UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CURSO DE ENGENHARIA MECÂNICA

JEAN GABRIEL SANTOS

CÁLCULOS DE TEMPOS DE PRODUÇÃO EM LINHAS *FLOW SHOP*

TRABALHO DE CONCLUSÃO DE CURSO

Cornélio Procópio – PR

2014

JEAN GABRIEL SANTOS

CÁLCULOS DE TEMPOS DE PRODUÇÃO EM LINHAS *FLOW SHOP*

TRABALHO DE CONCLUSÃO DE CURSO

Trabalho de Conclusão de Curso apresentada como requisito parcial à obtenção do título de Bacharel em Engenharia Mecânica da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Me. Mauricio Iwama Takano.

CORNÉLIO PROCÓPIO

2014

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Coordenação de Engenharia Mecânica – COEME
Coordenação de Tec. Manutenção Industrial - COMIN

TERMO DE APROVAÇÃO

Trabalho de Conclusão de Curso

CÁLCULOS DE TEMPOS DE PRODUÇÃO EM LINHAS *FLOW SHOP*.

por

Jean Gabriel Santos

Este trabalho de conclusão de curso foi apresentado às 10h20min do dia **03 de março 2015** como requisito parcial para a obtenção do diploma de graduação no curso de BACHARELADO EM ENGENHARIA MECÂNICA, Linha de Pesquisa – Tecnologia e Desenvolvimento, Departamento Acadêmico de Mecânica, Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

(aprovado, aprovado com restrições, ou reprovado)

Prof. Me. Fernando Henrique De Oliveira Câmara
(UTFPR)

Prof. Dr. Adailton Silva Borges
(UTFPR)

Prof. Me. Mauricio Iwama Takano
(UTFPR)
Orientador

RESUMO

SANTOS, Jean. Cálculos de tempos de produção em linhas flow shop. 2014, 58 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Mecânica) – Departamento acadêmico de mecânica – Universidade Tecnológico Federal do Paraná. Cornélio Procópio, 2014.

O cálculo para encontrar a data de término do processamento em cada máquina, para todas as tarefas, é algo que pode se tornar complexo dependendo das restrições tecnológicas do problema. Por este motivo o presente trabalho tem como objetivo desenvolver equações capazes de encontrar a data de término de processamento de cada tarefa em cada máquina em ambientes *flow shop* considerando diferentes restrições tecnológicas. São apresentadas no trabalho todas as restrições tecnológicas que podem ocorrer em ambientes *flow shop*. Todas as equações foram desenvolvidas considerando apenas a ocorrência de uma única restrição por vez, com exceção da restrição de data de lançamento, que foi considerada ocorrendo junto a tempo de *setup* da tarefa e da máquina, uma vez que a restrição de data de lançamento não ocorre sozinha. As equações são, então, validadas utilizando uma base de problemas conhecidos. Como resultado dos estudos, são apresentados todos os cálculos recursivos dos problemas analisados. Os resultados obtidos de um exemplo selecionado, podem mostrar o funcionamento correto das equações, uma vez que o exemplo comtemplou todas equações desenvolvidas, além dos resultados poderem ser utilizados futuramente para estudo de métodos de otimização de diferentes problemas envolvendo ambiente *flow shop*.

Palavras chave: Cálculo recursivo; *Flow Shop*; Sequenciamento.

ABSTRACT

SANTOS, Jean. Calculations time production in flow shop lines. 2014, 58 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Mecânica) – Departamento acadêmico de mecânica – Universidade Tecnológico Federal do Paraná. Cornélio Procópio, 2014.

The calculation to find the completion time of all jobs on each machine, is something that may become complex depending on the technological restraints. of the problem. Therefore, the present work has the objective to develop equations which are able to find the completion time of each job on each machine in flow shop problems considering different technological restraints All constrains that are applicable in a flow shop environment are presented and studied, The equations were developed considering only the occurrence of a single constraint at a time, except for the restriction of the release date, which was regarded as occurring along the setup time of the task and the machine. The equations are then validated using a well known problem base. As a result of the studies all recursive calculus for the studied problems are presented. The result of this work can be used to study optimization methods in flow shop environment.

Keywords: Recursive calculation; Flow shop; Sequencing.

Lista de Figuras

Sumário

1. INTRODUÇÃO

Nas últimas décadas, a globalização tem se fortalecido, devido a uma interligação mais fácil e rápida pelo mundo e juntamente um crescimento de consumo. Visando este crescimento, surgiram empresas buscando uma parcela deste mercado, gerando assim uma concorrência entre elas, pois atualmente o consumidor busca melhor relação custo-benefício nos produtos, devido à maior oferta de produtos similares (TUBINO, 2007).

Nesse contexto criou-se um departamento para gerenciar a produção, buscando otimizar o processo para diminuir o custo da produção, surgindo assim o setor de planejamento e controle da produção (PCP) que visa o gerenciamento desde a entrada de matéria-prima até a sua transformação final em produto acabado, facilitando as tomadas de decisões. Esse gerenciamento é feito por meio de um planejamento de partes fundamentais da produção, como por exemplo: quantidade de insumos a comprar; quantidade de produtos a ser produzida; mão-de-obra humana empregada; máquinas a serem utilizadas; as fases do processo; a definição do ambiente da fábrica, que é importante para um fluxo melhor na produção; e o próprio controle da produção baseado neste planejamento, para garantir a produção dentro da expectativa e mantendo a qualidade.

Algumas etapas desenvolvidas no PCP são (TUBINO, 2007):

- Planejamento Estratégico: É um planejamento de longo prazo, que auxilia no gerenciamento dos recursos necessários, com a finalidade de atender a previsão de demanda, diminuindo ou aumentando a produção, de acordo com capacidade da empresa.
- Planejamento Tático: É o planejamento de médio prazo, onde buscam encontrar de que maneira a empresa deve produzir, para se obter a melhor eficiência de produção em relação a demanda, já que nesta atividade, não se considera apenas a previsão de demanda, mas também pedidos de compra e venda já firmados.
- Planejamento Operacional: Define como ocorrerá a produção na fábrica em seu dia-a-dia, de acordo com algumas atividades. Sendo a última etapa do PCP, ela tem como finalidade o acompanhamento e verificação da produção, assim constatar se o que foi planejado nas etapas anteriores

está realmente ocorrendo na linha de produção. O levantamento destes dados, pode auxiliar em decisões posteriores na fábrica.

- Administração de materiais: Cuida da logística dos materiais, como estoques, tamanho dos lotes e reposição de matéria-prima.
- Sequenciamento: Visa a otimização do processo, buscando encontrar o melhor sequenciamento de operação nas máquinas.
- Emissão de ordens: É a parte de documentação para a liberação da produção.

Pode-se, dar um destaque ao sequenciamento, que muitas vezes com pequenas modificações, consegue gerar um ganho relativamente grande na produção, proporcional à quantidade de produção e principalmente ao fluxo contínuo. Assim também, um melhor aproveitamento da sequência se faz necessário, isso porque, em uma linha de produção, sempre que há um ganho ou uma perca em determinado momento, devido ao fluxo contínuo, essa perca ou ganho é cumulativo, gerando assim, maior lucro ou gasto no processo.

1.1 OBJETIVOS

1.1.1 Objetivo geral

Elaborar programas capazes de calcular a data de término de *setup* e processamento das tarefas em cada uma das máquinas, em ambientes *flow shop*, para as diferentes restrições tecnológicas, que possam ocorrer neste ambiente. Também elaborar uma apostila didática, onde mostra-se os detalhes de como desenvolver cada equação para as restrições tecnológicas presentes no ambiente em estudo.

1.1.2 Objetivos específicos

- Desenvolver o cálculo recursivo das datas de término do processamento nas diversas restrições tecnológicas que podem ocorrer no ambiente *flow shop;*
- Programar as equações utilizando o *software MatLab*;

- Testar e validar os programas.
- Desenvolver uma apostila didática, onde é detalhado o desenvolvimento das equações.

1.2 JUSTIFICATIVA

A competitividade nas empresas faz com que todo processo de melhoria seja justificável. Os métodos computacionais tradicionais (pesquisa operacional) podem demandar muito tempo para serem implementados e chegar a uma solução para os problemas. As equações desenvolvidas serão implementadas em software, assim, tanto a sequência das tarefas como as restrições tecnológicas podem ser alteradas facilmente e testadas previamente no software, fornecendo dados importantes que auxiliarão na otimização da linha de produção sem a necessidade de métodos computacionais.

Estes programas possibilitam que em um ambiente *flow sho*p se possa calcular as datas de conclusão de cada tarefa em cada máquina, antecipadamente, tornando-se uma informação de grande valor.

Os programas desenvolvidos podem ainda servir de base para o desenvolvimento de métodos computacionais de otimização de problemas de sequenciamento da produção em ambientes *flow shop*.

2. SEQUENCIAMENTO

Como em um ambiente, é possível ter diversas máquinas e tarefas, para facilitar o entendimento são utilizadas notações. Seguem algumas das notações utilizadas em problemas de sequenciamento, considerando: *j* se refere a uma tarefa, podendo variar até *n*; *k* representa uma máquina, que varia até *m*; *c* representa uma estação de trabalho, em ambientes de máquinas em paralelo; *g* é a quantidade de máquinas em cada estação de trabalho (PINEDO, 2008).

- **Tempo de processamento (p_{jk}):** representa o tempo de processamento da tarefa *j* na máquina *k*;
- **Data de lançamento (r_j):** refere-se a data que a tarefa *j* pode começar seu processamento;
- **Prazo de conclusão (d_j):** Quando uma data de conclusão (data de entrega para o cliente) da tarefa *j* deve ser atendida dentro de um prazo, é denotada por d_j;
- **Peso (w_j):** É um fator de prioridade, onde é denotada a importância de uma tarefa *j* em relação às outras;
- **C_{jk}:** Indica a data em que a tarefa *j* termina o processamento na máquina *k*;
- **C_{max}:** Se refere à data em que a última tarefa da sequência deixa a última máquina, ou seja, o tempo de trabalho;
- ***Lateness* (L_j):** Esta notação pode assumir valores positivos, igual a zero ou negativos. Sendo o valor positivo quando a tarefa *j* é concluída após o prazo de entrega, zero quando a tarefa *j* conclui seu processamento na data programada e negativo quando a tarefa *j* conclui seu processamento antes do prazo de entrega;
- ***Tardiness* (T_j):** Assume valores positivos ou igual a zero. O que ocorre é que enquanto a data de término da tarefa *j* não ultrapassar o seu prazo de conclusão, o valor é constante igual a zero, após esta data, então, *Tj* passa a assumir valores positivos;
- **Penalidade unitária (U_j):** Pode assumir valor zero ou um. Assume valor zero se concluir o processamento da tarefa *j* até o prazo de conclusão. Caso haja atraso, independentemente do tempo, assume valor um.

No gráfico 1 é apresentada a diferença entre as varáveis *lateness* (L_j), *tardiness* (T_j) e penalidade unitária (U_j), com relação ao prazo de conclusão (d_j) das tarefas e à data de conclusão da tarefa (*Cj*).

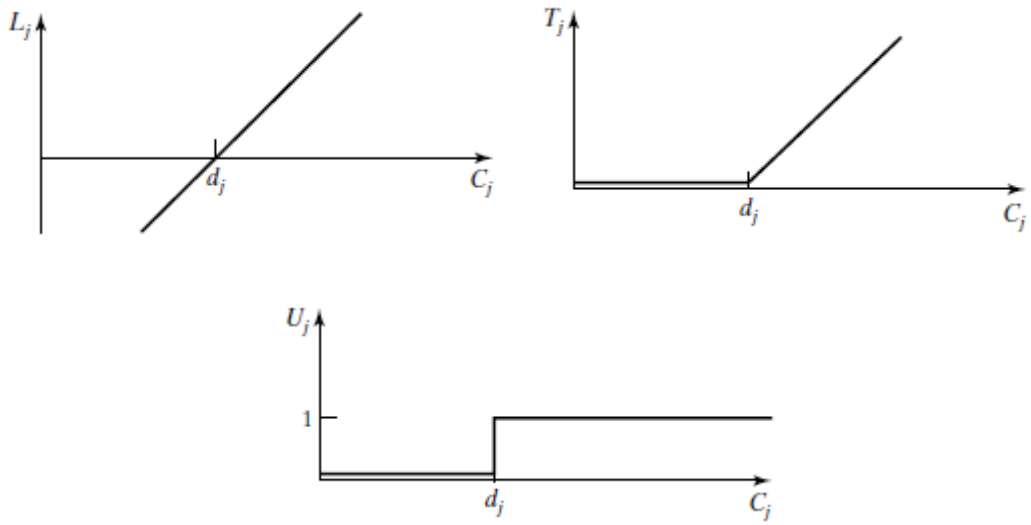

Figura 1 - Gráficos que relacionam os tipos de atrasos com a data de conclusão.

Fonte: Adaptado de Pinedo (2008).

Os problemas de sequenciamento da produção podem ser classificados por três identificadores (GRAHAM *et al.*, 1979): **α** (*Alpha*); **β** (*Beta*); **ɣ** (*Gama*). Sendo no campo α descrito o ambiente de trabalho e este campo possui apenas uma entrada. No campo β que pode ter nenhuma, uma ou várias entradas contém informações sobre as restrições tecnológicas do problema. Já no campo ɣ são definidos os critérios de avaliação do problema, podendo ter uma ou mais entradas.

2.1 Campo α

Para o campo α, pode-se ter os seguintes ambientes de trabalho (PINEDO, 2008):

- **Máquina única (*1*):** Trata-se de um ambiente com uma única máquina;
- **Máquinas idênticas em paralelo (P_m):** É um modo de trabalho com máquinas idênticas em paralelo, onde a tarefa necessita de apenas uma operação e pode ser processada por qualquer uma das máquinas;

- **Máquinas com diferentes velocidades em paralelo (Q_m):** Neste ambiente, tem-se máquinas trabalhando em paralelo, porém com diferentes velocidades, caso a velocidade de trabalho delas sejam iguais, pode-se dizer que é um ambiente de máquinas idênticas em paralelo;
- **Máquinas independentes em paralelo (R_m):** É um ambiente de máquinas em paralelo, que trabalham independentes, pois são máquinas diferentes umas das outras, então fazem diferentes trabalho com diferentes velocidades;
- ***Flow shop* (F_m):** Este tipo de ambiente, trabalha com máquinas em série, onde todas as tarefas devem seguir a sequência das máquinas e ser processadas em cada uma delas;
- ***Flexible flow shop* (FF_c):** É uma variação do ambiente *flow shop*, onde cada uma das *k* máquinas é substituída por *c* estações de trabalho. Em cada estação de trabalho existem *g* máquinas idênticas em paralelo;
- **Job *shop* (J_m):** Neste ambiente operacional, cada tarefa tem seu próprio fluxo determinado;
- ***Flexible job shop* (FJ_c):** É uma variação do *job shop*, como no *flexible flow shop* também tem as *k* máquinas substituídas por *c* estações de trabalho com *g* máquinas idênticas em paralelo nas estações;
- ***Open shop* (O_m):** Neste ambiente, não há definição de fluxo para as tarefas.

Na figura 2 são apresentadas as relações existentes entre os ambientes:

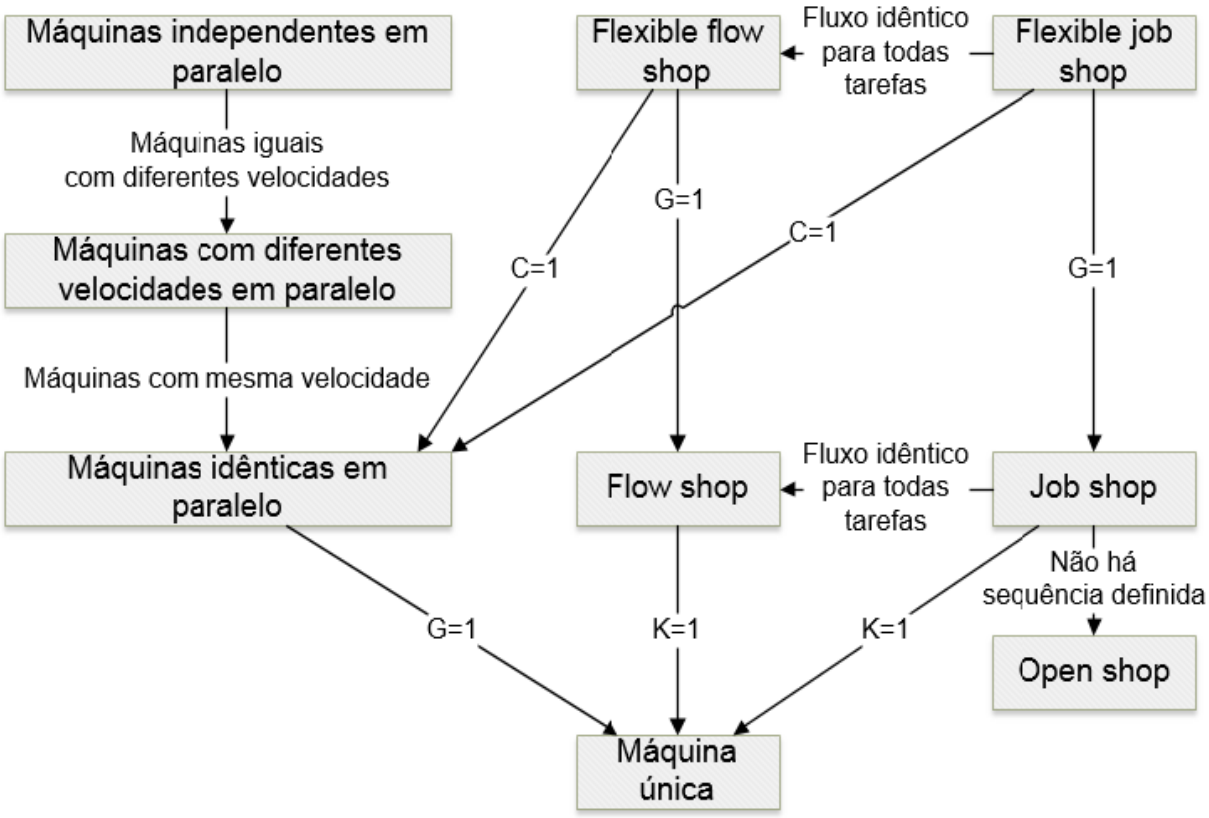

Figura 2 - As relações existentes entre os ambientes do campo α.

Fonte: Adaptado de Maccarthy e Liu (1993).

2.2 Campo β

No campo β, pode-se ter as seguintes restrições (PINEDO, 2008):

- **Datas de lançamento (r_j):** Esta restrição implica que uma tarefa *j* só poderá ser iniciada após uma data programada de lançamento;
- **Preempção (*prmp*):** Neste tipo de entrada, ao ser iniciada a tarefa em uma máquina, ela não tem que necessariamente ser concluída, ou seja, é permitido ao programador, interromper o processamento e colocar uma outra tarefa que tenha preferência;
- **Restrições de precedência (*prec*)**: Aplicável em ambientes de máquina única ou de máquinas em paralelo, esta restrição define que uma ou mais tarefas na máquina devem ser concluídas antes que outra tenha autorização para começar;

- **Tempo de *setup* dependente da sequência e da máquina (S_{ij}):** O que ocorre nesta restrição, é que o tempo de *setup* da máquina, vai depender da sequência de produção que foi definida. Caso o tempo de *setup* dependa também da máquina, a restrição é simbolizada por S_{ijk};
- **Famílias de trabalho (*fmls*):** O trabalho de uma mesma família pode ser feito na mesma máquina, com diferentes tempos de processamentos, porém sem necessidade de *setup* da máquina. O setup só será necessário quando houver a troca de famílias;
- **Processamento em lote (*batch*(*b*)):** É possível quando a máquina é capaz de processar um número de trabalhos simultaneamente, assim, o tempo de conclusão do lote é determinado pelo maior tempo de processamento;
- **Quebras (*brkdwn*):** Isto implica que a máquina pode não estar sempre disponível. Isso pode ocorrer devido a manutenção na máquina, mudanças no programa, entre outras paradas que podem ocorrer;
- **Restrições da qualificação da máquina (M_j):** Ocorre somente em ambiente de máquinas em paralelo, implicando que nem todas as máquinas em paralelo podem realizar o trabalho *j*. M_j refere-se ás máquinas que podem realizar a tarefa;
- **Permutação (*prmu*):** É uma restrição que pode ocorrer em ambientes *flow shop*. É conhecida como *First In First Out* (*FIFO*), ou seja, o primeiro que entra é o primeiro que sai, assim, a ordem das tarefas da primeira máquina, é mantida até o fim do processo;
- **Bloqueio (*block*):** Esta restrição implica que a tarefa, mesmo depois de concluída, deve permanecer na máquina até que a tarefa que se encontra na máquina seguinte tenha sido concluída, isto para não sobrepor o trabalho. Assim, o Bloqueio implica em *FIFO*;
- **Sem espera (*nwt*):** Não há espera entre as máquinas. Desse modo, a tarefa na primeira máquina deve ser adiada para garantir que não haja espera em nenhuma máquina. Também resulta em *FIFO*;
- **Recirculação (*rcrc*):** Quando esta restrição está presente define que uma tarefa pode utilizar uma mesma máquina mais de uma vez. Só ocorre em *Job Shop* ou *Flexible Job Shop*.

2.3 Campo ɣ

Dos critérios de avaliação do problema que podem ser utilizados no campo ɣ, tem-se (PINEDO, 2008):

- ***Makespan* (C_{max}):** É o tempo de conclusão do último trabalho a deixar o sistema. Geralmente, um tempo mínimo, indica uma boa utilização da máquina;
- ***Maximum lateness* (L_{max}):** Considera o maior atraso das datas de vencimento;
- ***Total weighted completion time* ($\sum w_j c_j$):** É a soma ponderada dos tempos de conclusão dos trabalhos. Esse tempo da uma indicação do total de retenção ou de custos ocorridos pela programação;
- ***Discounted total weighted completion time* ($\sum w_j(1-e^{-rC_j})$):** Neste critério, tem-se que, se a tarefa *j* não é concluída no tempo *t*, então é adicionado um custo $w_j r e^{-rt} dt$ a este atraso;
- ***Total weighted tardiness* ($\sum w_j T_j$):** Considera o tempo ponderado de atraso. É também uma função de custo mais geral que o *total weighted completion time*;
- ***Weighted number of tardy jobs* ($\sum w_j U_j$):** Informa o número ponderado de trabalhos que estão atrasados.

3. DESENVOLVIMENTO DAS EQUAÇÕES

Um ambiente *flow* shop pode apresentar as seguintes restrições tecnológicas: Permutação; Tempo de *setup* dependente da sequência; Tempo de *setup* dependente da sequência e da máquina; Sem espera; Família de trabalho; Data de lançamento; e Bloqueio. A seguir são apresentados os cálculos recursivos para as datas de término de processamento das tarefas para cada uma dessas restrições tecnológicas que podem ocorrer em ambientes *flow shop*.

3.1 Permutação

$$C_{[1]1} = P_{[1]1}, \tag{1}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k-1)}, \quad k = 2, \dots, m. \tag{2}$$

$$C_{[j]1} = P_{[j]1} + C_{[(j-1)]1}, \quad j = 2, \dots, n. \tag{3}$$

$$C_{[j]k} = P_{[j]k} + \max(C_{[(j-1)]k}, C_{[j](k-1)}), \quad k = 2, \dots, m, \quad j = 2, \dots, n. \tag{4}$$

As equações (1) e (2) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas, enquanto a equação (3) é para encontrar a data do término de todas as tarefas na primeira máquina, assim a equação (4) é utilizada para calcular a data de término das demais tarefas.

3.2 Tempo de *setup*

$$C_{[1]1} = P_{[1]1}, \tag{5}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k-1)}, \quad k = 2, \dots, m. \tag{6}$$

$$R_{[j]1} = C_{[(j-1)]1} + S_{[(j-1)][j]}, \quad j = 2, \dots, n. \tag{7}$$

$$C_{[j]1} = P_{[j]1} + R_{[j]1}, \quad j = 2, \dots, n. \tag{8}$$

$$R_{[j]k} = \max((C_{[(j-1)]k} + S_{[(j-1)][j]}), C_{[j](k-1)}), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{9}$$

$$C_{[j]k} = P_{[j]k} + R_{[j]k}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{10}$$

As equações (5) e (6) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas, assim como as equações (7) e (8) são para determinar a data de término de todas as tarefas na primeira máquina, onde a equação (7) é a soma da data de término da tarefa anterior na máquina um com o tempo de *setup* devido a mudança de tarefa e a equação (8) considera a adição do tempo de processamento na equação (7). A equação (9) compara a maior data entre a tarefa anterior mais o tempo de *setup* e o término da tarefa atual na máquina anterior, utilizando esse resultado na equação (10) para encontrar as datas de término do processamento das tarefas nas máquinas dois em diante.

3.3 Tempo de *setup* máquina

$$C_{[1]1} = P_{[1]1}, \tag{11}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k-1)}, \quad k = 2, \dots, m. \tag{12}$$

$$R_{[j]1} = C_{[(j-1)]1} + S_{[(j-1)][j]1}, \quad j = 2, \dots, n. \tag{13}$$

$$C_{[j]1} = P_{[j]1} + R_{[j]1}, \quad j = 2, \dots, n. \tag{14}$$

$$R_{[j]k} = \max\left(\left(C_{[(j-1)]k} + S_{[(j-1)][j]k}\right), C_{[j](k-1)}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{15}$$

$$C_{[j]k} = P_{[j]k} + R_{[j]k}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{16}$$

As equações (11) e (12) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas, assim como as equações (13) e (14) são para determinar a data de término de todas as tarefas na primeira máquina, onde a equação (13) é a soma da data de término da tarefa anterior na máquina um com o tempo de *setup* devido a mudança de tarefa na mesma máquina, já a equação (14) considera a adição do tempo de processamento na equação (13). A equação (15) compara a maior data entre a tarefa anterior mais o tempo de *setup*

dependendo da máquina e o término da tarefa atual na máquina anterior, utilizando o maior resultado na equação (16), para encontrar as datas de término do processamento das tarefas nas máquinas restantes.

3.4 Sem espera

As equações para a restrição sem espera, foram desenvolvidas com base em um artigo (QIAN et al., 2009).

$$C_{[1]1} = P_{[1]1}, \tag{17}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k\text{-}1)}, \quad k = 2, \dots, m. \tag{18}$$

$$R_{[j-1](k-1)} = R_{[(j-1)](k-1)} + \left(P_{[(j-1)]k} - P_{[j](k-1)}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{19}$$

$$E_{[j-1]1} = \max\left(E_{[j-1]1}, R_{[j-1](k-1)}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{20}$$

$$C_{[j]k} = \left(E_{[j-1]1} + P_{[(j-1)]1}\right) + C_{[1](k-1)}, \quad j = 2, \dots, n. \tag{21}$$

As equações (17) e (18) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas. A equação (19) ocorre em paralelo com a (20), uma vez que cada valor encontrado na equação (19), já é testado na (20), identificando assim, o maior intervalo de tempo, ou seja, a data em que a próxima tarefa pode ser iniciada. A equação (21), é utilizada cada vez que o *k* das equações (19) e (20) chega até *m*, encontrando assim a data de término do processamento da tarefa nas máquinas restantes. Após isso, o ciclo se reinicia na equação (19), até que o *j* chegue a *n*.

3.5 Família de trabalho

$$C_{[1]1} = P_{[1]1}, \tag{22}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k\text{-}1)}, \quad k = 2, \dots, m. \tag{23}$$

$$SE\left(Fa_{[(j-1)]1} = Fa_{[j]1}\right), \quad j = 2, \dots, n. \tag{24}$$

$$C_{[j]1} = C_{[(j-1)1} + P_{[j]1}, \quad j = 2, \dots, n. \tag{25}$$

$$SE\ N\tilde{A}O\left(C_{[j]1} = C_{[(j-1)1} + P_{[j]1} + S_{[(j-1)][j]}\right), \quad j = 2, \dots, n. \tag{26}$$

$$SE\left(Fa_{[(j-1)]k} = Fa_{[j]k}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{27}$$

$$C_{[j]1} = C_{[(j-1)k} + P_{[j]k}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{28}$$

$$SE\ N\tilde{A}O\left(C_{[j]k} = max\left(C_{[j](k-1)}, C_{[(j-1)]k}\right) + P_{[j]k}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{29}$$

As equações (22) e (23) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas. A equação (24), faz um comparativo, para identificar se a tarefa a ser iniciada pertence à mesma família da tarefa a frente. Caso sejam da mesma família, então é executada a equação (25), onde encontra-se a data de processamento da tarefa na primeira máquina. Caso elas não pertençam à mesma família, é executada a equação (26), também utilizada para encontrar a data de processamento da tarefa na primeira máquina, porém, levando em consideração o tempo de *setup* devido a troca de família. Tendo as últimas equações (27; 28; 29), finalidade de encontrar a data de termino de processamento das tarefas nas máquinas restantes, seguindo a mesma lógica das equações (24), (25) e (26), porém considerando a data de término da tarefa na máquina anterior.

3.6 Data lançamento

$$C_{[1]1} = P_{[1]1} + S_{[1][1]1}, \tag{30}$$

$$C_{[1]k} = \max\left(\left(C_{[1](k-1)} + P_{[1]k}\right), \left(S_{[1][1]k} + P_{[1]k}\right)\right), \quad k = 2, \dots, m. \tag{31}$$

$$R_{[j]1} = \max\left(\left(C_{[(j-1)]1} + S_{[(j-1)][j]1}\right), S_{[j][j]1}\right), \quad j = 2, \dots, n, \tag{32}$$

$$C_{[j]1} = P_{[j]1} + R_{[j]1}, \quad j = 2, \dots, n. \tag{33}$$

$$R_{[j]k} = \max\left(\left(C_{[(j-1)]k} + S_{[(j-1)][j]k}\right), C_{[j](k-1)}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{34}$$

$$R_{[j]k} = \max\left(S_{[j][j]k}, R_{[j]k}\right), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{35}$$

$$C_{[j]k} = P_{[j]k} + R_{[j]k}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{36}$$

As equações (30) e (31) são utilizadas para calcular a data de término do processamento da primeira tarefa em todas as máquinas. Porém, na equação (30) é considerado a soma da data em que a tarefa pode ser iniciada, assim como na (31), leva em consideração, a data mais longa de início (a tarefa na máquina anterior, ou a data em que ela pode ser iniciada na máquina atual). Para encontrar a data de término de todas as tarefas na primeira máquina, são utilizadas as equações (32) e (33), onde na equação (32), determina-se a maior data (término da tarefa anterior na máquina, mais o *setup* para a tarefa atual, ou a data de lançamento da tarefa na máquina), para que possa ser somada na equação (33), com a data de processamento da tarefa na máquina.

As equações (34), (35) e (36) possuem a mesma finalidade para a determinação da data de término das tarefas nas máquinas restantes, porém, a equação (37), é adicionada para criar um comparativo com a data de término da tarefa atual na máquina anterior, já que a partir da máquina dois passa a ser um fator decisório.

3.7 Bloqueio B

$$C_{[1]1} = P_{[1]1}, \tag{37}$$

$$C_{[1]k} = P_{[1]k} + C_{[1](k\text{-}1)}, \quad k = 2, \dots, m. \tag{38}$$

$$SE\ (L_{[1](k-1)} \neq 0\ \&\ (C_{[(j-1)](k-1)} + P_{[j](k-1)}) > C_{[(j-T_{[1](k-1)})]k}), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{39}$$

$$L_{[1](k-1)} = L_{[1](k-1)} - 1, \quad k = 2, \dots, m. \tag{40}$$

$$SE\ (k-1) = 1 \tag{41}$$

$$SE(L_{[1](k-1)} < B_{(k-1)}), \quad k = 2, \dots, m. \tag{42}$$

$$C_{[j](k-1)} = C_{[(j-1)](k-1)} + P_{[j](k-1)}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{43}$$

$$SE\ NÃO(C_{[j](k-1)} = \max(C_{[j-(L_{[1](k-1)}+1)]k}, C_{[(j-1)](k-1)} + P_{[j](k-1)}) \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{44}$$

$$SE\ NÃO(SE(L_{[1](k-1)} < B_{(k-1)})), \quad k = 2, \dots, m. \tag{45}$$

$$R_{[j](k-1)} = \max\Big(\big(C_{[(j-1)](k-1)} + P_{[j](k-1)}\big), \big(C_{[j](k-2)} + P_{[j](k-1)}\big)\Big), \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{46}$$

$$C_{[j](k-1)} = R_{[j](k-1)}, \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{47}$$

$$SE\ NÃO(R_{[j](k-1)} = \max(\big(C_{[(j-1)](k-1)} + P_{[j](k-1)}\big), \big(C_{[j](k-2)} + P_{[j](k-1)}\big)) \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{48}$$

$$C_{[j](k-1)} = \max(C_{[j-(L_{[1](k-1)}+1)]k}, R_{[j](k-1)}) \quad j = 2, \dots, n, \quad k = 2, \dots, m. \tag{49}$$

$$SE(C_{[j](k-1)} < C_{[j-(L_{[1](k-1)}+1)]k}, \quad k = 2, \dots, m. \tag{50}$$

$$L_{[1](k-1)} = L_{[1](k-1)} + 1, \quad k = 2, \dots, m. \tag{51}$$

$$SE(T_{[1](k-1)} \leq B_{(k-1)}), \quad k = 2, \dots, m. \tag{52}$$

$$T_{[1](k-1)} = T_{[1](k-1)} + 1, \quad k = 2, \dots, m. \tag{53}$$

$$C_{[j]k} = \max\big(C_{[(j-1)]k}, C_{[j](k-1)}\big) + P_{[j]k}, \quad k = m. \tag{54}$$

A equação (37), informa a data de término do processamento da primeira tarefa na primeira máquina. A partir daí, é calculado a data de término das tarefas em todas as máquinas, com exceção da última, que é calculada na equação (54), da mesma forma que em Permutação.

O cálculo da equação (39) à (53), é feito da seguinte forma: A equação (39) faz uma verificação, se a tarefa que está ocasionando o bloquei terminou, para que possa ser acrescentado espaço no *buffer* daquela máquina (equação (40)). A equação (41), verifica se a tarefa está na primeira máquina e caso ela esteja e tenha espaço no *buffer* (equação (42)), executa a equação (43) que encontra a data de término da tarefa na primeira máquina, caso seja falsa a equação (42), a data de término da tarefa na primeira máquina é calculada pela equação (44), que considera a maior data de término entre a tarefa que pode causar o bloqueio, e a data de término da tarefa anterior mais o seu processamento na máquina. Sendo falsa a equação (41), ou seja, para as máquinas de dois até (*m*-1), vai diretamente para equação (45), que verifica se há espaço no *buffer*, caso haja, calcula-se normalmente como na restrição do tipo permutação, através das equações (46) e (47). No caso do b*uffer* esteja cheio, então da equação (45), vai para equação (48) e (49) que além de fazer as duas considerações do caso normal de permutação, considera também, a data de término da tarefa que pode ocasionar o bloqueio.

Na equação (50), tem-se uma verificação, se a tarefa que poderá bloquear as demais, ainda está em processamento, caso esteja, o *buffer*, recebe mais uma tarefa (equação(51)). A equação (52) verifica, se o vetor utilizado para encontrar a tarefa que fará o bloqueio, está no seu máximo, caso não esteja, é acrescentada uma unidade por tarefa concluída (equação (53)). Após percorrer as máquinas até *m-1*, é calculado a data de término da tarefa na última máquina (máquina *m*), pela equação (54) que é como a equação (4) de permutação, isto porque, na última máquina, não existe tarefa à frente que possa ocasionar o bloqueio.

4. RESULTADOS

Os resultados foram obtidos através de um exemplo selecionado, com o intuito de mostrar o funcionamento de cada equação desenvolvida na seção anterior. O exemplo foi selecionado, com intuito de que contemplasse todas equações e condições desenvolvidas, de forma a encontrar o mesmo resultado previamente calculado e verificado. Sendo o exemplo utilizado:

$P_{jk} =$

	k_1	k_2	k_3	k_4
j_1	3	2	9	3
j_2	2	3	2	1
j_3	3	1	3	2
j_4	2	2	1	3
j_5	1	5	3	1

Ordem = [1 2 3 4 5]

4.1 Permutação

Abaixo é demonstrado como é feito o cálculo utilizando as equações desenvolvidas para restrição do tipo permutação:

Conforme equação (1), temos:

$$C_{[1]1} = P_{[1]1} = 3$$

Seguindo para equação (2), onde considera-se a soma do tempo de processamento da tarefa na máquina anterior, encontramos:

$$C_{[1]2} = P_{[1]2} + C_{[1](2-1)} = 2 + 3 = 5$$
$$C_{[1]3} = P_{[1]3} + C_{[1](3-1)} = 9 + 5 = 14$$
$$C_{[1]4} = P_{[1]4} + C_{[1](4-1)} = 3 + 14 = 17$$

A equação (3), obtém a data de término de todas as tarefas na primeira máquina, sendo a soma do tempo de processamento com o término da tarefa anterior na máquina um:

$$C_{[2]1} = P_{[2]1} + C_{[2-1]1} = 2 + 3 = 5$$
$$C_{[3]1} = P_{[3]1} + C_{[3-1]1} = 3 + 5 = 8$$
$$C_{[4]1} = P_{[4]1} + C_{[4-1]1} = 2 + 8 = 10$$
$$C_{[5]1} = P_{[5]1} + C_{[5-1]1} = 1 + 10 = 11$$

Assim, pode-se encontrar as datas de término das demais tarefas nas máquinas restantes pela equação (4):

$$C_{[2]2} = P_{[2]2} + \max(C_{[2-1]2};C_{[2](2-1)}) = 3 + \max(5;5) = 8$$
$$C_{[3]2} = P_{[3]2} + \max(C_{[3-1]2};C_{[3](2-1)}) = 1 + \max(8;8) = 9$$
$$C_{[4]2} = P_{[4]2} + \max(C_{[4-1]2};C_{[4](2-1)}) = 2 + \max(9;10) = 12$$
$$C_{[5]2} = P_{[5]2} + \max(C_{[5-1]2};C_{[5](2-1)}) = 5 + \max(12;11) = 17$$
...

A equação (4), seria utilizada novamente, para o valor de k=3, e assim sucessivamente até o valor de k=m, encontrando como resultado:

$$C_{jk} = \begin{array}{c|cccc} & k_1 & k_2 & k_3 & k_4 \\ j_1 & 3 & 5 & 14 & 17 \\ j_2 & 5 & 8 & 16 & 18 \\ j_3 & 8 & 9 & 19 & 21 \\ j_4 & 10 & 12 & 20 & 24 \\ j_5 & 11 & 17 & 23 & 25 \end{array}$$

O gráfico de GANTT da figura 3, mostra o resultado obtido:

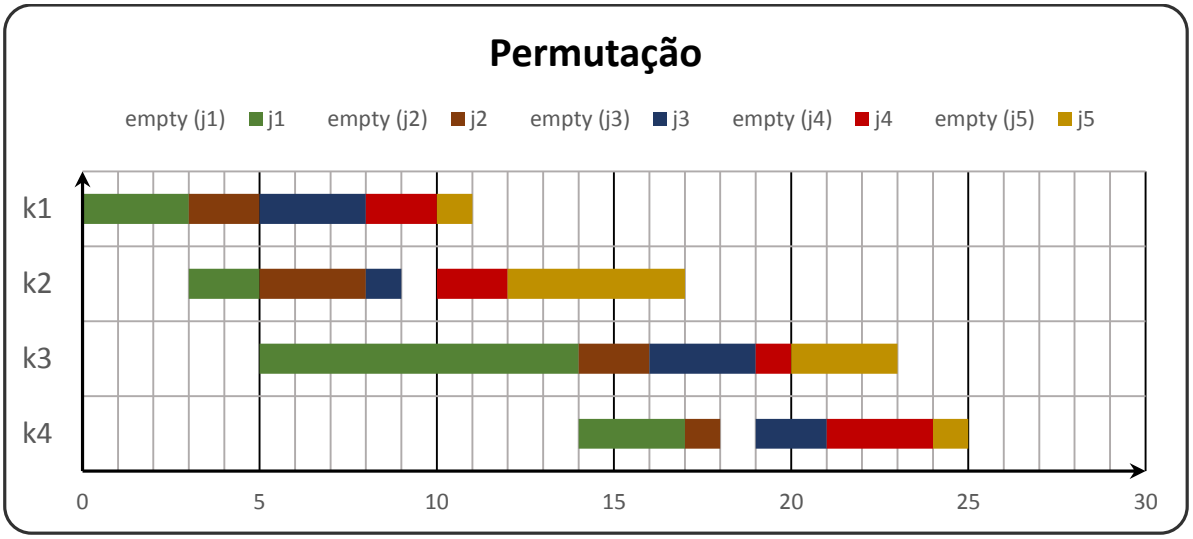

Figura 3 - Gráfico da restrição de permutação.

Fonte: Autoria própria.

Do gráfico acima, pode-se tirar como informação o *Makespan* (última tarefa a deixar a última máquina), e observar que quase não há *empty*, ou seja, tempo em que a máquina não está processando nada (vazia). Tendo assim um bom aproveitamento da sequência.

4.2 Tempo de *setup*

Para a restrição de Tempo de setup, é considerado a seguinte matriz com os tempos de setup de cada tarefa:

$$S_{ij} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 3 & 2 & 5 & 2 \\ j_2 & 5 & 0 & 3 & 7 & 1 \\ j_3 & 2 & 4 & 0 & 1 & 2 \\ j_4 & 7 & 1 & 2 & 0 & 2 \\ j_5 & 9 & 5 & 4 & 2 & 0 \end{array}$$

As equações (5) e (6), de forma análoga as (1) e (2) de permutação, não precisa de *setup* para a primeira tarefa. Assim, o resultado encontrado da primeira tarefa em todas as máquinas é idêntico ao encontrado na restrição anterior.

Das equações (7) e (8) determina-se a data de término de todas as tarefas na primeira máquina, agregando o tempo de *setup* pela mudança de tarefa:

$$C_{[2]1} = P_{[2]1} + C_{[2-1]1} + S_{[(2-1)][2]} = 2 + 3 + 3 = 8$$
$$C_{[3]1} = P_{[3]1} + C_{[3-1]1} + S_{[(3-1)][3]} = 3 + 8 + 3 = 14$$
$$C_{[4]1} = P_{[4]1} + C_{[4-1]1} + S_{[(4-1)][4]} = 2 + 14 + 1 = 17$$
$$C_{[5]1} = P_{[5]1} + C_{[5-1]1} + S_{[(5-1)][5]} = 1 + 17 + 2 = 20$$

Utilizando a equação (9) juntamente com a (10), encontra-se as datas de término das demais tarefas, também considerando o tempo de *setup*, quando necessário:

$$C_{[2]2} = P_{[2]2} + \max((C_{[2-1]2} + S_{[2-1][2]});C_{[2](2-1)}) = 3 + \max((5+3);8) = 11$$
$$C_{[3]2} = P_{[3]2} + \max((C_{[3-1]2} + S_{[3-1][3]});C_{[3](2-1)}) = 1 + \max((11+3);14) = 15$$
$$C_{[4]2} = P_{[4]2} + \max((C_{[4-1]2} + S_{[4-1][4]});C_{[4](2-1)}) = 2 + \max((15+1);17) = 19$$
$$C_{[5]2} = P_{[5]2} + \max((C_{[5-1]2} + S_{[5-1][5]});C_{[5](2-1)}) = 5 + \max((19+2);20) = 26$$

…

Da mesma forma que foi feito para à máquina 2, utilizando as equações (9) e (10), é seguido a mesma lógica até a última máquina, encontrando como resultado:

$$C_{jk} = \begin{array}{c|cccc} & k_1 & k_2 & k_3 & k_4 \\ j_1 & 3 & 5 & 14 & 17 \\ j_2 & 8 & 11 & 19 & 21 \\ j_3 & 14 & 15 & 25 & 27 \\ j_4 & 17 & 19 & 27 & 31 \\ j_5 & 20 & 26 & 32 & 34 \end{array}$$

O gráfico de GANTT da figura 4, mostra o resultado obtido:

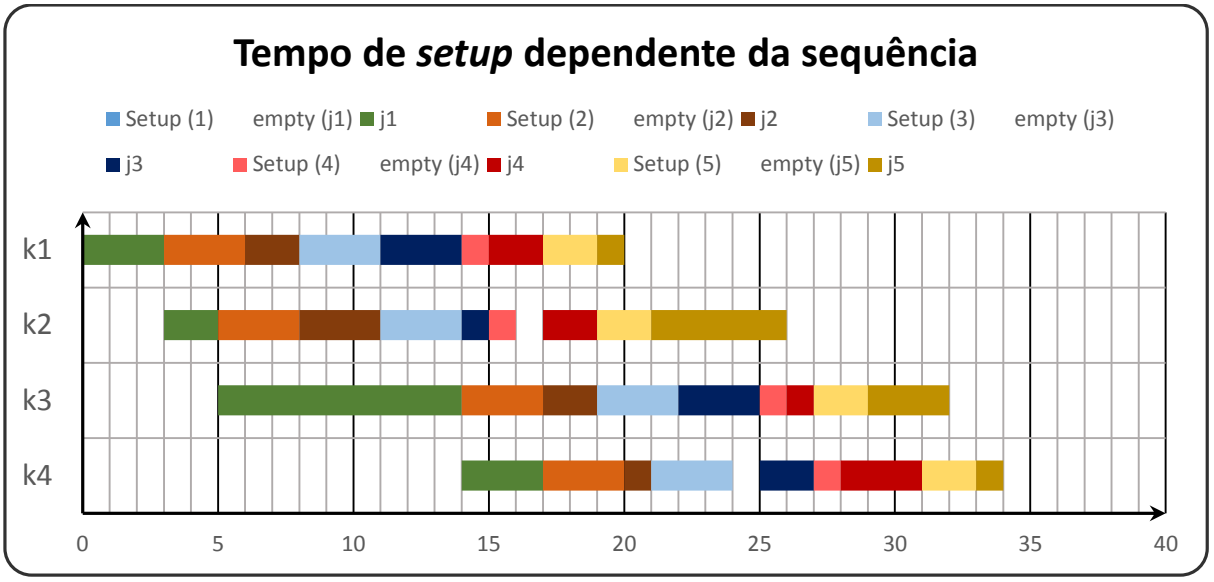

Figura 4 - Gráfico da restrição de tempo de setup dependente da sequência.

Fonte: Autoria própria.

Através do gráfico, pode-se notar que o tempo de *setup*, muitas vezes é maior que o próprio tempo de processamento da tarefa. Uma maneira de minimizar o tempo de *setup*, seria a troca da sequência (ordem) de operação.

4.3 Tempo de *setup* máquina

Na restrição de Tempo de setup dependendo da sequência e da máquina, temos as seguintes matrizes que informam o tempo de setup, dependendo também da máquina:

$$S_{ij1} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 3 & 2 & 5 & 2 \\ j_2 & 5 & 0 & 3 & 7 & 1 \\ j_3 & 2 & 4 & 0 & 1 & 2 \\ j_4 & 7 & 1 & 2 & 0 & 2 \\ j_5 & 9 & 5 & 4 & 2 & 0 \end{array} ; \quad S_{ij2} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 5 & 11 & 7 & 6 \\ j_2 & 4 & 0 & 2 & 3 & 2 \\ j_3 & 2 & 6 & 0 & 2 & 4 \\ j_4 & 1 & 3 & 5 & 0 & 2 \\ j_5 & 5 & 9 & 2 & 4 & 0 \end{array} ;$$

$$S_{ij3} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 1 & 9 & 5 & 2 \\ j_2 & 6 & 0 & 1 & 8 & 3 \\ j_3 & 2 & 3 & 0 & 1 & 5 \\ j_4 & 1 & 8 & 4 & 0 & 3 \\ j_5 & 9 & 11 & 7 & 8 & 0 \end{array} ; S_{ij4} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 3 & 2 & 7 & 6 \\ j_2 & 6 & 0 & 1 & 9 & 15 \\ j_3 & 3 & 9 & 0 & 1 & 8 \\ j_4 & 2 & 5 & 1 & 0 & 4 \\ j_5 & 7 & 6 & 9 & 10 & 0 \end{array}$$

Da mesma forma que na restrição de Tempo de *setup*, a primeira tarefa não possui *setup*, independente da máquina, assim o resultado da equação (11) e (12), também são iguais aos das equações (1) e (2) de Permutação.

As equações (13) e (14) são utilizadas como as equações (7) e (8) da restrição tecnológica de Tempo de *setup*. Porém, na equação (13), considera também em qual máquina está ocorrendo a restrição e como para o exemplo, a máquina um, possui a mesma matriz de tempo de *setup* do exemplo anterior, então o resultado será o mesmo para todas tarefas na primeira máquina.

Os resultados, mudam a partir da máquina dois, uma vez que o *setup* desta máquina é diferente, assim, os resultados podem ser encontrados a partir das equações (15) e (16):

$$C_{[2]2} = P_{[2]2} + \max((C_{[2-1]2} + S_{[2-1][2]2});C_{[2](2-1)}) = 3 + \max((5+5);8) = 13$$

$$C_{[3]2} = P_{[3]2} + \max((C_{[3-1]2} + S_{[3-1][3]2});C_{[3](2-1)}) = 1 + \max((13+2);14) = 16$$

$$C_{[4]2} = P_{[4]2} + \max((C_{[4-1]2} + S_{[4-1][4]2});C_{[4](2-1)}) = 2 + \max((16+2);17) = 20$$

$$C_{[5]2} = P_{[5]2} + \max((C_{[5-1]2} + S_{[5-1][5]2});C_{[5](2-1)}) = 5 + \max((20+2);20) = 27$$

…

Seguindo a mesma linha de raciocínio utilizando as equações (15) e (16), para as demais máquinas, encontramos como resultado:

$$C_{jk} = \begin{array}{c|cccc} & k_1 & k_2 & k_3 & k_4 \\ j_1 & 3 & 5 & 14 & 17 \\ j_2 & 8 & 13 & 17 & 21 \\ j_3 & 14 & 16 & 21 & 24 \\ j_4 & 17 & 20 & 23 & 28 \\ j_5 & 20 & 27 & 30 & 33 \end{array}$$

O gráfico de GANTT da figura 5, mostra o resultado obtido:

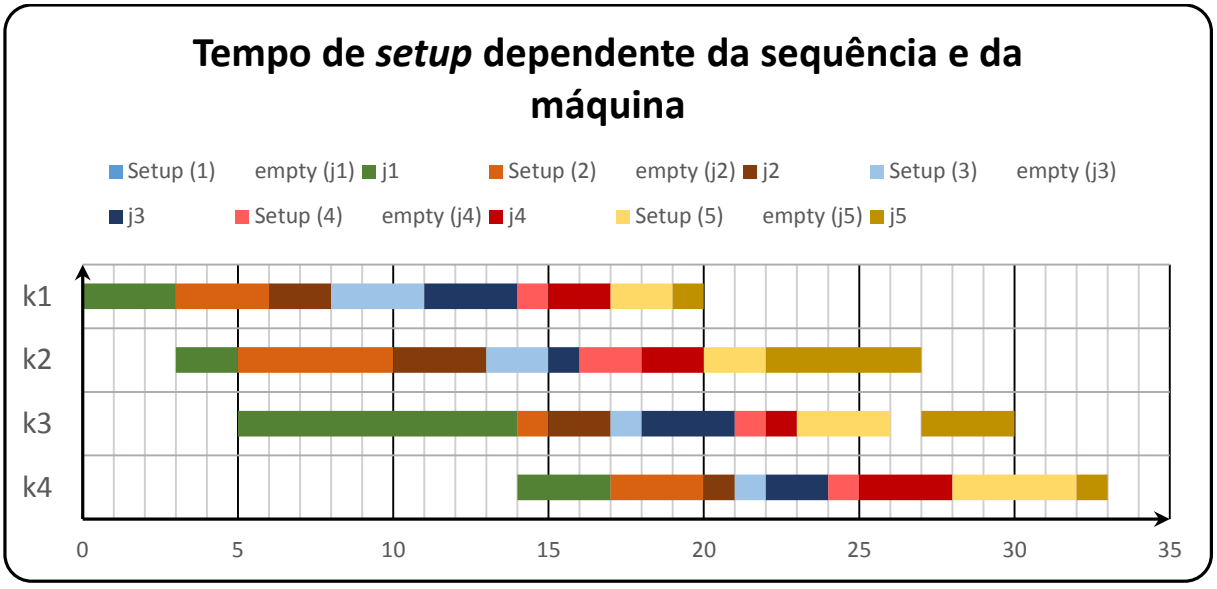

Figura 5 - Gráfico da restrição de setup dependente da sequência e da máquina.

Fonte: Autoria própria.

Como no gráfico da restrição anterior, pode-se notar grandes tempos de setup, no caso desta restrição que depende também da máquina, este tempo em termos gerais teve um aumento, devido a maiores tempos de *setup* (Mas este tempo pode também diminuir). Por este motivo encontra-se um *Makespan* relativamente alto para esta sequência.

4.4 Sem espera

O termo sem espera, é uma restrição tecnológica, em que a tarefa só é liberada para ser iniciada na primeira máquina, após a certeza de que ela não sofrerá nenhuma parada nas próximas máquinas.

Assim, como nada restringe a tarefa um, pode-se considerar os resultados das equações (17) e (18), como iguais aos das equações (1) e (2).

As equações (19) e (20), calculam a diferença entre o tempo de processamento da tarefa anterior na máquina atual e o tempo de processamento da tarefa atual na máquina anterior. Após, considera-se a maior diferença, ou o somatório das diferenças, caso ele seja maior. Assim, este maior valor, será o intervalo em que a

tarefa terá que adiar seu início para que não seja interrompida à frente (caso o somatório seja menor que zero, considera-se zero como o intervalo):

$$E_{[2]1} = \max(0;(P_{[2-1]2} - P_{[2](2-1)}); + (P_{[2-1]3} - P_{[2](3-1)}); + (P_{[2-1]4} - P_{[2](4-1)})) \ldots$$
$$\ldots = \max(0;((2-2) + (9-3) + (3-2))) = 7$$
$$E_{[3]1} = \max(0;(P_{[3-1]2} - P_{[3](2-1)}); + (P_{[3-1]3} - P_{[3](3-1)}); + (P_{[3-1]4} - P_{[3](4-1)})) \ldots$$
$$\ldots = \max(0;((3-3) + (2-1) + (1-3))) = 1$$
$$E_{[4]1} = \max(0;(P_{[4-1]2} - P_{[4](2-1)}); + (P_{[4-1]3} - P_{[4](3-1)}); + (P_{[4-1]4} - P_{[4](4-1)})) \ldots$$
$$\ldots = \max(0;((1-2) + (3-2) + (2-1))) = 1$$
$$E_{[5]1} = \max(0;(P_{[5-1]2} - P_{[5](2-1)}); + (P_{[5-1]3} - P_{[5](3-1)}); + (P_{[5-1]4} - P_{[5](4-1)})) \ldots$$
$$\ldots = \max(0;((2-1) + (1-5) + (3-3))) = 0$$

Como o tempo em que a tarefa deve ser adiada para que não seja interrompida à frente foi determinado. Com a equação (21), encontra-se as datas de términos das tarefas em cada máquina, como é calculado em permutação na equação (2). Sendo a única diferença, o acréscimo do tempo de intervalo encontrado para se iniciar a tarefa na máquina um, obtendo assim:

Eq. 21) $C_{jk} =$

	k_1	k_2	k_3	k_4
j_1	3	5	14	17
j_2	12	15	17	18
j_3	16	17	20	22
j_4	19	21	22	25
j_5	21	26	29	30

O gráfico de GANTT da figura 6, mostra o resultado obtido:

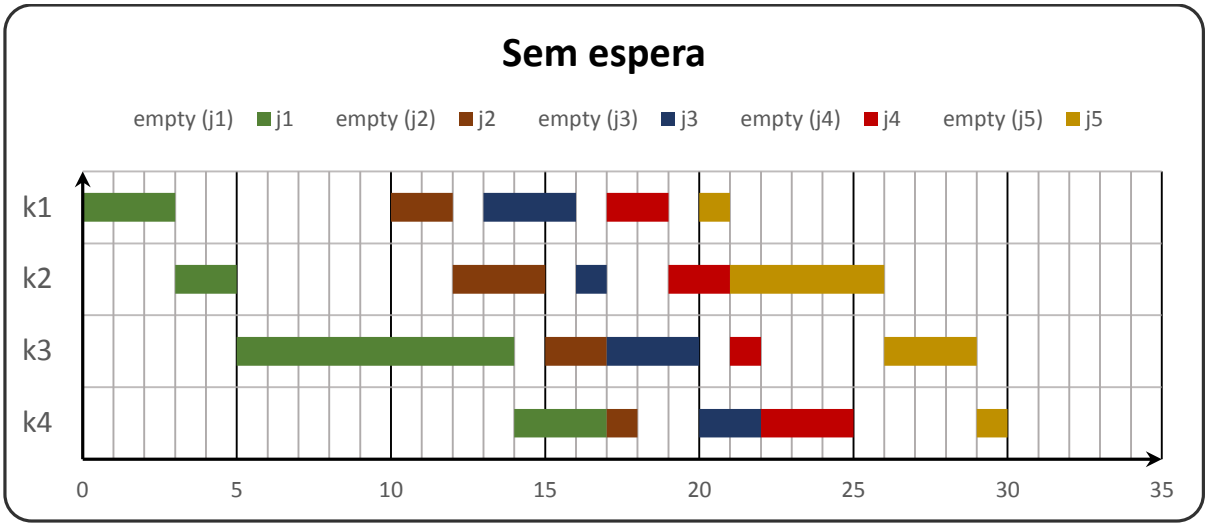

Figura 6 - Gráfico da restrição de sem espera.

Fonte: Autoria própria.

Para o gráfico 6, observa-se grandes intervalos de *empty* (máquina vazia), o que é esperado, uma vez que a tarefa só inicia, quando tem certeza de que não haverá nenhum bloqueio à frente.

4.5 Família de trabalho

A restrição de Família de trabalho, pode ser considerado uma derivação da restrição de Tempo de setup, uma vez que o setup também vai ocorrer. Porém, para esta restrição, o setup, irá ocorrer somente quando a tarefa anterior não for considerada da mesma família de trabalho da tarefa que irá ser processada. Sendo a matriz de tempo de *setup* e o vetor de quais tarefas pertencem à mesma família:

$$S_{ij} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 0 & 3 & 2 & 5 & 2 \\ j_2 & 5 & 0 & 3 & 7 & 1 \\ j_3 & 2 & 4 & 0 & 1 & 2 \\ j_4 & 7 & 1 & 2 & 0 & 2 \\ j_5 & 9 & 5 & 4 & 2 & 0 \end{array}$$

$$Fa = [2; 1; 1; 3; 2]$$

Como a matriz é igual à da restrição de Tempo de *setup*, desta forma, tem-se o resultado de término da primeira tarefa em todas as máquinas pelas equações (22)

e (23), iguais aos resultados encontrados pelas equações (11) e (12), que por sua vez, são iguais as equações (1) e (2).

Para a determinação da data de término de todas as tarefas na primeira máquina, verifica-se pela equação (24), se as tarefas são da mesma família. Caso sejam, calcula-se normalmente pela equação (25) (Como realizado também em permutação). Não sendo verdadeiro, é adicionado neste cálculo (equação (26)), um tempo de *setup* devido à troca de família, conforme nota-se abaixo:

$$C_{[2]1} = P_{[2]1} + C_{[2-1]1} + S_{[(2-1)][2]} = 2 + 3 + 3 = 8$$
$$C_{[3]1} = P_{[3]1} + C_{[(3-1)]1} = 3 + 8 = 11$$
$$C_{[4]1} = P_{[4]1} + C_{[4-1]1} + S_{[(4-1)][4]} = 2 + 11 + 1 = 14$$
$$C_{[5]1} = P_{[5]1} + C_{[5-1]1} + S_{[(5-1)][5]} = 1 + 14 + 2 = 17$$

Na segunda linha da equação, observa-se que não há a soma de *setup*, isto porque, a tarefa três, pertence à mesma família da tarefa anterior (dois). Estes resultados porém, foram para a data de término de todas tarefas na primeira máquina. Para as demais máquinas, segue-se o mesmo raciocínio, adicionando um comparativo na equação (29), que considera a data de término da tarefa na máquina anterior:

$$C_{[2]2} = P_{[2]2} + \max((C_{[2-1]2} + S_{[2-1][2]});C_{[2](2-1)}) = 3 + \max((5+3);8) = 11$$
$$C_{[3]2} = P_{[3]2} + \max(C_{[3-1]2} ;C_{[3](2-1)}) = 1 + \max(11;11) = 12$$
$$C_{[4]2} = P_{[4]2} + \max((C_{[4-1]2} + S_{[4-1][4]});C_{[4](2-1)}) = 2 + \max((12+1);14) = 16$$
$$C_{[5]2} = P_{[5]2} + \max((C_{[5-1]2} + S_{[5-1][5]});C_{[5](2-1)}) = 5 + \max((16+2);17) = 23$$
$$\dots$$

Observa-se, que novamente não existe *setup*, da tarefa dois para tarefa três. Segue-se executando estas mesmas equações até a máquina *k=m (m=4)*. Encontrando como resultado:

$$C_{jk} = \begin{array}{c|cccc} & k_1 & k_2 & k_3 & k_4 \\ j_1 & 3 & 5 & 14 & 17 \\ j_2 & 8 & 11 & 19 & 21 \\ j_3 & 11 & 12 & 22 & 24 \\ j_4 & 14 & 16 & 24 & 28 \\ j_5 & 17 & 23 & 29 & 31 \end{array}$$

O gráfico de GANTT da figura 7, mostra o resultado obtido:

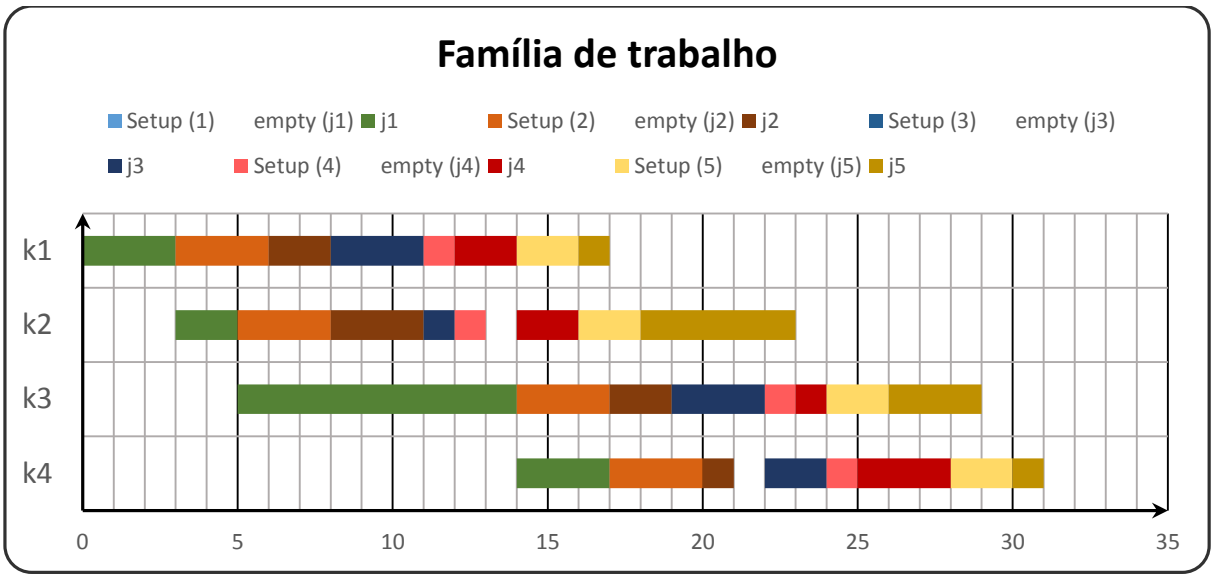

Figura 7 - Gráfico da restrição de família de trabalho.

Fonte: Autoria própria.

No gráfico da figura 7, nota-se que não houve setup nas máquinas ao mudar a execução da tarefa dois para tarefa três, diferente do que ocorre nas outras mudanças de tarefas, onde podemos notar a presença de *setup* nas máquinas.

4.6 Data lançamento

Para restrição tecnológica de Data de lançamento, utilizou-se juntamente com a restrição de Tempo de setup dependente da sequência e da máquina.

O que realmente diferencia esta restrição, é que para se iniciar uma tarefa em uma determinada máquina, deve-se considerar no comparativo de tempo, se a data em que a tarefa pode ser lançada na máquina para seu início, está sendo respeitada.

Para data de lançamento, utilizou-se das mesmas matrizes da restrição de Tempo de *setup* máquina. Porém, foi adicionado valores às diagonais principais, ou seja, a data em que a tarefa pode ser iniciada na máquina.

Considerando: $S_{ij1} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 1 & 3 & 2 & 5 & 2 \\ j_2 & 5 & 9 & 3 & 7 & 1 \\ j_3 & 2 & 4 & 6 & 1 & 2 \\ j_4 & 7 & 1 & 2 & 13 & 2 \\ j_5 & 9 & 5 & 4 & 2 & 7 \end{array}$; $S_{ij2} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 3 & 5 & 11 & 7 & 6 \\ j_2 & 4 & 12 & 2 & 3 & 2 \\ j_3 & 2 & 6 & 17 & 2 & 4 \\ j_4 & 1 & 3 & 5 & 12 & 2 \\ j_5 & 5 & 9 & 2 & 4 & 9 \end{array}$;

$S_{ij3} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 4 & 1 & 9 & 5 & 2 \\ j_2 & 6 & 4 & 1 & 8 & 3 \\ j_3 & 2 & 3 & 20 & 1 & 5 \\ j_4 & 1 & 8 & 4 & 7 & 3 \\ j_5 & 9 & 11 & 7 & 8 & 1 \end{array}$; $S_{ij4} = \begin{array}{c|ccccc} & j_1 & j_2 & j_3 & j_4 & j_5 \\ j_1 & 11 & 3 & 2 & 7 & 6 \\ j_2 & 6 & 4 & 1 & 9 & 15 \\ j_3 & 3 & 9 & 7 & 1 & 8 \\ j_4 & 2 & 5 & 1 & 2 & 4 \\ j_5 & 7 & 6 & 9 & 10 & 3 \end{array}$

Como foi utilizado de base, a restrição de tempo de *setup* máquina. Caso não haja restrição de lançamento, o resultado fica como à restrição de base. Havendo à restrição por data de lançamento, as datas alteram-se, como pode-se notar pelas equações (30) e (31), que determinam o término de processamento da primeira tarefa em todas as máquinas:

$$C_{[1]1} = P_{[1]1} + S_{[1][1]1} = 3 + 1 = 4$$

$$C_{[1]2} = P_{[1]2} + \max(S_{[1][1]2};C_{[1](2-1)}) = 2 + \max(3;4) = 6$$

$$C_{[1]3} = P_{[1]3} + \max(S_{[1][1]3};C_{[1](3-1)}) = 9 + \max(4;6) = 15$$

$$C_{[1]4} = P_{[1]4} + \max(S_{[1][1]4};C_{[1](4-1)}) = 3 + \max(11;15) = 18$$

Encontrado a data de término da primeira tarefa em todas máquinas, calcula-se o término de todas tarefas na primeira máquina, como em Tempo de *setup* máquina. Porém, adicionando a comparação com a data de lançamento (inicio), para que caso a tarefa esteja pronta para iniciar antes, ela entre em espera até a data programada. Isto é feito nas equações (32) e (33):

$$C_{[2]1} = P_{[2]1} + \max((C_{[2-1]1} + S_{[2-1][2]1});S_{[2][2]1}) = 2 + \max((4+3);9) = 11$$
$$C_{[3]1} = P_{[3]1} + \max((C_{[3-1]1} + S_{[3-1][3]1});S_{[3][3]1}) = 3 + \max((11+3);6) = 17$$
$$C_{[4]1} = P_{[4]1} + \max((C_{[4-1]1} + S_{[4-1][4]1});S_{[4][4]1}) = 2 + \max((17+1);13) = 20$$
$$C_{[5]1} = P_{[5]1} + \max((C_{[5-1]1} + S_{[5-1][5]1});S_{[5][5]1}) = 1 + \max((20+2);7) = 23$$

Para as demais máquinas, é seguido o mesmo raciocínio. Mas como a partir da máquina dois, também pode ocorrer o caso da tarefa não estar pronta na máquina anterior, então, cria-se este comparativo junto aos outros dois citados anteriores. Assim, o resultado obtido pelas equações (34), (35) e (36) foi:

$$C_{[2]2} = P_{[2]2} + \max((C_{[2-1]2} + S_{[2-1][2]2});C_{[2](2-1)};S_{[2][2]2}) = 3 + \max((6+5);11;12) = 15$$
$$C_{[3]2} = P_{[3]2} + \max((C_{[3-1]2} + S_{[3-1][3]2});C_{[3](2-1)};S_{[3][3]2}) = 1 + \max((15+2);17;17) = 18$$
$$C_{[4]2} = P_{[4]2} + \max((C_{[4-1]2} + S_{[4-1][4]2});C_{[4](2-1)};S_{[4][4]2}) = 2 + \max((18+2);20;12) = 22$$
$$C_{[5]2} = P_{[5]2} + \max((C_{[5-1]2} + S_{[5-1][5]2});C_{[5](2-1)};S_{[5][5]2}) = 5 + \max((22+2);23;9) = 29$$
$$\ldots$$

Como *k* varia de 2 até *m*, as equações (34), (35) e (36) são executadas para todas as máquinas restantes, encontrando como resultado, a seguinte matriz:

$$C_{jk} = \begin{array}{c|cccc} & k_1 & k_2 & k_3 & k_4 \\ j_1 & 4 & 6 & 15 & 18 \\ j_2 & 11 & 15 & 18 & 22 \\ j_3 & 17 & 18 & 23 & 25 \\ j_4 & 20 & 22 & 25 & 29 \\ j_5 & 23 & 29 & 32 & 34 \end{array}$$

O gráfico de GANTT da figura 8, mostra o resultado obtido:

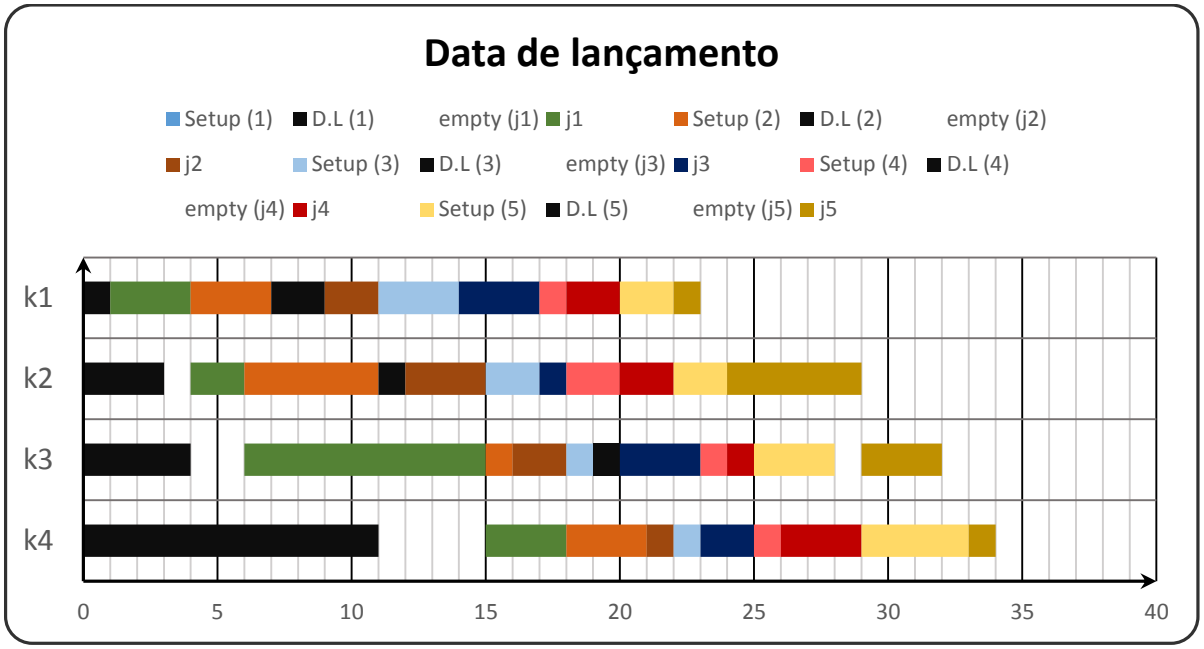

Figura 8 - Gráfico da restrição de data de lançamento.

Fonte: Autoria própria.

Nota-se na figura 8, que em alguns casos, mesmo as máquinas estando com o *setup* feito, as tarefas não são iniciadas. Elas são liberadas para processamento somente quando chega sua data de lançamento. Assim, quando isto ocorre, gera um atraso na conclusão da tarefa e também pode deixar máquinas em vazio.

4.7 Bloqueio B

Nesta restrição tecnológica, quando a tarefa termina seu processamento na máquina, é feito uma avaliação, se existe espaço no *Buffer* (espaço onde a tarefa aguarda chegar o início de seu processamento na próxima máquina). Quando existe este espaço, a tarefa libera a máquina onde estava, quando não existe mais espeço, a tarefa então, fica na máquina, bloqueando assim, a próxima tarefa.

Para o exemplo, foi considerado o seguinte vetor de *Buffer*:

$$B = [0; 2; 1]$$

Sendo, assim, temos *Buffer* zero entre a máquina um e dois, *Buffer* dois entre as máquinas dois e três e *Buffer* um entre as máquinas três e quatro.

Assim, para primeira tarefa, não ocorre bloqueio, tendo como resultado encontrado nas equações (37) e (38), iguais aos resultados encontrados nas equações (1) e (2) da restrição de permutação.

Após calcular o término da primeira tarefa em todas as máquinas, é realizado o cálculo para as demais tarefas em todas as máquinas restantes. Para isto, considera-se algumas condições de avaliação, como:

- Se tem alguma tarefa ocasionando bloqueio;
- Se a tarefa está na máquina um;
- Se existe espaço no b*uffer*;
- Se o b*uffer* deve receber mais uma tarefa.

Através destas condições, quando não ocorre bloqueio, devido à máquina estar liberada, ou ter espaço no b*uffer*, o cálculo é feito da mesma forma, que é feito na restrição do tipo Permutação. Quando a próxima máquina está ocupada e o *buffer* está cheio, a tarefa então, é bloqueada na máquina em que está, aguardando até que a tarefa que está ocasionando o bloqueio termine seu processamento. Assim, obtém-se como resultado:

Eq. 47) C_{jk} =

	k_1	k_2	k_3	k_4
j_1	3	5	14	17
j_2	5	8	16	0
j_3	8	9	19	0
j_4	10	14	20	0
j_5	14	19	23	0

Como na última máquina (*k*=*m*) não tem como existir bloqueio, devido a não ter processamento à sua frente, ela é calculada separadamente, com a equação (54).

Assim, obtêm-se como resultado final da restrição de Bloqueio, a seguinte matriz:

Eq. 54) C_{jk} =

	k_1	k_2	k_3	k_4
j_1	3	5	14	17
j_2	5	8	16	18
j_3	8	9	19	21
j_4	10	14	20	24
j_5	14	19	23	25

O gráfico de GANTT da figura 9, mostra o resultado obtido:

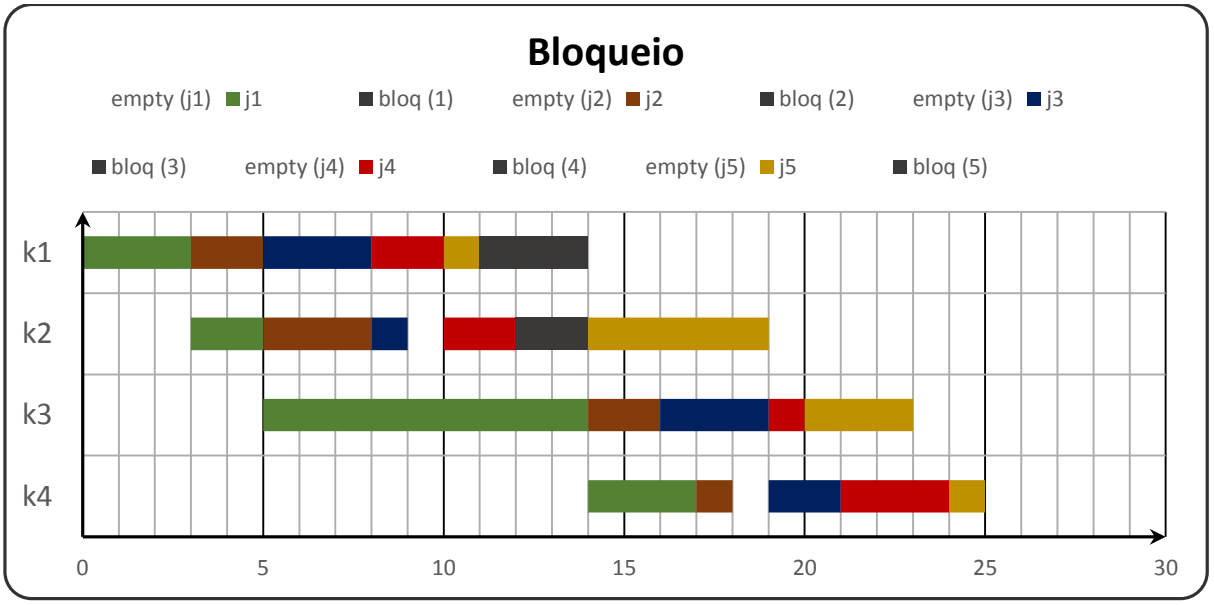

Figura 9 - Gráfico da restrição de bloqueio.

Fonte: Autoria própria.

Para o gráfico de Bloqueio, pode-se notar há presença de bloqueio em duas tarefas, na tarefa quatro na máquina dois e tarefa cinco na máquina um. Na primeira situação, o que acontece, é que mesmo tendo dois espaços no b*uffer* da máquina dois para máquina três, tem-se um grande tempo de processamento da tarefa um na máquina três, assim, termina-se as outras próximas três tarefas na máquina dois, sem que a tarefa um termine na máquina três, por este motivo a quarta tarefa é bloqueada na máquina dois. Como consequência da tarefa quatro ficar bloqueada, não permite-se que a tarefa cinco seja liberada na máquina um, porque, pode-se observar que o *buffer* da máquina um para máquina dois é zero.

Das restrições tecnológicas vistas em ambientes *flow shop,* pode-se destacar com maior *Makespan*, restrições que apresentam *setup*, que é um fator relevante, uma vez que em muitos casos os valores de *setup* são maiores do que os de processamento. Mas vale aplicação de tecnologia e outros métodos para reduzir estes tempos de *setup* e maximizar o tempo de produção da máquina.

5. CONCLUSÃO

Pode-se avaliar o desenvolvimento das equações deste trabalho como ferramenta de grande contribuição para as indústrias com linha de produção do tipo *flow shop*. Essa contribuição aumenta sua importância proporcionalmente à quantidade de máquinas, que a empresa possui e à quantidade de tarefas processadas por essas máquinas. Salienta-se que, as equações são utilizadas como ferramentas para encontrar a data de término do processamento de cada tarefa em cada máquina, o que são, dados importantes nas tomadas de decisões para melhoramento e otimização do processo de fabricação.

As alterações planejadas no sequenciamento da produção podem ser testadas e avaliadas, antes mesmo de serem aplicadas, o que aumenta a possibilidade de sucesso na implementação, além de ter a possibilidade de simplesmente alterar a ordem da sequência, sem necessidade de inserir novamente todos os dados, para testar novas possibilidades que melhorem a produção.

Como possíveis trabalhos futuros pode-se citar uma continuação do trabalho desenvolvido, ao complementar os programas, com a possibilidade de geração do gráfico de GANTT do resultado obtido. Pode-se também, estudar o desenvolvimento das equações para outros ambientes, como por exemplo: *Job Shop*; *Open Shop*; Máquina única. As equações desenvolvidas, podem ainda, servir de base para o desenvolvimento de métodos de otimização de diferentes classes de problemas e com diferentes critérios de avaliação.

REFERÊNCIAS BIBLIOGRÁFICAS

TUBINO, D. F. **Planejamento e controle da produção:** Teoria e prática. 1ª ed. SÃO PAULO: Atlas, 2007, 190 p.

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

PINEDO, L. M. **Scheduling:** Theory, Algorithms and Systems. 3 ed. Nova York: Springer Science+Business Media, LLC, 2008, 671 p.

MACCARTHY, B. L; LIU, J. Addressing the gap in scheduling: a review of optimization and heuristic methods in production scheduling. **International Journal of Production Research**, v. 31, p. 59-79, 1993.

QIAN, B; WANG, L; HU, R; HUANG, D. X; WANG, X. A DE-based approach to no-wait flow-shop scheduling. **Computers & Industrial Engineering**, v. 57, p. 787-805, 2009.

APÊNDICES

APÊNDICE A – PERMUTAÇÃO

```
function [Cjk]=Permutacao(Pjk,Ordem)
[n,m]=size(Pjk);

Cjk=zeros(n,m);
%Calculando a data de conclusão para a primeira tarefa
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

%Calculando a data de conclusão de todas as tarefas na primeira máquina
Cjk(1:n,1)=cumsum(Pjk(Ordem(1:n),1),1);

%Calculando a data de conclusão para as demais tarefas

for j=2:n

   for k=2:m
      Cjk(j,k)=Pjk(Ordem(j),k) + max(Cjk(j-1,k),Cjk(j,k-1));
   end

end
```

APÊNDICE B – TEMPO DE *SETUP*

```
function [Rjk,Cjk]= Tempo_de_setup(Pjk,Sjk,Ordem)

[n,m]=size(Pjk);

Cjk=zeros(n,m);
Rjk=zeros(n,m);

% Calculando a data de término da primeira tarefa
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

% Calculando a data de término de todas tarefas na primeira máquina
for j=2:n
   Rjk(j,1)=Cjk((j-1),1)+Sjk(Ordem(j-1),Ordem(j));
   Cjk(j,1)=Rjk(j,1)+Pjk(Ordem(j),1);
end

% Calculando a data de término para as demais tarefas
for k=2:m
   for j=2:n
      Rjk(j,k)= max((Cjk((j-1),k)+Sjk(Ordem(j-1),Ordem(j))),Cjk(j,(k-1)));
      Cjk(j,k)= Rjk(j,k)+Pjk(Ordem(j),k);
   end
end
```

APÊNDICE C – TEMPO DE *SETUP* MAQUINA

```
function [Rjk,Cjk]= Tempo_de_setup_maquina(Pjk,Sijk,Ordem)

[n,m]=size(Pjk);

Cjk=zeros(n,m);
Rjk=zeros(n,m);

% Calculando a data de término da primeira tarefa em todas as máquinas
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

%  Calculando o tempo de término de todas as tarefas na primeira máquina
for j=2:n
   Rjk(j,1)=Cjk((j-1),1)+Sijk(Ordem(j-1),Ordem(j),1);
   Cjk(j,1)=Rjk(j,1)+Pjk(Ordem(j),1);
end

% Calculando o a data de término das tarefas restantes
for k=2:m
   for j=2:n
      Rjk(j,k)= max((Cjk((j-1),k)+Sijk(Ordem(j-1),Ordem(j),k)),Cjk(j,(k-1)));
      Cjk(j,k)= Rjk(j,k)+Pjk(Ordem(j),k);
   end
end
```

APÊNDICE D – SEM ESPERA

```
function [Cjk]=Sem_Espera(Pjk,Ordem)
[n,m]=size(Pjk);

Cjk=zeros(n,m);
Rjk=zeros(n-1,m);
Ejk=zeros(1,m-1);
e=1;
%Calculando a data de conclusão para a primeira tarefa
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

%Calculando a data de conclusão para as demais tarefas

for j=2:n
   for k=2:m
      Rjk(j-1,k-1)=(Rjk(j-1,k-1)+(Pjk(Ordem(j-1),k)-Pjk(Ordem(j),k-1)));
      Rjk(j-1,k)=Rjk(j-1,k-1);
      Ejk(1,e)=max(Ejk(1,e),Rjk(j-1,k-1));
   end
   Cjk(j,:)=(Ejk(1,e)+Pjk(Ordem(j-1),e))+cumsum(Pjk(Ordem(j),:),2);
   e=(e+1);
end
```

APÊNDICE E – FAMILIA DE TRABALHO

```
function [Cjk]= Familia_de_trabalho(Pjk,Sij,Ordem,Fa)

[n,m]=size(Pjk);

Cjk=zeros(n,m);

% Calculando a data de término da primeira tarefa
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

% Calculando a data de término de todas tarefas na primeira máquina
for j=2:n
   if Fa(Ordem(j-1),1)==Fa(Ordem(j),1);
      Cjk(j,1)=Cjk(j-1,1)+Pjk(Ordem(j),1);
   else
      Cjk(j,1)=Cjk(j-1,1)+Pjk(Ordem(j),1)+Sij(Ordem(j-1),Ordem(j));
   end
end

% Calculando a data de término para as demais tarefas
for j=2:n
   for k=2:m
      if Fa(Ordem(j-1),1)==Fa(Ordem(j),1);
         Cjk(j,k)=max(Cjk(j,(k-1)),(Cjk((j-1),k)))+Pjk(Ordem(j),k);
      else
         Cjk(j,k)=max(Cjk(j,(k-1)),(Cjk((j-1),k)+Sij(Ordem(j-1),Ordem(j))))
         +Pjk(Ordem(j),k);
      end
   end
end
```

APÊNDICE F – DATA LANCAMENTO

```
function [Rjk,Cjk]= Data_lancamento(Pjk,Sijk,Ordem)

[n,m]=size(Pjk);

Cjk=zeros(n,m);
Rjk=zeros(n,m);

% Calculando a data de término da primeira tarefa em todas as máquinas
Cjk(1,1)=Pjk(Ordem(1),1)+Sijk(Ordem(1),Ordem(1),1);
for k=2:m
        Cjk(1,k)=max(Cjk(1,(k-1))+Pjk(Ordem(1),k),Sijk(Ordem(1),Ordem(1),k)
        +Pjk(Ordem(1),k));
end

%  Calculando o tempo de término de todas as tarefas na primeira máquina
for j=2:n
        Rjk(j,1)=max((Cjk((j-1),1)+Sijk(Ordem(j-1),Ordem(j),1)),
        Sijk(Ordem(j),Ordem(j),1));
        Cjk(j,1)= Rjk(j,1)+Pjk(Ordem(j),1);
end

% Calculando a data de término das tarefas restantes
for k=2:m
   for j=2:n
      Rjk(j,k)= max((Cjk((j-1),k)+Sijk(Ordem(j-1),Ordem(j),k)),(Cjk(j,(k-1))));
      Rjk(j,k)= max((Sijk(Ordem(j),Ordem(j),k)),Rjk(j,k));
      Cjk(j,k)= Rjk(j,k)+Pjk(Ordem(j),k);
   end
end
```

APÊNDICE G – BLOQUEIO B

```
function [Cjk]=Bloqueio_b(Pjk,Ordem,B)

[n,m]=size(Pjk);

Cjk=zeros(n,m);
Rjk=zeros(n,m);
Ljk=zeros(1,(m-1));
Tjk(1:(m-1))=1;

% Calculando a data de conclusão para a primeira tarefa
Cjk(1,:)=cumsum(Pjk(Ordem(1),:),2);

% Calculando a data de todas as tarefas na primeira máquina.
for j=2:n
   for k=2:m
      if Ljk(1,(k-1))~=0 && (Cjk((j-1),(k-1))+Pjk(Ordem(j),(k-1)))>Cjk(j-Tjk(1,k-1),k)
         Ljk(1,k-1)=(Ljk(1,k-1)-1)
      end
      if (k-1)==1
         if Ljk(1,k-1)<B(k-1)
            Cjk(j,k-1)=(Cjk((j-1),k-1)+Pjk(Ordem(j),k-1))
         else
            Cjk(j,k-1)=max(Cjk(j-(Ljk(1,k-1)+1),k),(Cjk((j-1),k-1)+Pjk(Ordem(j),k-1)))
         end
      else
         if Ljk(1,k-1)<B(k-1)
               Rjk(j,k-1)=max((Cjk((j-1),k-1)+Pjk(Ordem(j),k-1)),(Cjk(j,(k-
               2))+Pjk(Ordem(j),k-1)))
            Cjk(j,k-1)=Rjk(j,k-1)
         else
            Rjk(j,k-1)=max((Cjk((j-1),k-1)+Pjk(Ordem(j),k-1)),(Cjk(j,(k-2))
           +Pjk(Ordem(j),k-1)))
            Cjk(j,k-1)=max(Cjk(j-(Ljk(1,k-1)+1),k),Rjk(j,k-1))
         end
      end
      if Cjk(j,k-1)<Cjk(j-(Ljk(1,k-1)+1),k)
         Ljk(1,k-1)=Ljk(1,k-1)+1
      end
      if Tjk(1,k-1)<=B(k-1)
         Tjk(1,k-1)=Tjk(1,k-1)+1
      end
   end
%     Para k=m.
   Cjk(j,m)=max(Cjk((j-1),m),Cjk(j,(m-1)))+Pjk(Ordem(j),m)
end

end
```