**Título do projeto:**

Job Shop Scheduling

**Projeto submetido para**: PIBIC (X), PIBIC – Af ( ) , PIBIC – EM ( ) , PIBITI ( )

**I. Autor:**

Agnelo Denis Vieira

**II. Projetos financiados por Agência de Fomento:**

( ) Sim ( X ) Não

**III. Projeto Submetido ao PIBITI:**

( ) Sim (X) Não (neste caso, avance para o **item VII**)

**VII. Resumo:**

A participação competitiva das empresas requer o uso eficiente e efetivo dos seus recursos produtivos para atender às demandas do mercado. A indústria de manufatura pode ser organizada de diferentes formas. Os arranjos básicos são denominados *flow shop*, *job shop* e *open shop*. Um problema do tipo job shop visa sequenciar as tarefas, determinando o instante de início de cada operação em cada máquina de forma a minimizar uma função objetivo, tipicamente o tempo máximo necessário para realizar o processamento completo de todas as tarefas (makespan). Conforme demonstrado na literatura, não há algoritmos conhecidos que produzam uma solução ótima em um tempo polinomial. Em função disto são adotadas heurísticas capazes de produzir resultados satisfatórios em tempo de processamento computacional aceitável. Este projeto de pesquisa tem por objetivo geral comparar formulações e algoritmos para obtenção de soluções ótimas e sub ótimas de problemas do tipo *job shop*. Este projeto de pesquisa será desenvolvido através da revisão bibliográfica em livros e artigos de congressos e periódicos; da análise de algoritmos disponíveis nas fontes anteriores bem como em repositórios da comunidade científica; da implementação de algoritmos em linguagens de programação; da análise de aplicativos computacionais utilizados pela comunidade acadêmica e industrial. Tais atividades serão realizadas em computadores pessoais dos pesquisadores envolvidos ou de laboratórios de informática da UFPR. Este projeto de pesquisa não envolve a participação de voluntários humanos ou animais, desta forma considera-se que não é necessária aprovação por Comitês de Ética em Pesquisa.

**VIII. Introdução**

Desde a época da publicação do artigo de (Jain e Meeran 1999), os autores já apontavam que as tendências do mercado, tais como a demanda dos consumidores por variedade de produtos, reduzidos tempos de produção aliado à pressão competitiva para redução de custos resulta na necessidade de minimizar, ou mesmo zerar, estoques. Contudo, para manter a participação no mercado, o sistema produtivo deve ser capaz de responder de forma rápida, o que requer a manutenção de estoques. Estes requisitos conflitantes fazem necessário que a programação das tarefas de produção (*schedulling*), o que é uma atividade essencialmente complexa mesmo para os sistemas mais simples, seja realizada de forma eficiente e efetiva. Como eficiente entende-se que utilize da melhor forma possível os recursos da empresa, enquanto que efetiva entende-se como direcionada a alcançar os objetivos da empresa.

Pinedo (2008) define *schedulling* como um processo decisório que é utilizado de forma regular nas indústrias de manufatura e de serviços com o objetivo de realizar a alocação de recursos para tarefas ao longo de um determinado período de tempo, com o objetivo de otimizar um ou mais objetivos da empresa.

A indústria de manufatura pode ser organizada de diferentes formas. Os arranjos básicos são denominados *flow shop*, *job shop* e *open shop*, sendo que há variações de tais arranjos incluindo diferentes tipos de flexibilidade e a especificação determinística ou probabilística de parâmetros como tempo de processamento, quebra de máquina, possibilidade de interrupção de tarefas já iniciadas. Este projeto está limitado ao âmbito da organização do tipo *job shop*, ou seja, quando há *m* máquinas em um ambiente produtivo e *n* tarefas a serem realizadas. Cada tarefa possui um conjunto de operações a ser realizado em uma sequência específica de processamento em cada uma das referidas máquinas (Pinedo, 2008). Além disto, será considerado o caso determinístico, isto é, quando o tempo de processamento de cada operação a ser realizada por uma determinada máquina em uma determinada tarefa é conhecido e não-nulo.

Um problema do tipo *job shop* visa sequenciar as tarefas, determinando o instante de início de cada operação em cada máquina de forma a minimizar uma função objetivo, tipicamente o tempo máximo necessário para realizar o processamento completo de todas as tarefas (*makespan*). Tal sequenciamento deve ser proposto previamente ao início do processamento das tarefas. Conforme Ozguven et al. (2010), este problema pertence à categoria de problemas NP-hard, ou seja, para o caso geral não possui solução polinomial no número de máquinas e tarefas, sendo um dos problemas de otimização combinatória mais difíceis de serem resolvidos. Consequentemente não há algoritmos conhecidos que produzam uma solução ótima em um tempo polinomial em relação às variáveis recém mencionadas. Isto significa que os métodos clássicos de otimização, como por exemplo *Branch and Bound* pode ser aplicado de forma prática apenas para problemas de pequena escala.

A modelagem matemática como uma técnica para solução de problemas do tipo *job shop* teve como primeiros trabalhos publicados os de Wagner (1959) Bowman (1959) e Manne (1960). A aplicação desta técnica é necessária para a aplicação de técnicas de soluções ótimas. Conforme Phan (1997) apud *Morales* (2012) a formulação proposta por Manne (1960) é a que resulta em melhor desempenho para o problema *job shop* sendo que na formulação para programação inteira mista (modelo incluindo variáveis inteiras, binárias e contínuas) a velocidade de convergência para a solução ótima depende de três fatores importantes, em ordem decrescente de influência: número de variáveis binárias, número de restrições, número de variáveis contínuas.

Conforme Jain e Meeran (1999), um grande número de abordagens para realizar a modelagem e solução de problemas do tipo *shop shop* tem sido propostas na literatura, com diferentes graus de sucesso. Tais abordagens empregam programação matemática, regras de despacho, sistemas especialistas, algoritmos genéticos, dentre outras.

Conforme Pinedo (2008) diversos métodos heurísticos tem sido empregados com sucesso na resolução de problemas *job shop* em aplicações práticas, com destaque para Regras de despacho, *Simulated annealing*, *Tabu search* e Algoritmos genéticos.

Apesar da estrutura geral dos algoritmos mencionados no parágrafo anterior ser relativamente simples, a especificação de detalhes envolvidos na geração e seleção de soluções é aspecto crucial na performance dos mesmos sendo objeto de intensa pesquisa na atualidade. A forma como os pesquisadores comparam a performance dos algoritmos propostos é através da resolução de estudos de caso em que são especificadas configurações de tarefas e de máquinas para serem resolvidos. Conforme busca realizada pela plataforma Google Acadêmico, apenas no ano de 2017 foram localizados aproximadamente 1.500 publicações com o termo de busca "*job shop scheduling*". Tal fato demonstra a importância que a comunidade científica tem dedicado a este tema de pesquisa.

Este projeto de pesquisa tem por **objetivo geral** comparar formulações e algoritmos para obtenção de soluções ótimas e sub ótimas de problemas do tipo *job shop*.

Como **objetivos específicos**, no tocante a problemas do tipo *job shop*, destacam-se:

- identificar propriedades e aplicações de formulações matemáticas apropriadas para representação;

- identificar propriedades e aplicações de algoritmos para obtenção de soluções ótimas e sub ótimas;

- implementar algoritmos para obtenção de soluções ótimas e sub ótimas;

- especificar estudos de caso para realizar análises comparativas de formulações matemáticas e de algoritmos;

- comparar soluções obtidas com a realização de estudos de caso;

**IX. Revisão bibliográfica/literatura;**

Conforme Pinedo (2008), a indústria de manufatura pode ser organizada de diferentes formas. Os arranjos básicos são relacionados a seguir:

- *flow shop*: quando há m máquinas em série, cada tarefa deve ser processada em cada uma das máquinas e todas as tarefas possuem a mesma sequência de produção;

- *job shop*: quando há m máquinas em um ambiente, cada uma das n tarefas possui uma sequência específica de processamento em cada uma das referidas máquinas;

- *open shop*: quando há m máquinas em um ambiente, cada uma das n tarefas deve ser processada em cada uma das máquinas, sendo que o tempo de processamento pode ser nulo, e não há uma sequência de processamento pré-determinada nas máquinas; o programador pode determinar uma sequência para cada tarefa.

Ainda em Pinedo (2008) são apresentadas variações de tais arranjos os quais incorporam flexibilidade à sequência de execução de tarefas ou à alocação de máquinas para realizar operações sobre as tarefas.

Um grafo disjuntivo (Balas, 1969) permite realizar uma representação gráfica do problema *job shop* e identificar soluções para o mesmo, porém é limitado para problemas de pequena escala. Um grafo disjuntivo pode ser representado na forma G = (S, C, D), onde:

- S é um conjunto de nós (representado graficamente por círculos) tal que cada nó representa uma operação de uma determinada tarefa a ser realizada em uma determinada máquina, a cada nó é associado um peso igual ao tempo para realização da operação; além disto são incluídos outros dois nós com peso nulo, um representando o início geral da operação e outro representando a conclusão das operações;

- C é um conjunto de arcos direcionados, denominados arcos conjuntivos, os quais especificam as relações de precedência entre operações de cada tarefa; para cada tarefa deve haver um arco do nó inicial para a primeira operação da tarefa, um nó entre operações conforme a relação de precedência de tarefas, um nó da última operação para o nó final;

- D é um conjunto de arcos, inicialmente, não direcionados, denominados arcos disjuntivos, tais arcos especificam a limitação de recursos; devem haver arcos disjuntivos interligando todos os pares de nós que estão associados a operações que utilizam uma mesma máquina;

O princípio do sequenciamento da produção consiste em atribuir direções aos arcos disjuntivos, definindo a sequência de processamento das operações em cada uma das máquinas. Com isto são determinados caminhos no grafo disjuntivo, a cada caminho do nó inicial ao nó final é associado um tempo total de processamento obtido pela soma do peso dos nós que são percorridos em tal caminho. O caminho crítico é o de maior tempo de processamento e corresponde ao tempo necessário para realizar o processamento do conjunto de tarefas, na literatura este tempo é usualmente denominado *makespan*.

Conforme Roshanaei (2012), a modelagem matemática como forma de solução para a programação das tarefas de produção teve como um dos primeiros trabalhos o apresentado por Wagner (1959) seguido por Bowman (1959) e Manne (1960). Estas três formas de modelagem empregam distintos paradigmas de representação do problema, sendo usualmente denominados *position-based*, *time-based* e *sequence-based*, respectivamente, as quais foram estendidas para a representação de problemas do tipo *job shop*, sem que paradigmas distintos tenham sido propostos desde então. As formulações matemáticas para estes três paradigmas utilizam as seguintes variáveis binárias que permitem compreender simplificadamente cada um deles:

- *position-based*

Xj, i,f = 1 se a tarefa j é alocada na posição f da máquina i; 0 em caso contrário;

- *time-based*

Xj,i,t = 1 se a tarefa j é processada na máquina i durante o período t; 0 em caso contrário;

- *sequence-based*

Xj,h,i = 1 se a tarefa j sucede a tarefa h (não necessariamente de forma imediata) na máquina i; 0 em caso contrário;

Desde então pesquisadores tem adotado diferentes funções objetivo, hipóteses e perspectivas de modelagem, e especialmente técnicas para solução do problema.

O algoritmo do tipo *Branch and Bound* adaptado por Applegate e Cook (1991) pode ser utilizado para resolver de forma ótima o problema do sequenciamento de tarefas em *job shop*. Conforme os autores este algoritmo se mostrou eficiente para resolver de forma ótima um problema de sequenciamento de 10 tarefas e 10 máquinas que estava sem solução havia mais de 10 anos. O algoritmo também se mostrou eficiente para resolver diversos outros problemas propostos na literatura. Os autores destacam a importância de adotar uma boa heurística para encontrar uma solução inicial para alimentar o algoritmo. Conforme Jain e Meeran (1999), o *Branch and Bound* é a principal estratégia enumerativa de busca de solução, na qual uma árvore de soluções é iterativamente construída. Esta técnica disponibiliza regras que permitem que grande parte do espaço de soluções possa ser descartado da busca, o que reduz o esforço computacional na busca da solução ótima.

Morales (2012) destaca que a maioria dos estudos realizados sobre o problema da programação em ambiente *job shop* enfoca a geração e aplicação de métodos aproximados como heurísticas e meta-heurísticas. Heurísticas visam atingir soluções de boa qualidade, sub ótimas, mas em tempo computacional moderado. Embora não possam garantir a localização da solução ótima, são adequadas para problemas de alta complexidade, como é o caso do *job shop*.

Conforme Pinedo (2008) a aplicação de algoritmos do tipo *Branch and Bound* torna-se proibitiva para problemas com apenas 30 tarefas, requerendo tempo de solução muito elevado. Desta forma é importante adotar heurísticas que produzam um resultado minimamente satisfatório com um custo de processamento computacional razoável.

As heurísticas mais elementares são as denominadas regras de despacho. Tais regras podem ser combinadas com regras de prioridade gerando heurísticas mais elaboradas. Regras de despacho podem ser do tipo estático no tempo, sendo dependentes apenas da tarefa e da máquina. Na regra WSPT (*weighted shortest processing time first*) é atribuído um peso à cada tarefa (wj) e é determinada a razão "wj/pj", a tarefa com maior razão é processada primeiro, onde pj é o tempo de processamento da tarefa. As regras de despacho também podem ser dinâmicas, ou seja, dependentes do tempo. Por exemplo a regra MS (*minimum slack first*) ordena as tarefas determinando "max(dj - pj - t, 0)", a tarefa com o menor valor recebe prioridade sobre as demais, onde dj é o instante para entrega da tarefa e t o instante atual. Observando-se estas duas regras fica evidente que a prioridade estabelecida às tarefas com a primeira regra é constante no tempo, podendo ser avaliada uma única vez, enquanto que na segunda regra, em um determinado instante uma tarefa j1 pode ter prioridade sobre a tarefa j2 e, transcorrido algum tempo ambas podem ter a mesma prioridade. As regras podem ser locais, considerando apenas a fila em que a tarefa está aguardando ou à máquina a que a tarefa está alocada, ou globais. Regras usuais são SIRO (*service in random order*) FIFS (*first come first served*) ERD (*earliest release date first*), dentre outras. Algumas regras produzem resultados ótimos para arranjos específicos de máquinas e tarefas ou podem produzir resultados apenas satisfatórios em outros casos. Algumas regras de despacho podem produzir bons resultados com funções objetivo simples, tal como minimizar o valor de *makespan* e não se mostrarem tão eficientes quando são considerados objetivos mais complexos como a combinação de diversos objetivos mais simples. Para tais casos a composição de regras simples torna-se importante. A regra ATC (*aparente tardiness cost*) combina as regras WSPT e MS, estabelecendo um índice para cada tarefa conforme a equação apresentada na sequência, onde K é um fator de escala e pode ser determinado empiricamente e pm é a média do tempo de processamento das tarefas pendentes. Se K é nulo a regra se reduz à WSPT. Se K é muito pequeno a regra se reduz à MS quando não há tarefas atrasadas e novamente à WSPT quando há tarefas atrasadas. A equação abaixo é utilizada para determinar tal índice.

Ij = (wj/pj). exp( - max(dj-pj-t) / (K.pm))

Com a utilização das regras de despacho, simples ou combinadas, parte-se de uma solução inicial vazia, e todas as operações estão pendentes para serem alocadas. A cada passo do algoritmo todas as operações pendentes são submetidas à regra de despacho e as com melhor avaliação são selecionadas para serem executadas, tais operações deixam de estar pendentes para serem alocadas. Após a execução de diversos passos uma solução válida é obtida. Conforme Chang et al. (1996) *apud* Morales (2012) a regra SPT (*shortest processing time*), em que a tarefa com o menor tempo de processamento na máquina considerada tem prioridade, apresenta para o caso geral resultados consistentes. Conforme Shazad e Mebarki (2010) as regras de despacho combinadas produzem resultados melhores do que a regras simples.

Na sequência é apresentado uma síntese dos conceitos essenciais associados à heurísticas utilizadas no planejamento da produção em problemas do tipo *job shop*, conforme apresentado em (Pinedo, 2008).

As regras de despacho são heurísticas construtivas pois iniciam sem qualquer sequência de produção inicial e gradualmente constroem a sequência adicionando uma operação por vez. Outra estratégia de construção de sequências completamente distinta é a baseada em algoritmos de melhoria de solução. Nestes algoritmos parte-se de uma solução inicial, que pode ser obtida aleatoriamente ou mesmo através da aplicação de regras de despacho, e gradualmente busca-se uma nova solução que melhore o valor da função objetivo em relação à solução anterior. Um tipo importante de algoritmo é o de busca local. Conforme o algoritmo é executado novas soluções vão sendo geradas em torno de uma "vizinhança" em relação à solução anterior. Duas soluções são vizinhas se uma pode ser obtida através de uma modificação bem definida em relação à outra. As novas soluções são comparadas com a anterior e, ou são aceitas ou são rejeitadas conforme um critério de aceite/descarte específico. Um aspecto crucial na construção de um algoritmo deste tipo é a definição da vizinhança. Estratégias de definição da vizinhança baseadas na inversão de operações a serem executadas em subconjuntos de máquina e tarefas no caminho crítico do grafo disjuntivo são denominadas *One step* (*multi-step*) *look-back interchanges*. Diferentes critérios de seleção de máquinas e tarefas a serem invertidas podem ser selecionados.

Dois algoritmos que adotam esta estratégia são o *Simulated annealing* e o *Tabu search*. A diferença básica entre eles reside no critério de aceite/descarte de novas soluções. O primeiro adota um procedimento probabilístico e o segundo um procedimento determinístico. Nos dois algoritmos uma nova solução não precisa ser necessariamente melhor do que a anterior. A justificativa para tanto é permitir que o procedimento evolua para longe de um mínimo local e encontre uma solução melhor. No algoritmo *Tabu search* é criada uma lista de tamanho pré-estabelecido, ou mesmo uma árvore, com mutações proibidas de serem realizadas, por exemplo um par de tarefas cuja sequência não pode ser invertida. A cada nova solução o reverso da alteração realizada da solução anterior para a seleção atual é incluído no início da lista e os demais elementos são deslocados para as próximas posições, sendo que o último elemento é excluído da lista. Isto é realizado para evitar que um mínimo local seja novamente alcançado.

Os algoritmos genéticos também são algoritmos de busca local. A cada iteração há um número finito, tipicamente constante, de soluções, que sobreviveram de iterações anteriores bem como soluções que podem ter sido geradas por reprodução ou mutação de soluções de iterações anteriores. Este conjunto de soluções é denominado população da iteração. Soluções obtidas por reprodução adotam combinações de segmentos de soluções anteriores (*cross over operator*). Os algoritmos *Simulated annealing* e *Tabu search* podem ser considerados como um caso particular do algoritmo genético em que o tamanho da população em cada iteração é igual a um.

Na aplicação de algoritmos genéticos, uma forma usual de representação de uma sequência de operações é a "baseada em operação". Supondo um arranjo de 3 tarefas e 5 máquinas, uma possível forma de representação de uma sequência é como a apresentada abaixo. Nesta representação, o número "3" na quarta posição (da esquerda para a direita) representa a primeira operação da tarefa 3 enquanto que o número "2" na sexta posição representa a terceira operação da tarefa 2. Nesta sequência o número de cada tarefa aparece tantas vezes quanto for o número de máquinas (Ranjini e Zoraida,2013).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 2 | 1 | 3 | 1 | 2 | 2 | 3 | 1 | 3 | 3 | 2 | 1 | 3 | 1 |

Conforme Pinedo (2008), a aplicação prática destes conceitos pode ser realizada conforme as três fases relacionadas abaixo.

fase 1) inicialmente são determinadas informações gerais sobre as tarefas, tal como data de conclusão da tarefa (*due date*) , tempo de processamento da operação, dentre outras;

fase 2) com base nas informações determinadas na fase 1, são determinados fatores de escala para a composição de regras de despacho (similar ao parâmetro K na regra ATC) e uma solução inicial é gerada a partir das regras de despacho selecionadas;

fase 3) a solução inicial obtida na fase 2 é utilizada para alimentar um algoritmo estruturado com base no conceito do *Simulated annealing* ou no *Tabu search*, um número pré-determinado de iterações do algoritmo é executado.

Os algoritmos mencionados acima são relativamente simples de serem implementados, a título de ilustração apresenta-se o algoritmo *Simulated annealing* apresentado em Pinedo (2008). Neste algoritmo G(S) corresponde ao valor da função objetivo, possivelmente o valor de *makespan* para a sequência S. Neste algoritmo o critério de parada é o número de iterações, mas poderia ser substituído por uma comparação do valor da função objetivo com uma meta pré-estabelecida. Verifica-se que S0 corresponde à melhor sequência obtida até a iteração k, Sc é a sequência candidata e Sk é a sequência da iteração k, βk é um valor inteiro que influencia a probabilidade de aceitar a sequência candidata ou descartá-la. Quanto maior o valor de k (mais iterações foram realizadas) menor o valor de βk (possivelmente βk = ak com a ∈ [0,1]) e portanto menor será a probabilidade de aceitar uma sequência (Sk) que não seja melhor do que a sequência candidata.

Step 1

Set k = 1 and select β1

Select an initial sequence S1 using some heuristic

Set S0 = S1.

Step 2

Select a candidate schedule Sc from the neighbourhood of Sk.

If G(S0) < G(Sc) < G(Sk), set Sk+1 = Sc and go to Step 3.

If G(Sc) < G(S0), set S0 = Sk+1 = Sc and go to Step 3.

If G(Sc) > G(Sk) generate a random number Uk from a Uniform(0,1)

distribution;

P(Sk, Sc) = exp( (G(Sk, G(Sc))/ βk )

If Uk ≤ P(Sk, Sc) set Sk+1 = Sc otherwise set Sk+1 = Sk and go to Step 3.

Step 3

Select βk+1 ≤ βk.

Increment k by 1.

If k = N then STOP, otherwise go to Step 2.

Ziaee (2014) apresenta um algoritmo de heurísticas construtivas com o objetivo de minimizar o *makespan*. O autor afirma que este algoritmo pode ser utilizado para obter uma solução inicial factível para alimentar os algoritmos de melhoria de solução. Neste algoritmo TC é uma soma ponderada de algum critério que afeta a função objetivo. A minimização de TC no processo resulta na melhora da solução obtida.

until all operations of all jobs are scheduled, repeat

Find i, j that minimizes TC such that:

1. j=1 or (j-1)th operation of job i is already scheduled, and

2. jth operation of job i is unscheduled.

Schedule jth operation of job i on the last position of current partial sequence on machine k (the machine capable of processing operation j of job i).

Conforme Seda (2007) o ponto crucial no sucesso dos algoritmos *Simulated annealing* e *Tabu search* é o processo de definição da vizinhança para seleção de novos candidatos. Tais autores destacam que a literatura apresenta diversas estratégias sofisticadas tais como as estratégias definidas em Nowicki e Smutnicki (1996) e Watson et al. (2006).

Conforme Ranjini e Zoraida (2013) a literatura propõem diversos operadores para obtenção de novas soluções, e destacam os seguintes *Order Preserving Multipoint, Ordered Partially Mapped Crossover*, sendo que o operador *Unordered Subsequence Exchange Crossover* produziu resultados muito melhores que os anteriores em análises comparativas de performance de algoritmos. Outro aspecto importante nos algoritmos genéticos é o processo de seleção de novas soluções. Os autores utilizaram três processos distintos denominados, *Stochastic Universal Sampling, Roulette Whell Selection*, *Rank Based Roulet Whell Selection* e *Binary Tournament Selection*. A população inicial, apesar de poder ser obtida de forma aleatória, influencia na convergência da solução. O tamanho da população também é um fator importante na especificação do algoritmo.

Conforme Jaim e Meeran (1999) a aplicação dos algoritmos mencionados acima pode ser considerada uma arte tendo em vista as inúmeras decisões relativas a especificações de parâmetros e procedimentos as quais não são triviais e que podem gerar resultados insatisfatórios se realizados na base da tentativa e erro. Porém ainda assim válidas pois, conforme mencionado anteriormente, técnicas com *Branch and Bound* são limitadas a instâncias de problemas de reduzidas dimensões. Além disto, a qualidade da solução obtida com a aplicação do algoritmo *Tabu search* é, no caso geral, comparável à solução obtida com o *Branch and Bound* e o tempo de processamento pode ser até 80 vezes inferior.

Na literatura os pesquisadores realizam análise comparativa de performance dos algoritmos propostos através da resolução de estudos de caso (*benchmark*). Jaim e Meeran (1999) afirmam que através de tais resoluções é possível realizar afirmações do tipo "algoritmo A alcança *makespan* X em tempo Y com o problema B". Visto que os problemas em tais estudos de caso tem dimensões e graus de dificuldade distintos, é possível determinar as potencialidades e limitações de um determinado método realizando sua avaliação em diferentes problemas. Tais estudos de caso são elaborados por diversos autores e denominados em função dos mesmos. Um dos problemas mais utilizados são os propostos por Fisher e Thompson (1963), por exemplo o problema FT10 é constituído por 10 tarefas a serem processadas em 10 máquinas, cada tarefa constituída por 10 operações, sendo que o tempo de processamento de cada operação é determinado de forma aleatória no intervalo [1,99], a sequência das operações em cada tarefa também é determinada de forma aleatória. O objetivo deste problema é minimizar o *makespan*. Uma possível instância deste problema é a apresentada abaixo, onde "jobx" identifica a tarefa, cada par (Y Z) está associado a uma operação, onde Y informa a máquina em que a operação deve ser realizada e Z informa o tempo de processamento.

job1: 0 29 ; 1 78 ; 2 9 ; 3 36 ; 4 49 ; 5 11 ; 6 62 ; 7 56 ; 8 44 ; 9 21

job2: 0 43 ; 2 90 ; 4 75 ; 9 11 ; 3 69 ; 1 28 ; 6 46 ; 5 46 ; 7 72 ; 8 30

job3: 1 91 ; 0 85 ; 3 39 ; 2 74 ; 8 90 ; 5 10 ; 7 12 ; 6 89 ; 9 45 ; 4 33

job4: 1 81 ; 2 95 ; 0 71 ; 4 99 ; 6 9 ; 8 52 ; 7 85 ; 3 98 ; 9 22 ; 5 43

job5: 2 14 ; 0 6 ; 1 22 ; 5 61 ; 3 26 ; 4 69 ; 8 21 ; 7 49 ; 9 72 ; 6 53

job6: 2 84 ; 1 2 ; 5 52 ; 3 95 ; 8 48 ; 9 72 ; 0 47 ; 6 65 ; 4 6 ; 7 25

job7: 1 46 ; 0 37 ; 3 61 ; 2 13 ; 6 32 ; 5 21 ; 9 32 ; 8 89 ; 7 30 ; 4 55

job8: 2 31 ; 0 86 ; 1 46 ; 5 74 ; 4 32 ; 6 88 ; 8 19 ; 9 48 ; 7 36 ; 3 79

job9: 0 76 ; 1 69 ; 3 76 ; 5 51 ; 2 85 ; 9 11 ; 6 40 ; 7 89 ; 4 26 ; 8 74

job10: 1 85 ; 0 13 ; 2 61 ; 6 7 ; 8 64 ; 9 76 ; 5 47 ; 3 52 ; 4 90 ; 7 45

Grupo de pesquisadores vinculados à Stern School of Business e liderados por Michael Pinedo desenvolveu um sistema de programação das tarefas de produção denominado LEKIN (LEKIN, 2018). Este sistema é distribuído gratuitamente pelos pesquisadores e permite a solução de diversas categorias de problemas no âmbito da programação das tarefas de produção. São disponibilizadas diversas heurísticas simples (regras de despaho) denominadas na literatura ATCS (*Apparent Tardiness Cost*), EDD (*Earliest Due Date*), MS (*Minimum Slack*), FCFS (*First Come – First Served*), LPT (*Longest Processing Time first*), SPT (*Shortest Processing Time first*), WSPT (*Weighted Shortest Processing Time first*), CR (*Critical Ratio*). Também são disponibilizadas diversas variações da heurística *Shifting Bottleneck*. Uma peculiaridade deste sistema é que ele permite a importação de novas estratégias de solução desenvolvidas pelo usuário, o que o torna conveniente para a avaliação de tais estratégias, pois permite a comparação do resultado com estratégias consolidadas na literatura implementadas em algoritmos plenamente validados. Isto também minimiza o esforço de programação pois a interface de entrada de dados do sistema a ser resolvido e a interface de apresentação visual dos resultados já está implementada. Para utilização deste recurso é necessário o domínio do uso do sistema com as funcionalidades já disponibilizadas bem como o domínio no desenvolvimento da interface entre o algoritmo implementado pelo usuário e o próprio sistema LEKIN.

**X. Materiais e métodos/Metodologia:**

Este projeto de pesquisa será desenvolvido através das seguintes atividades:

- revisão bibliográfica em livros e artigos de congressos e periódicos;

- análise de algoritmos disponíveis nas fontes anteriores bem como em repositórios da comunidade científica;

- implementação de algoritmos em linguagens de programação;

- análise de aplicativos computacionais utilizados pela comunidade acadêmica e industrial;

Tais atividades serão realizadas em computadores pessoais dos pesquisadores envolvidos ou de laboratórios de informática da UFPR.

Este projeto de pesquisa não envolve a participação de voluntários humanos ou animais, desta forma considera-se que não é necessária aprovação por Comitês de Ética em Pesquisa.

Para alcançar os objetivos propostos o presente projeto de pesquisa será desenvolvido através da realização das seguintes etapas:

## 1) Seleção e solução de casos de dimensões reduzidas

Partindo-se do conhecimento prévio do autor deste projeto de pesquisa, será realizada a seleção de casos de reduzidas dimensões com objetivo de minimizar o *makespan*. Tais casos poderão ser selecionados da plataforma OR Library (http://people.brunel.ac.uk/~mastjjb/jeb/info.html) ou outras plataformas. Os casos selecionados serão formulados matematicamente em conformidade com o proposto na literatura. Será realizada a seleção e implementação de algoritmos para obtenção de soluções ótimas e sub ótimas para problemas *job shop*. Será realizada análise comparativa da performance dos algoritmos selecionados na resolução dos casos selecionados.

Com a execução desta etapa será redigido artigo científico e realizada a preparação de material didático que poderá ser utilizado em disciplinas de graduação e pós-graduação já no início do ano de 2018.

Para realização desta etapa prevê-se a realização das seguintes atividades:

1.1 - seleção de casos para estudo e de formulações matemáticas a serem aplicadas;

1.2 - formulação matemática dos casos selecionados;

1.3 - seleção e adequação ou implementação da algoritmos;

1.4 - resolução dos casos selecionados;

1.5 - análise de resultados;

1.6 - redação de artigo científico;

1.7 - redação de material didático.

## 2) Consolidação e expansão da revisão bibliográfica referente às formulações matemáticas apresentadas na literatura e aplicação em estudos de caso selecionados.

Esta atividade será realizada através da leitura de livros e artigos científicos. Para tanto será realizada busca em bases de dados incluindo os termos relacionados abaixo:

- job shop schedulling; mathematical formulation; representation scheme; integer programming; mixed integer programming; makespan;

Durante esta etapa também será realizada a seleção de novos estudos de caso com complexidade superior aos dos selecionados na etapa 1. Tais casos serão formulados matematicamente em conformidade com o proposto na literatura.

Com a realização desta atividade espera-se relacionar as principais formulações matemáticas adotadas para a representação de problemas do tipo *job shop*, especificando suas propriedades e limitações. Será dedicada atenção aos parâmetros influentes na velocidade de convergência para a solução ótima, a saber número de variáveis binárias, número de restrições, número de variáveis contínuas.

Como resultado será realizada a redação de artigo científico relatando os resultados alcançados nesta etapa. Será revista e expandida a redação de material didático.

Para realização desta etapa prevê-se a realização das seguintes atividades:

2.1 - refinamento dos termos de busca em bases científicas;

2.2 - análise preliminar de artigos científicos e seleção de artigos relevantes;

2.3 - análise de artigos relevantes e seleção de estudos de caso;

2.4 - aplicação de formulações selecionadas para a representação de estudos de caso selecionados;

2.5 - síntese dos resultados identificados nos artigos selecionados;

2.6 - redação de artigo científico;

2.7 - revisão da redação de material didático;

## 3) Consolidação da revisão bibliográfica referente aos algoritmos para obtenção de soluções ótimas e aplicação em estudos de caso selecionados.

Esta atividade será realizada através da leitura de livros e artigos científicos. Para tanto será realizada busca em bases de dados incluindo os termos relacionados abaixo:

- job shop schedulling; mathematical formulation; representation scheme; integer programming; mixed integer programming; makespan; algorithm

Durante esta etapa também será realizada a seleção, analise, adaptação e implementação de algoritmos para obtenção de soluções ótimas. Tais algoritmos serão utilizados na resolução do estudos de caso selecionados na etapa 2. A princípio os algoritmos serão adaptados e ou implementados no aplicativo Matlab. Possivelmente será utilizado o pacote de extensão denominado TORSCHE Scheduling toolbox for Matlab (Sucha et al., 2006), dentre outros.

Com a realização desta atividade espera-se relacionar as principais variações de algoritmos para obtenção de soluções ótimas, em particular soluções de descarte de ramos na árvore de soluções geradas pelo algoritmo *Branch and Bound*. Para consolidação dos resultados serão selecionados, analisados e adaptados, conforme a necessidade, códigos propostos pela comunidade científica. Tais códigos serão avaliados com a solução de estudos de caso.

Como resultado será realizada a redação de artigo científico relatando os resultados alcançados nesta etapa. Será revista e expandida a redação de material didático.

Para realização desta etapa prevê-se a realização das seguintes atividades:

3.1 - refinamento dos termos de busca em bases científicas;

3.2 - análise preliminar de artigos científicos e seleção de artigos relevantes;

3.3 - análise de artigos relevantes e seleção de estudos de caso;

3.4 - seleção, analise, adaptação e implementação de algoritmos selecionados;

3.5 - resolução de estudos de caso selecionados;

3.6 - síntese dos resultados identificados nos artigos selecionados e resolução de casos selecionados;

3.7 - redação de artigo científico;

3.8 - revisão da redação de material didático.

## 4) Consolidação da revisão bibliográfica referente aos algoritmos para obtenção de soluções sub ótimas e aplicação em estudos de caso selecionados.

Esta atividade será realizada através da leitura de livros e artigos científicos. Para tanto será realizada busca em bases de dados com os termos relacionados abaixo, dentre outros.

- job shop schedulling; heuristic; simulated annealing; tabu search, genetic algorithm; dispaching rule; reproduction; mutation; size population; initial population; makespan;

Durante esta etapa também será realizada a seleção, analise, adaptação e implementação de algoritmos para obtenção de soluções sub ótimas. Tais algoritmos serão utilizados na resolução do estudos de caso selecionados na etapa 2.

Com a realização desta atividade espera-se relacionar as principais variações de algoritmos para obtenção de soluções sub ótimas, em particular soluções para geração da população inicial com regras de despacho, definição de vizinhança associada aos algoritmos simulated annealing e tabu search, operações de reprodução e mutação em algoritmos genéticos, tamanho da população.

Como resultado será realizada a redação de artigo científico relatando os resultados alcançados nesta etapa. Será revista e expandida a redação de material didático.

Para realização desta etapa prevê-se a realização das seguintes atividades:

4.1 - refinamento dos termos de busca em bases científicas;

4.2 - análise preliminar de artigos científicos e seleção de artigos relevantes;

4.3 - análise de artigos relevantes e seleção de estudos de caso;

4.4 - seleção, analise, adaptação e implementação de algoritmos selecionados;

4.5 - resolução de estudos de caso selecionados;

4.6 - síntese dos resultados identificados nos artigos selecionados e resolução de casos selecionados;

4.7 - redação de artigo científico;

4.8 - revisão da redação de material didático.

## 5) Análise comparativa dos resultados obtidos com a aplicação de algoritmos para obtenção de soluções ótimas e sub ótimas

Esta atividade será realizada através da análise crítica comparativa dos resultados obtidos com a execução das etapas 3 e 4.

Como resultado será realizada a redação de artigo científico relatando os resultados alcançados nesta etapa.

Para realização desta etapa prevê-se a realização das seguintes atividades:

5.1 - análise comparativa de resultados;

5.2 - redação de artigo científico;

# **Cronograma preliminar**

A tabela abaixo apresenta o cronograma preliminar de trabalho considerando uma dedicação média de 24 horas semanais ao desenvolvimento do projeto. Como se trata de um projeto de pesquisa pode-se vislumbrar a necessidade de revisões na execução do mesmo em função do redirecionamento para outras atividades ou pela inconsistência na previsão do tempo necessário para realização das atividades. O projeto foi concebido considerando um período de 36 meses.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cronograma | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ano | | 2018 | | | | | | | | | | | | 2019 | | | | | | | | | | | | 2020 | | | | | | | | | | | |
| Etapa | Atividade | janeiro | fevereiro | março | abril | maio | junho | julho | agosto | setembro | outubro | novembro | dezembro | janeiro | fevereiro | março | abril | maio | junho | julho | agosto | setembro | outubro | novembro | dezembro | janeiro | fevereiro | março | abril | maio | junho | julho | agosto | setembro | outubro | novembro | dezembro |
| 1 | 1 | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 2 |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 3 |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 4 |  |  |  | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 5 |  |  |  | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 6 |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 7 |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | 1 |  |  |  |  |  |  |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 2 |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 3 |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 4 |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |
|  | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |  |  |  |  |  |  |  |  |  |
|  | 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |  |  |  |  |  |  |  |  |
|  | 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |
|  | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |  |
|  | 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ | √ |  |  |  |  |  |  |
|  | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ | √ | √ |  |  |  |  |
|  | 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ | √ | √ |  |  |  |
|  | 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |
|  | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ | √ |
|  | 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |
|  | 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | √ | √ |

**XI. Referências:**

Applegate, D., and Cook, W., (1991), “A computational study of the job-shop scheduling problem”, ORSA Journal on Computing 3/2, Spring, 149-156.

Balas, E., (1969), “Machine scheduling via disjunctive graphs: An implicit enumeration algorithm”, Operations Research 17, 941-957.

Baker, K. R., Keller, B. (2010). Solving the single-machine sequencing problem using integer programming. *Computers & Industrial Engineering*, *59*(4), 730-735.

Baruwa, O. T., Piera, M. A., Guasch, A. (2015). Deadlock-free scheduling method for flexible manufacturing systems based on timed colored Petri nets and anytime heuristic search. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *45*(5), 831-846.

Bowman, E. H., (1959), “The schedule-sequencing problem”, Operations Research 7, 621- 624.

Chang, Y. L., Sueyoshi, T., Sullivan, R. S. (1996). Ranking dispatching rules by data envelopment analysis in a job shop environment. *IIE transactions*, *28*(8), 631-642.

Fisher, H., Thompson, G. L., (1963), “Probabilistic learning combinations of local job shop scheduling rules”, in: J. F. Muth and G. L. Thompson, (eds.), Industrial Scheduling, Prentice Hall, Englewood Cliffs, New Jersey, 225-251.

Jain, A. S., Meeran, S. (1999). Deterministic job-shop scheduling: Past, present and future. *European journal of operational research*, *113*(2), 390-434.

Kammoun, M. A., Ezzeddine, W., Rezg, N., Achour, Z. (2017). FMS Scheduling under Availability Constraint with Supervisor Based on Timed Petri Nets. *Applied Sciences*, *7*(4), 399.

LEKIN (2018) http://web-static.stern.nyu.edu/om/software/lekin/

Li, C., Wu, W., Feng, Y., Rong, G. (2015). Scheduling FMS problems with heuristic search function and transition-timed Petri nets. *Journal of Intelligent Manufacturing*, *26*(5), 933-944.

Li, Z., Wu, N., Zhou, M. (2012). Deadlock control of automated manufacturing systems based on Petri nets — A literature review. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), 437-462.

Lin, L., Shehabinia, A. R., Su, R., Brandin, B. (2014). Automaton-based timed supervisory control for operational planning and scheduling under multiple job deadlines. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on* (pp. 5943-5948). IEEE.

Morales, G.S.W. (2012). Formulações matemáticas e estratégias de resolução para o problema Job Shop Clássico. Dissertação. Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

Mejía, G., Niño, K., Montoya, C., Sánchez, M. A., Palacios, J., Amodeo, L. (2016). A Petri Net-based framework for realistic project management and scheduling: An application in animation and videogames. *Computers & Operations Research*, *66*, 190-198.

Nowicki, E., Smutnicki, C. (1996). A fast taboo search algorithm for the job shop problem. *Management science*, *42*(6), 797-813.

Özgüven, C., Özbakır, L., Yavuz, Y. (2010). Mathematical models for job-shop scheduling problems with routing and process plan flexibility. *Applied Mathematical Modelling*, 34(6), 1539-1548.

Park, J., Reveliotis, S. A. (2001). Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings. *IEEE Transactions on Automatic Control*, *46*(10), 1572-1583.

Pham DN (2008) Complex job shop scheduling: Formulations, algorithms and a healthcare application. PhD thesis, University of Fribourg, Switzerland

Pinedo, M. (2012). *Scheduling:* Theory, Algorithms, and Systems New York: Springer.

Ranjini, A., Zoraida, B. (2013). Analysis of selection schemes for solving job shop scheduling problem using genetic algorithm. *International Journal of Research in Engineering and Technology*, *2*(11), 2319-1163.

Šeda, M. (2007). Mathematical models of flow shop and job shop scheduling problems. *World Academy of Science, Engineering and Technology*, *1*(31), 122-127.

Shahzad, A., Mebarki, N., IRCCyN, I. (2010). Discovering dispatching rules for job shop scheduling problem through data mining. In *8th International Conference of Modeling and Simulation-MOSIM* (pp. 10-12).

Shehabinia, A. R., Lin, L., Su, R. (2016). Timed Supervisory Control for Operational Planning and Scheduling under Multiple Job Deadlines. *arXiv preprint arXiv:1607.04255*.

Sucha, P., Kutil, M., Sojka, M., Hanzálek, Z. (2006). Torsche scheduling toolbox for matlab. In Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE (pp. 1181-1186). IEEE.

Vamsikrishna, C., Padmanabhan, G. Role of Petri Nets in Flexible Manufacturing System–A Review (2016). International Journal of Engineering Trends and Technology, 41(2), 90-100.

Vilela, J. N., Pena, P. N. (2016). Supervisor abstraction to deal with planning problems in manufacturing systems. In *Discrete Event Systems (WODES), 2016 13th International Workshop on,* 117-122.

Ziaee, M. (2014). Job shop scheduling with makespan objective: A heuristic approach. *International Journal of Industrial Engineering Computations*, *5*(2), 273-280.

Wagner, H. M., (1959), “An integer programming model for machine scheduling”, Naval Research Logistics Quarterly 6, 131-140.

Wang, Q., Wang, Z. (2012). Hybrid heuristic search based on Petri net for FMS scheduling. *Energy Procedia*, *17*, 506-512.

Watson, J. P., Howe, A. E., Whitley, L. D. (2006). Deconstructing Nowicki and Smutnicki's i-TSAB tabu search algorithm for the job-shop scheduling problem. *Computers & Operations Research*, *33*(9), 2623-2644.