custo de uma trajetória de custo mínimo do nó n para algum particular nó, t_i, é então, dado por $k(n, t_i)$.

Fazendo $h^*(n)$ ser o menor de todos os $k(n, t_i)$ sobre o conjunto todo de nós meta $\{t_i\}$.

Assim, $h^*(n)$ é o custo da trajetória de menor custo do nó n a um nó meta; e qualquer trajetória do nó n ao nó meta que faça n, ter o valor $h^*(n)$ é uma trajetória ótima de n para o nó meta.

O custo $k(s, n)$ de uma trajetória ótima de s para n, será notado como g^* e, então, temos:

$g^*(n) = k(s, n)$ para todo n, acessível a s.

Define-se, então, uma função f^* tal que seu valor em n, $f^*(n)$, para qualquer n é o custo real de uma trajetória ótima do nó n ao nó meta, isto é,

$f^*(n) = g^*(n) + h^*(n)$

O valor de $f^*(n)$ é então o custo de uma trajetória ótima de s, necessariamente passando pelo nó n. (Notamos que $f^*(s) = h^*(s)$ é o custo real de uma trajetória ótima, sem necessariamente passar por n, de s para o nó meta, chamada de trajetória irrestrita).

A função avaliação f será usada como uma estimativa

de f* e será dada por:

$$f(n) = g(n) + h(n)$$

onde g é uma estimativa de g* e h é uma estimativa de h*.

Uma escolha óbvia de g(n) é o custo da trajetória na árvore de busca de s para n, calculada como a soma dos custos nos arcos encontrados, enquanto o processo de pesquisa até a programação completa não termina. Assim, $g(n) \geq g^*(n)$, posto que o desenvolvimento do processo poderá definir um nó pai para n, que diminua o valor de g, se a trajetória for de custo menor.

Para a estimativa h(n), de h*(n) usamos uma informação sobre o domínio do problema.

O algoritmo de busca em grafo usando a função de avaliação

$$f(n) = g(n) + h(n)$$

estimativa de f*, será referido como ALGORITMO A.

Notamos que se usamos $g \equiv d$ (d é o RANK do nó n na árvore de busca) e $h \equiv 0$, o algoritmo de A é idêntico ao algoritmo de busca em profundidade, que ordena as programações parciais em ordem crescente de posição na árvore de busca (Nilsson, página 69). Nenhuma exploração sobre a configuração do problema é feita para a escolha das programações parciais. (Método seriado, como definido pro Hooper).

Quando A usa uma função h que é um limitante inferior para h*, notamos como algoritmo A*.

Apresentamos a seguir algumas propriedades de A*:

Resultado 1: O algoritmo de busca em GRAFO, sempre termina para um grafo finito.

PESQUISADOR	ABORDAGEM E CARACTERÍSITCAS GERAIS DO PROBLEMA	METODOLOGIA USADA	REGRAS ANALISADAS	MELHORES REGRAS E RESULTADOS
Patterson (1973)	Único projeto; multi recursos; recursos retidos Método paralelo Caso não preemptivo Critérios: atraso total e atraso ponderado Tempo de Ociosidade - Tempo Computacional	34 projetos para um período de 10 meses, no máximo 7 tipos de recursos por atividade e 13 tipos para o problema $0.75 \leq CC\ (Pascoe) \leq 0.99$ $0.26 \leq W_{jk}/R_k \leq 0.81$ $1.03 \leq W_{jk}/NAS_k \leq 2.72$	Análise fatorial sobre a combinação dos tratamentos: A: LFT, GTRD; GRRD, SIO, GRU B: tratamento B: RAN C: tratamento C: RES Utilização do "MPSP" para programação com as heurísticas	Atraso ponderado e Ociosidade estão relacionados. Atraso total Melhor regra: SIO; pior regra: GRRD Atraso ponderado: empate entre LTF e GRU. Escolha de LTF por simplificação de implementação. Recurso Ocioso: melhor regra: GRRD pior regra: SIO O uso da regra RES, combinada para desempate com as regras do grupo A, são bastante efetivas. Problema de S. Martino: GRU
Davies (1973)	Projeto único/recurso único por atividade Caso preemptivo e não preemptivo Método paralelo Critérios: OVDUR (CR/CP)	Geração de 648 redes obedecendo a: $0 \leq NDUMMY \leq 20$ NNODE = 16 $25 \leq NAS \leq 68$ $0.1 \leq CC(DAVIES) \leq 0.5$ $0.5 \leq DENSITY\text{-}F \leq 0.9$ $0.1 \leq PRES \leq 0.5$	Regressão refletindo demanda de atividade em consideração e posição relativa na rede, baseada nos valores: Σr_j; r_j; max d_{ij}; $\max(CP\text{-}EST_j)$, $\min_{k>j} \Sigma r_j$; $\max_{k>j} \Sigma r_j$	Fatores mais significativos p/ o problema: PRES >> DENS > CC. Regra mais significante Melhores regras: LTF, LST, MAX r_j, MAX(CP-EST), LFT Amplitude dos resultados: 5% do melhor para o pior.
Davis & Patterson (1975)	Projeto único - Multi recursos Redes pequenas Método paralelo Caso não preemptivo AZ/C = aumento percentual relativo à duração ótima obtida pela exaustão do método de enumeração limitada.	22 Ni 27 NPROJ = 57 83 problemas 3 tipos de recursos 1 d_{ij} 9 0.58 $UTIL_k$ 1.50 1.0 CC(Pascoe) 3.0 $\bar{x}$ DUR = 2.99	LTF, RSM, LFT, GRD, GRU, SIO, MJP, RAN Ordena atividades pela: Média do AZC*, variação, número de vezes que deu o 1º resultado, etc...	MÉDIA DE AZC* Melhores regras: LTF, RSM, LFT Piores regras: GRU, GRD, SIO, MJP (embora exceto GRU, tenham ocorrência de produzirem solução menor que as demais (inclusive ótima como é o caso de RAN e GRD)] Nenhuma regra consistentemente deu o melhor resultado geral. Estes resultados podem estender-se a grandes redes?
PATTERSON (1976)	Multi-projetos - Método Paralelo [45 parâmetros para caracterização do probelma] Critério: atraso total e atraso ponderado Projeto único: [43 parâmetros para caracterização do problema] Critério: MAXCPL (Makespan) AZCP	1. Construção das redes segundo os parâmetros de tempo, estrutura e utilização de recursos. $6 \leq Nprog \leq 10$ $20 \leq NAS_{ij} \leq 40$ 13 tipos de recursos: $\bar{X}DUR = 7.49$ PDENSITY-F = 0.82 60 problemas de multiprojetos 2. 83 problemas de Davis (75) Abordagem de projeto único Regressão múltipla para previsão da performance das heurísticas.	LTF GRD (GTRD) GRRD RSM (único projeto) SIO LFT MJP RAN	Existe relação entre estrutura do problema e a heurística usada para efeito da performance da solução. A maioria dos parâmetros relacionando demanda e disponibilidade são altamente significativos para qualquer heurística usada. A performance da heurística também é função do objetivo que se pretende. Propõe "heurística projetada" baseada nos dados de regressão para a abordagem de multi-projetos e minimização do atraso total (e secundáriamente) do atraso ponderado. Melhores regras para a abordagem multi-projetos: SIO, MJP, GRU (atraso total), GRD (atraso ponderado) Melhores regras para abordagem do projeto único: LFT, LTF, GRRD (MAXCPL), RSM (AZCP)

PESQUISADOR	ABORDAGEM E CARACTERISTICAS GERAIS DO PROBLEMA	METODOLOGIA USADA	REGRAS ANALISADAS	MELHORES REGRAS E RESULTADOS
Thesen (1976)	Abordagem de múltiprojetos Caso não preemptivo Multi recursos Utiliza problema da mochila para combinar as atividades selecionadas para programação Recursos restritos Critério: atraso	Implementado em FORTRAN para minis da Harris-Datacraft (WASP). Necessita 11.2K para problemas com 200 atividades e 10 recursos. Propõe fator urgência e usa parâmetro que lhe dá o conteúdo de informação que a rede oferece (I) Estimativa de espaço de memória (600 + (16+K) x NAS) $10 \leq NAS \leq 20$ $1 \leq d_j \leq 10$ nº de recursos: 1, 5, 10 $0 \leq UTIL_k \leq 0.5$ $I = \{0; 0.2; 0.4; 0.6; 0.8; 1.0\}$	$MAXZ = \Sigma c_j x_j \quad j \in AE$ $s.a. \Sigma w_{jk} x_j + R_k \leq R_k$; $x_j = 0$ ou 1 $c_j \rightarrow$ fator urgência relacionando utilização de recurso, folga relativa e tempo para término da programação relativa a um limite superior. Comparação entre o uso do fator urgência e as heurísticas: SIO, GRD, LFT, MINSLK, e(SLK/MAXSLK), log(SLK); fator urgência com a = 10 e b = 700	Problemas com restrições de recursos uniformemente restritos são mais difíceis de resolver que outros com poucas restrições críticas. Tempo de execução cresce linearmente com: tamanho, número de restrições, criticalidade de recursos. A taxa de aumento no tempo computacional com recursos críticos é proporcional a I. A segunda melhor performance é a MINSK (capaz de buscar solução que é 5% próxima da ótima). Fator urgência determinou o ótimo em 58% dos casos e apresentou a melhor solução para 69%. Mais eficaz computacionalemnte que o método sequencial.
Cooper (1976)	Projeto único, multi-recursos Caso não preemptivo Proposição de seleção da candidata a ser programada, por meio da ponderação do valor da heurística, pela soma do valor das demais e então ordenação. COMSOAL Critério: duração	32 projetos OS = {0.1; 0.25; 0.5; 0.75} PDENSITY-F = {0.5; 0.75} K = 5 por projeto $r_f = \{0.5; 1.0\}$ $r_{jk} = E[1, 10]$ Análise de variância	Método amostral: FCFS, LST, EST, LFT, SIO, LONG, MSAI, MST, RAN, MSA, MINA, MONA, RANK Método paralelo: as de cima e mais: MST, PROD, RED, CUMRED, PRANK, RRANK, MSAP, MISA, FSFS, LTFI, LEVEL, LFTP	As melhores regras para o método paralelo foram: LST, CUMRED, sob todas as combinações dos parâmetros. Para altos valores de OS, P-DENSITY, r_f (mais restritivos os recursos) há pouca diferença entre os resultados das regras. A melhor regra para o método amostral foi a combinação das regras: LONG, MSA, MONA, MST, RANK, pelo produto dos seus valores.
Elsayed (1982) n/n/1	Projeto único. Caso não preemptivo. Aplicação do algoritmo BAG (método paralelo). Recurso único por atividade. Rede em nós.	16 projetos já testados. $6 \leq NAS \leq 38$	ROT, ROT-ACTIM, ROT-GENRES, ACTIM, ACTRES, TIMRES	Melhores regras: GENRES, ROT/ACTIM e ROT/ACTRES
Kurtulus e Davis (1982)	Abordagem de Multiprojetos. Regras específicas a esta abordagem. Multi recursos e caso não preemptivo.	Geração de 77 projetos com as características: ARLF = {-2, -2, -1, 0, 2, 3,} (7 níveis) $0.6 \leq AUF \leq 1.6$ (11 níveis) Estatística não paramétrica. Proposta de "heurística projetada".	Regras específicas a esta abordagem: SASP, LALP, MAXSLK, MOF, MINTWK, MAXTWK, e as regras: SIO, LTF, FCFS. $34 \leq NAS \leq 63$	Sugestões de melhor regra por categorização de AUF e ARLF. Regras com atraso médio menor: SASP e MAXTWK (específicas para a abordagem). Indicando que esta abordagem pode ser melhor que a de projeto único. Comparando a performance para os dados de Patterson(76) as melhores regras: SASP, SIO, LTF, MAXTWK.
Whitehouse (1983) n/n/1	Único projeto. Único recurso por atividade	$R_k = 7$; $K = 1$	ACTIM, ACTRES, TIMBRES e GENRES, aplicados segundo o algoritmo COMSOAL.	Ordenação das regras por duração menor obtida: GENRES, TIMBRES, ACTRES, ACTIM Sugere estratégias para definição de novas regras.

ESQUISADOR	ABORDAGEM E CARACTERÍSTICAS GERAIS DO PROJETO	METODOLOGIA UTILIZADA	REGRAS ANALISADAS	MELHORES REGRAS E RESULTADOS
n (1983)	Único projeto. Único recurso por atividade. caso não preemptivo. Comparação da performance de regras sob o método paralelo e o seriado. Recursos restritos.	Categorização dos projetos numa escala de 0 a 120, a maior densidade entre 10 a 17. $12 \leq N_i \leq 180$ $5 \leq K_i \leq 15$ $0.5 \leq AUF_G \leq 0.7$	Combinação de regras, caso haja desempate: LFT, LTF, LST/LTF, EFT/LTF, EST/LTF, LTF/SIO, DUT/LST, SUF/LST, SIO/LST, MOF/LST, ROT/LST, MOT/LST, ROF/LST, MOF/LST, NI/NF.	No caso paralelo a regra de desempate não tem muita significância. A melhoria na performance das regras cresce proporcionalmente ao aumento do valor na escala, de forma uniforme. Melhores regras são: LFT e LST. Método seriado apresenta oscilação na performance das heurísticas, esperando-se melhores resultados das regras: LFT/LTF e LST/LTF
lus & Narula 985)	Abordagem de multiprojetos para multi-recursos. Caso não preemptivo. Consideração de função penalidade e diferenciação por tamanho Critério: duração	CC (Pascoe) = {1.3; 1.54; 1.74} $-3.5 \leq ALF \leq 3.0$ (8 níveis) $0.6 \leq AUF \leq 1.6$ (11 níveis) pequeno: $24 \leq NAS \leq 33$ grande : $50 \leq NAS \leq 66$ Cada problema tem 3 projetos 6 funções penalidade	Regras criadas e específicas para a abordagem: DURPEN/(LONG), MAXPEN/(FCFS), MAXTOP/(LALP), SASP1 = SASP (RAN) SASP2/(FCFS) SLKPEN/MAXSLK (GRES) e as regras: SIO/FCFS; MINTWK/FCFS MINSLK/FCFS; MAXTWK/FCFS	Penalidades iguais: SASP1, SASP2, MINSLK desiguais: MAXPEN, MAXTOP, SASP2 Penalidades iguais - pequenos: MAXTWK, SASP1 e SASP2 grandes: SASP1, MINSLK, SIO Grandes e pequenos e recursos restritivos: MINSLK Grandes e recursos críticos: SASP Pequenos e recursos críticos: MAXTWK Penalidades desiguais - pequenos: MAXPEN, MAXTWK, MAXTOP Grandes: MAXPEN, SASP2, SASP1 grandes e pequenos com recursos restritivos: MINSLK Grandes e pequenos com recursos críticos: MAXPEN Difere dos resultados de Kurtulus e Davis para AUF = 0.9 (MINSLK); AUF = 1.3 (SASP1) e AUF = 1.6 (MAXTWK)
l (1986)	Multi-projetos. Multi recursos. Maximizar valor líquido presente sob restrições de entrada e saída do fluxo de caixa e de recursos limitados.	2 grupos de projetos: G1 - $0.75 \leq AUF \leq 2.85$ G2 - $1.6 \leq AUF \leq 7.3$ (mais críticos)	3 regras baseadas em informações do fluxo de caixa e as demais: LTF, LFT, RAN	Afirmação da LTF para recursos restritos. Para problemas com maior criticalidade de recursos sugere: TS, DUAL TS = $[EFT_i - CP_i]$ combinada com DUAL = propriedade dada à atividade cujo valor marginal de atraso na programação dadà pela análise do caminho crítico seja maior.
(1987)	Experimento sobre performance dos softwares: CIPREC, ARTEMIS, PREMIS e PROJECT/2 sobre o uso das regras LST, EST, LFT e folga. (Multi recursos).	1 projeto com 15 atividades e 5 diferentes tipos de recursos.	Método paralelo e seriado. Regras LST, EST, LFT e folga.	Melhor performance PROJECT/2 processando com método paralelo e usando a heurística de menor folga. Em dois problemas atinge a solução ótima.

Resultado 2: Antes que A* termine, existe um nó n', candidato, que está na trajetória de s para o nó meta, com $f(n') \leq f^*(s)$

Resultado 3: Se existe uma trajetória s para o nó meta, A* termina

Resultado 4: O algoritmo A* é admissível (ou seja, se existe trajetória de s para o nó meta, A* termina por achar uma trajetória ótima).

Resultado 5: Para qualquer nó n, selecionada para expansão por A*, $f(n) \leq f^*(s)$

Comparação de algoritmos

A precisão da função heurística h, depende do domínio de conhecimento sobre o problema. Quando $h \equiv 0$, nenhuma informação é usada.

Para comparação de duas versões de A*, A_1 e A_2, usamos as seguintes funções avaliação:

$$f_1(n) = g_1(n) + h_1(n)$$

e

$$f_2(n) = g_2(n) + f_2(n)$$

onde h_1 e h_2 são ambos limitantes inferiores para h*.

Dizemos que A_2 é menos informado que A_1 se para qualquer nó n, que não seja meta,

$$h_2(n) > h_1(n).$$

Resultado 6: Se A_1 e A_2, são duas versões de A*, tal que A_2 é mais informado que A_1, então no processo de suas buscas de s para o nó meta, qualquer nó expandido por A_2, é também expandido por A_1. Assim, A_1 expandirá no mínimo, tantos nós quanto A_2.

Restrição monótona

No processo de busca, quando um nó n é expandido, ele pode resultar num nó em CLOSED (já gerado anteriormente) ou está em OPEN.

Então a árvore de busca deverá ajustar-se e escolher entre o nó pai existente e o recém advindo da descoberta. Este ajuste é feito pela trajetória de menor custo em G, do nó s para os descendentes do nó n.

É demonstrado que, sob certas condições sobre h, quando A* seleciona um nó para expansão, é sobre uma trajetória ótima.

Portanto, não há necessidade de testar o efeito de A* para saber se o nó gerado, já está em CLOSED, assim, não necessitando a mudança do pai, dos sucessores deste nó, na árvore de busca.

Uma função heurística, h, é dita satisfazer uma Restrição monotônica se para todo nó n_i e n_j, tal que n_j é um sucessor de n_i,

$$h(n_i) - h(n_j) \leq c(n_i, n_j)$$

com $h(t) \equiv 0$.

Escrevendo esta restrição como:

$h(n_i) \leq h(n_j) + c(n_i, n_j)$

Se a função h é mudada durante o processo de busca, esta restrição não é válida.

Resultado 7: Se A* seleciona n para expansão, e se a restrição monotônica é satisfeita,

$g(n) = g^*(n)$

Corolário: Os valores f da sequência de nós expandidos por A* são não decrescentes.

Bibliografia

_____ (1982) CIPRE C - Conversation and interactive project evaluation and control general informatization manual. IBM file nº 5370/4300 - GH 19-1150-0.

_____ (1986) SUPER PROJECT-PLUS Guia do usuário e manual de referência. Computer Associates-Micro Products Division.

ALVES, M.R.P.A. (1987) - Programação heurística: alguns princípios e aplicações. Anais do IV ENEGEP - Piracicaba - SP., p. 486-505.

ALVES, M.R.P.A. (1987) - Programação de projetos: performance de regras de priorização usadas para alocação de recursos escassos. Anais do VII ENEGEP - Niterói - RJ, p. 1101-1111.

BLAZEWICS, J. (1985) - Scheduling problems. Tutorial of the School on Combinatorial on Optimization. Rio de Janeiro, julho (xerox).

CARRUTHERS, J.A. & BATTERSBY, A. (1966) - Advances in critical path methods. Operational Research Quarterly, v. 17, nº 4.

CONWAY, R.W. (1965) - Priority dispatching and job lateness. The Journal of Industrial Engineering, v. XVI, nº 4.

COOPER, D.F. (1976) - Heuristics for scheduling resource--constrained projects: an experimental investigation management science. v. 22, nº 11.

CROWSTON, W.B. (1971) - Models for project management. Sloan Management Review, v. 12, nº 3.

DAVIES, E.M. (1973) - An experimental investigation of resource allocation in multiactivity projects. Operational Research Quarterly, v. 24, nº 4.

DAVIES, W.W. (1966) - Resource allocation in project network models - a survey. The Journal of Industrial Engineering, v. 17, nº 4.

DAVIES, E.W. & HEIDORN (1971) - An algorithm for optimal project scheduling under multiple resource constraints. Management Science, v. 17, nº 12.

DAVIES, E.W. & PATTERSON, J.H. (1985) - A comparison of heuristic and optimum solutions in resource-constrained project scheduling. Management Science, v. 21. nº 8.

DAVIES, E.W. (1973) - Project scheduling under resource constraints--historical review and categorization of procedures. AIIE Trans., v. 5, nº 4.

DAVIES, E.W. (1975) - Project network summary measures constrained - resource scheduling. AIIE Trans., v. 7. nº 2.

DAVIES, E.W. & HEIDORN, G.E. (1971) - An algorithm for optimal project scheduling under multiple resource constraints. Management Science, v. 17, nº 12.

ELMAGHRABY, S.E. & HERROELEN, W.S. (1980) - On the measurement of complexity in activity networks. E. J. Operational Research, v. 5, nº 4.

ELSAYED, E.A. (1982) - Algorithms for project scheduling with resource constraints. Int. J. Prod. Res., v. 20, nº 1.

FENDLEY, L.G. (1968) - Toward the development of a complete multiproject scheduling system. The Journal of Ind. Engin,, v. 19, nº 10.

FOULDS, L.R. (1983) - The heuristic problem-solving approach. Journal of The Operational Research Society, v. 34, nº 10.

FRAISLEBEM, F. & ALVES, M.R.P.A.(1987) - Planejamento e controle de projetos. Publicações EESC/USP nº 062/87.

GAREY, M.R.; GRAHAM, R.L. and JOHNSON, D.S. (1978) - Performance guaranteees for scheduling algorithms. Operations Research, v. 26, nº 1.

GIFFLER, B. & THOMPSON, G.L. (1960) - Algorithms for solving production-scheduling problems. Operations Research, v. 8, nº 4.

GORDON, J.H. (1983) - Heuristic methods in resource allocation. Project Management, v. 1. nº 3.

GUPTA, S,K. & TAUBE, L.R. (1985) - A state ot the art-survey of research on project management (1976-1983). Project Management: Methods and Studies. North-Holland.

HASTINGS, N.A.J. & WILLS, R.J. (1976) - Project scheduling with resource constraints using branch and bound methods. Operational Research Quarterly, v. 27, nº 2, p. 341-349.

HERROELEN, W.S. (1972) - Resource-constrained project scheduling - the state of the art. Operational Research Quarterly, v. 23, nº 3.

HOLLANDER, M. & WOLFE, D. A. (1973) - Nonparametric statistical methods. John Wiley & Sons.

HOLLOWAY, C.H.; NELSON, R.T. and SURAPHOGSCHAI, V. (1979) - Comparison of a multi-pass heuristic decomposition resource--constrained projetc scheduling procedures. Management Science, v. 25, nº 9.

JOHNSTON, D.W. (1981) - Linear scheduling for highway construction. Journal of Construction Division (ASCE), junho, p. 247-261.

KELLEY, J.E. (1961) - Critical-Path planning and scheduling: mathematical basis. Operations Research, v. 9, nº 3, p. 2961.

KURTULUS, I. & DAVIES, E.W. (1982) - Multi-project scheduling: categorization of heuristic rules performance. Management Science, v. 28, nº 2.

KURTULUS, I. & NARULA, S.C. (1985) - Multi-project scheduling: analysis of project performance. IEE Trans. v, 17, nº 1.

LAMBOURN, S. (1963) - Resource allocation and multi-project scheduling (ramps) - a new tool in planning and control. The Journal Computer, v. 5.

LEACHMAN, R.C. (1983) - Multiple resource leveling in construction systems through variation of activity intensities. Naval Research Logistics Quarterly, v. 30, nº 2.

LEACHMAN, R.C. & BOYSEN, J. (1985) - An aggregate model for multi-project resource allocation. Project management: methods and studies. North-Holland.

LEE, S.M. & PARK, O.E. (1978) - Resource planning for multiple projects. Decision Sciences, v. 9. nº 1.

LENSTRA, J.K. & RINNOOY KAN, A.H.G. (1978) - Complexity of scheduling under precedence constraints. Operations Research, v. 26, nº 1.

LEVY, F.K.; THOMPSON, G.L. and WIEST, J.D. (1962) - Multi-ship, multi-shop, workload-smoothing program. Naval Research Logistic Quarterly, v. 9, nº 1.

MODER, J.J.; PHILLIPS, C.R. and DAVIS, E.E. (1983) - Resource constraints in project scheduling. Project management with CPM, PERT and precedence diagramming. Van Nostrand Reinhold, Co.

MULLER-MERBACH, H. (1981) - Heuristic and their design: a survey. European Journal of Operational Research (Holanda). p. 1-23.

NILSSON, N.J. (1980) - Principles of artificial intelligence. Springer-Verlag Berlin Heildelberg New York.

PANWALKAR, S.A. & ISKANDER, W. (1977) - A survey of scheduling rules. Operations Research, v. 25, nº 1.

PATTERSON, J.H. (1973) - Alternate methods of project scheduling with limited resources. Naval Research Logistic Quarterly, v. 23, nº 1.

PATTERSON, J.H. (1984) - A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. Management Science, v. 30, nº 7.

PATTERSON, J.H. & HUBER, W.D. (1974) - A horizon-varyng, zero--one approach to project scheduling. Management Science, v. 20, nº 6.

PETROVIC, R. (1968) - Optimization of resource allocation in project planning. Operational Research, v. 16, nº 3, p. 559--568.

PRITSKER, A.A.B.; WATTERS, L.J. and WOLFE, P.M. (1969) - Multi--project scheduling with limited resources: a zero-one programming approach. Management Science, v. 16. nº 1.

REEVES, G.R. & SWEIGART, J. R. (1982) - Multiperiod resource allocation with variable technology. Management Science, v. 28, nº 12.

RUSSEL, R. A. (1986) - A comparison of heuristics for scheduling projects with cash flows and resource restrictions. Management Science, v. 32, nº 10.

SARIN, S.C. (1982) - Scheduling independent projects against a single resource. Int. J. Prod. Res., v. 20, nº 2.

SELINGER, S. (1980) - Construction planning for linear program. Journal of Construction Division (ASCE), p. 195-205.

SLOWINSKI, R. (1981) - Multiobjective network scheduling with efficiente use of renewable and nonrenewable resources. European Journal of Operational Research, v. 7, nº 3.

SLOWINSKI, R. - Two approaches to problems of resource allocation among project activities - a comparative study. Journal of the Operational Research Society, v. 31, nº 8.

SUAREZ, L.F. (1987) - Resource allocation: a comparative study. Project Management Journal, v. XVIII, nº 1.

TALBOT, F.B. (1982) - Resource-constrained project scheduling with time-resource tradeoffs: the nonpreemptive case. Management Science, v. 28, nº 10.

THESEN, A. (1978) - Heuristic project scheduling. Project Management Quarterly, v. IX, nº 1.

THESEN, A. (1976) - Heuristic scheduling of activities under resource and precedence restrictions. Management Science, v. 23, nº 4.

WEGLARZ, J. (1981) - Project scheduling with constinuously--divisible, doubly constrained resources. Management Science, v. 27, nº 9.

WHITEHOUSE, G.E. (1983) - A comparison of computer search heuristics to analyse activity/networks with limited resources. Project Management Quarterly, p. 35-39 (junho).

WIEST, J.D. (1964) - Some properties of schedules for large project with limited resources. Operations Research, v. 12,nº 3.

WIEST, J.D. (1967) - A heuristic model for scheduling large projects with limited resources. Management Science, v. 13,nº 6.

WILD, R. (1970) - Concepts for operation management. John Wiley & Sons.

WILLIS, R.J. & HASTINGS, N.A.J. (1976) - Project scheduling with resource constraints using branch and bound methods. Operational Research Quarterly, v. 72, nº 2.

WOODWORTH, B.M. & WILLIEN, C.J. (1975) - A heuristic algorithm for resource leveling in multi-project, multi-resource scheduling. Decision Sciences, v. 6, nº 3.

ZANAKIS, S. & EVANS, J. (1981) - Heuristic optimization: why, whem, and how to use it. Interfaces (US), v. 11, p. 84-90.

Softwares Representativos

MPM — Linguagens fonte: Fortran, Cobol. Redes com 8000 atividades em linhas. Multi-projetos. Número ilimitado de recursos por atividade. Previsão de recursos. Disponível para sistema MARIII-GE.

MSCS — Linguagens fonte: Cobol, Fortran. Redes em linha ou Diagramas de Precedência: 42600 atividades. Alocação de recursos limitados com seleção da heuristica, feita pelo usuário. IBM 360/370.

PACII — Linguagens fonte: ANS Cobol. Redes nos nós ou em linhas. Número de atividades limitada pela capacidade da máquina. Programação de equipes de recursos. Multi-projetos, fazendo análise individual para cada projeto. Interativo. Qualquer mini-computador com compilador Cobol.

PMSC — Linguagem fonte: Fortran. Redes em linha ou Precedência. Número ilimitado de atividades. Nivelamento e Programação de Recursos Limitados de forma interativa. Grande porte.

ARTEMIS — Linguagem fonte: redes em linha ou Diagramas de Precedência. 32 mil atividades. Usa base de dados relacional para manusear dados de rede, formatos de relatórios e conjunto de dados criados pelo usuário.

Programação de Recursos Limitados é feito pelo método seriado. Opções disponíveis: definição de prioridade às atividades; caso preemptivo; uso de níveis alternativos de recurso e nivelamento de recursos . Resultados são acessíveis para criação de modelos de relatórios, os quais podem ser armazenados e/ou usados continuamente. HP100 mini com 20 Mbyte.

CIPREC Linguagens fonte: PL/1 e Assembler . Desenvolvido do PROJACS e PMS II. Trabalha com dois sistemas: processador central e função expandida. O último é opcional e contém: Programação de Recursos; Análise custo por atividade e pelo conteúdo de trabalho. Redes em linha ou precedência. Alocação de Recursos é feita pela execução de análise de simulação utilizando o método paralelo. Opções de heurística bastante extensa: EST, EFT, LST, LFT, SIO, LTF, NOI , NOF, SUSI, prioridade da atividade e de conjunto de atividades, número de prioridade do código organizacional. IBM 360/370.

PREMIS Linguagem fonte: Assembler. Redes em linha ou Diagrama de Precedência. 64000 atividades. Sem limite no número de tipo de recursos por atividade e o número de recursos por projeto é associado a capacidade da máquina. Programação de Recursos Limitados pelo método seriado. Opções disponíveis: definição de prioridade para atividades, caso preemptivo e nive-

lamento de recursos. Admite criação de relatórios, além do uso de relatório padrão. IBM 360.

PROJECT/2 Linguagem fonte: Icetran. Redes em linha ou Diagramas de Precedência. Programação de Recursos pode ser feita pelos métodos seriado ou paralelo. Opções disponíveis: definição de prioridade às atividades, caso preemptivo, nivelamento de recursos e níveis alternativos para limite de recursos. Tem oito processadores: Rede e CPM; Definidor de objetivos; Alocação de custo e recursos; Recurso limitado; multiprojetos; Interativo. Admite criação de redes pelo terminal. IBM OS/VS.

A N E X O 1

Relatório da pesquisa desenvolvida para apreciação e pronunciamento sobre a fase probatório no RDIDP (20.09.84-20.09.87):

"Caracterização de Procedimentos Heurísticos em Planejamento e Programação de Projetos".