Research, Society and Development, v. 11, n. 5, e17111528035, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i5.28035

Estudo sobre as operações críticas no agendamento de tarefas em sistemas produtivos do tipo job shop

Study on critical operations in job shop scheduling in productive systems

Estudio sobre operaciones críticas en las tareas de programación en sistemas productivos de tipo job shop

Recebido: 14/03/2022 | Revisado: 21/03/2022 | Aceito: 26/03/2022 | Publicado: 02/04/2022

Jonathan Farias Bordignon
ORCID: https://orcid.org/0000-0001-8971-7590
Universidade Nove de Julho, Brasil
E-mail: jonathanbordignon@uninove.br
Luis Carlos Caparroz dos Santos
ORCID: https://orcid.org/0000-0001-8847-5200
Universidade Nove de Julho, Brasil
E-mail: luissantos@uni9.edu.br
Marilda Fatima de Souza da Silva
ORCID: https://orcid.org/0000-0002-0286-3358
Universidade Nove de Julho, Brasil
E-mail: Marilda.silva@uni9.edu.br
Fabio Henrique Pereira
ORCID: https://orcid.org/0000-0002-6075-5566
Universidade Nove de Julho, Brasil
E-mail: fabiohp@uni9.pro.br

Resumo

Este trabalho estuda o problema clássico de agendamento da produção em job shop para minimização do makespan. Devido à natureza combinatória e complexidade computacional desse problema, o uso de técnicas metaheurísticas aliadas a métodos de busca local é bastante difundido por possibilitar resultados satisfatórios em um tempo computacional viável. Em geral, os métodos de busca local se baseiam em permutações empíricas das operações que compõe o caminho crítico de uma solução, as chamadas operações críticas, o que demanda o cálculo do caminho crítico para cada uma das soluções geradas no processo de busca. Além de elevar o custo computacional da busca local, tal abordagem promove permutações de operações que não resultam em qualquer melhoria da solução. Este trabalho investiga a distribuição das operações críticas nas máquinas e a correlação entre essa distribuição e características estatísticas dos problemas. O objetivo é estimar máquinas que concentram operações críticas e identificar características que possam contribuir para definição de métodos de busca local que não dependam do cálculo do caminho crítico a cada solução. Experimentos computacionais com instâncias usuais da literatura mostram que há uma concentração de operações críticas em algumas máquinas e, em alguns casos, uma correlação positiva significativa entre essa concentração e os tempos médios de processamento das operações, o que pode fornecer subsídios para criação de métodos de busca local computacionalmente mais eficientes.

Palavras-chave: Busca local; Caminho crítico; Scheduling; Manufatura; Planejamento do processo.

Abstract

This work studies the classic problem of job shop scheduling for makespan minimizing. Due to the combinatorial nature and computational complexity of this problem, the use of metaheuristic techniques combined with local search methods is quite widespread as it allows satisfactory results in a viable computational time. In general, local search methods are based on empirical permutations of the operations that make up the critical path of a solution, the so-called critical operations, which demand the calculation of the critical path for each of the solutions generated in the search process. In addition to increasing the computational cost of local search, this approach promotes permutations of operations that do not result in any improvement in the solution. This work investigates the distribution of critical operations on the machines and the correlation between this distribution and problem characteristics. The objective is to estimate machines that concentrate critical operations and identify characteristics that contribute to the definition of local search methods that do not depend on the calculation of the critical path for each solution. Computational experiments with usual instances from literature show that there is a concentration of critical operations in some machines and, in some cases, a positive correlation between this concentration and the average processing times of the operations, which can provide subsidies for creating more efficient local search methods.

Keywords: Local search; Critical path; Scheduling; Manufacturing; Process plan.

Resumen
Este trabajo estudia el problema de programación de la producción clásica en la tienda de empleo para minimizar a Makespan. Debido a la naturaleza combinativa y la complejidad computacional de este problema, el uso de técnicas metaheurísticas aliadas a los métodos de búsqueda locales está generalizada al permitir los resultados satisfactorios en un tiempo computacional viable. En general, los métodos de búsqueda locales se basan en permutaciones empíricas de operaciones que conforman la ruta crítica de una solución, llama a operaciones críticas, que exige el cálculo del camino crítico para cada una de las soluciones generadas en el proceso de búsqueda. Además de elevar el costo computacional de la búsqueda local, dicho enfoque promueve permutaciones de operaciones que no resultan en ninguna mejora en la solución. Este trabajo investiga la distribución de operaciones críticas sobre máquinas y correlación entre esta distribución y características estadísticas de los problemas. El objetivo es estimar las máquinas que concentren las operaciones críticas e identifican las características que pueden contribuir a la definición de métodos de búsqueda locales que no dependen del cálculo del camino crítico a cada solución. Los experimentos computacionales con instancias habituales de literatura muestran que hay una concentración de operaciones críticas en algunas máquinas y, en algunos casos, una correlación positiva significativa entre esta concentración y los tiempos de procesamiento de operaciones promedio, lo que puede proporcionar subsidios para crear métodos de búsqueda computacionalmente más eficientes.
Palabras clave: Búsqueda de ubicación; Ruta crítica; Programación; Fabricación; Planificación de processos.

1. Introdução

O agendamento da produção é uma importante etapa do Planejamento e Controle da Produção (PCP) que visa a organização do sistema de produção para otimizar algum objetivo de eficiência operacional. Essa etapa pode ser classificada como um problema de tomadas de decisão com considerável impacto em diversos setores econômicos da sociedade. Pode-se citar como exemplo indústrias de manufatura, serviços, computação em nuvem, internet das coisas, em especial no contexto da indústria 4.0 (Bueno et al., 2020; Parente *et al*., 2020; Zhang, *et al*., 2019; Li *et al*. 2016; Liu *et al*. 2016; Abdullahi *et al*. 2016; Kong *et al*. 2016; Freitag & Hildebrandt, 2016). A solução eficiente desses problemas pode contribuir significativamente com os tomadores de decisão, ajudando-os a otimizar a utilização de recursos, melhorar o índice de eficiência de execução das tarefas e, consequentemente, aumentar os ganhos econômicos (Lu *et al*., 2018). Essa importância é destaque em ambientes de produção do tipo *job shop*, os quais são caracterizados pela manufatura de um elevado número de diferentes produtos, usualmente em pequenas quantidades, com baixa padronização e especificações predefinidas pelo cliente (Wang *et al*., 2020; Wu et al., 2020; Burgy & Bulbul, 2018; Kuhpfahl & Bierwirth, 2016; Amirghasemi & Zamani, 2015). Nesse contexto, o problema é conhecido como *Job Shop Scheduling Problem* (JSSP) e é reconhecido não apenas como um problema de otimização combinatória com complexidade *NP–hard*, mas como um dos problemas mais difíceis de resolução para essa categoria (Çalis & Bulkan, 2015; Zhang et al., 2013; Zhang et al., 2013B; Lenstra et al., 1977).

De maneira formal, o JSSP pode ser definido como o agendamento do processamento de um conjunto de *n jobs* $\{J_j\}_{(1\leq j\leq n)}$ em um conjunto de *m* máquinas $\{M_r\}_{(1\leq r\leq m)}$. O processamento de um *job* por uma máquina é chamado de operação, e deve respeitar as seguintes características (Yamada, 2003): todas as máquinas estão disponíveis no instante $t_0 = 0$; cada *job* deve visitar as máquinas seguindo uma ordem predefinida, chamada de sequência tecnológica e única para cada *job*; cada máquina pode processar somente um *job* por vez e uma vez que o processamento de um *job* é iniciado, ele não pode ser interrompido. Também não há qualquer relação de precedência entre as operações de diferentes *jobs*.

A resolução do JSSP consiste em determinar uma ordem para o processamento das operações dos *jobs* em cada uma das máquinas de forma a minimizar uma ou mais medidas de desempenho. São exemplo comuns de medidas de desempenho nesses problemas o tempo total de atraso (Muminu & Aderemi, 2016, Kuhpfahl & BIerwirth, 2016), número de tarefas atrasadas (Adamu & Adewumi, 2016; Mattfeld & Bierwirth, 2004), custos de atraso, de estoque, e custos de *setup* (Chan *et al*., 2008). No entanto, a minimização do tempo máximo necessário para processar todos os *jobs*, conhecido como *makespan*, é o objetivo da maioria dos trabalhos (Wu et al., 2020; Pongchairerks, 2019, Pinedo, 2016).

Por se tratar de um problema complexo, os métodos de buscas locais no qual se baseiam em um conceito de vizinhança para refinar soluções por meio da avaliação de soluções vizinhas, tem sido naturalmente escolhido em abordagens distintas (Aarts & Lenstra, 2003). Entretanto, os métodos de buscas locais apresentam um elevado custo computacional e seu desempenho é dependente da estrutura de vizinhança adotada e dos movimentos de trocas de operações nessas vizinhanças (Jain et al., 2000).

Em geral, buscas locais se baseiam em permutações empíricas nas operações que compõe o caminho crítico de uma solução, as chamadas operações críticas, o que demanda o cálculo do caminho crítico para cada uma das soluções geradas no processo de busca. Grande parte de custo computacional deve-se a necessidade de calcular o caminho crítico de todas as soluções geradas pelo método de solução e por promover permutações de operações que não resultam em qualquer melhoria da solução. (Cruz-Chavez, 2015).

Este artigo aborda o JSSP com propósito de minimizar o *makespan* e investigar a distribuição das operações críticas nas máquinas e a correlação entre essa distribuição e características estatísticas dos problemas. O objetivo é estimar máquinas que concentram operações críticas e identificar características que possam contribuir para definição de métodos de buscas locais que não dependam do cálculo do caminho crítico a cada solução. O artigo está estruturado da seguinte maneira: o restante desta seção apresenta uma revisão da literatura relevante sobre o tema, determinando as lacunas e destacando as contribuições desta pesquisa. Na seção 2 apresenta-se o referencial teórico sobre os conceitos básicos do JSSP como grafo disjuntivo, caminho crítico, busca local e estruturas de vizinhança, e o algoritmo genético, respectivamente. Os materiais, métodos e procedimentos são descritos na seção 3 e os resultados e discussões, na seção 4. Por fim, as conclusões na seção 5 e as referências finalizam o texto.

1.1 Revisão da literatura

Devido à natureza combinatória do JSSP, o espaço de busca cresce exponencialmente à medida que aumentam os números de *jobs* e máquinas (Pinedo, 2016). Assim, metaheurísticas passam e ser a escolha natural, embora não garantam encontrar a solução ótima do problema, são capazes de encontrar soluções de qualidade próxima do ótimo (ou mesmo ótimas) com um custo computacional aceitável para aplicações do mundo real (Zhang *et al*., 2019; Tamssaouet *et al*., 2018). A despeito de sua complexidade, métodos exatos podem ser aplicados com sucesso em instâncias de problemas de pequena dimensão, não obstante para instâncias de problemas de maior dimensão tais métodos não são capazes, comumente, de encontrar soluções de qualidade próxima do ótimo em tempo computacional viável para aplicações reais da indústria (Fuchigami et al., 2017; Mahmud *et al*., 2021).

Em geral é necessário a aplicação de técnicas de refinamento de soluções como, por exemplo, métodos de buscas locais, associados às metaheurísticas. Destacam-se as técnicas de buscas locais baseadas na movimentação de operações críticas, as quais fazem parte do caminho mais longo entre a primeira e a última operação, chamado de caminho crítico, como denotado, por exemplo, em (Bierwirth & Kuhpfahl, 2017; Burgy & Bulbul, 2018; Wang *et al*., 2018). A forma como as operações são movimentadas e a estrutura de vizinhança utilizada para definir conjuntos de operações passíveis de movimentação representam os dois componentes fundamentais dos métodos de busca local (Grabowski & Wodecki, 2005).

A literatura disponibiliza diversas formas de movimentação de operações em uma máquina, baseadas em movimentos de permutação ou inserção de operações, bem como diversas estruturas de vizinhança. Em geral, essas estruturas de vizinhanças foram definidas propondo a movimentação cada vez mais restritas com a finalidade de reduzir o tamanho da vizinhança gerada (número de vizinhos) empenhando-se em manter a capacidade de escapar de regiões com mínimos locais e alcançar melhores soluções, idealmente ótimos globais (Jain et al., 2000). Estes autores apresentaram um exemplo para uma instância clássica do JSSP com $n = m = 6$, mostrando que o número de movimentos definidos por diferentes estruturas de

vizinhança pode ser reduzido de 20 para 2, mas mesmo a estratégia mais eficiente, proposta por Nowicki e Smutnicki (1996), é muito influenciada pela escolha do procedimento de inicialização da solução, e que cerca de 99,7% dos movimentos escolhidos nessa estratégia não produzem melhoria na solução, independentemente do procedimento de inicialização ou do procedimento de geração de caminho crítico empregado.

Kuhpfahl e Bierwirth (2016), por exemplo, identificaram seis estruturas de vizinhança mais utilizadas na literatura, as quais foram classificadas em N1, N2, ..., N6, conforme notação definida em (Błazewicz et al., 1996). A primeira estrutura de vizinhança (N1) para o problema de JSSP foi introduzida por Van Laarhoven *et al*. (1992), a qual permite que soluções vizinhas sejam geradas a partir de permutações entre qualquer uma das operações críticas adjacentes em uma mesma máquina. Essa estrutura foi definida com base no resultado de Matsuo *et al*. (1988), que estabelece que a troca entre operações não críticas não promove a redução do *makespan*. Posto que tenha gerado bons resultados, a estrutura N1 gera inúmeras vizinhanças, visto que todas as operações do caminho crítico são permutadas, resultando em movimentos excessivos sem melhoria na solução.

Com base nas vantagens e desvantagens da estrutura N1, outras estruturas de vizinhanças foram propostas na literatura. Como uma extensão do trabalho de Matsuo *et al*. (1998), Van Laarhoven, *et al*. (1992) propuseram a estrutura N2, seguido por Dell'Amico e Trubian (1993) com as vizinhanças N3 e N4, a vizinhança N5 de Nowick e Smutnicki (1996), aliada ao método de busca tabu, e Balas e Vazacopulos (1998) com a estrutura N6. Todas essas vizinhanças utilizaram como base o conceito de bloco crítico, com movimentações de operações críticas adjacentes ou não adjacentes em uma mesma máquina (Jain et al., 2000).

Entre as diversas estruturas de vizinhança disponíveis, a abordagem proposta por Nowick e Smutnicki (1996) ganhou notoriedade por ser a mais restritiva vizinhança da literatura e, ao mesmo tempo, um procedimento eficaz para obter soluções de qualidade em um tempo relativamente curto (Kuhpfahl & Bierwirth, 2016; Jain et al., 2000; Grabowski & wodecki, 2005). Assim, a vizinhança N5 tem sido utilizada na busca local em diversos trabalhos como, por exemplo, Wang, *et al*., (2020), Tang *et al*. (2019), Li *et al*. (2017), Amirghasemi e Zamani (2015), Zhang *et al*. (2013), Ponsich e Coello (2013), Yang *et al*. (2008), Gonçalves, Mendes e Resende (2005), entre outros.

Entretanto, apesar da eficiência das estruturas de vizinhança, o cálculo do caminho crítico ainda se faz necessário para cada solução gerada nessa vizinhança. Isso eleva o custo computacional do processo de resolução, pois mesmo que apenas algumas operações críticas sejam permutadas nas estruturas mais restritivas, todo o caminho crítico é calculado para definir as operações a serem movimentadas.

Abordagens alternativas que realizam atualizações do caminho crítico a cada permutação, evitando a necessidade de recalcular o caminho crítico por completo, reduzem o tempo computacional da busca local, apesar disso possuem efeitos limitados pelo elevado número de soluções geradas naturalmente nesse processo (Akram et al., 2016; Cruz-Chávez, 2014). Matiet al. (2011) propuseram uma abordagem para reduzir os cálculos necessários para encontrar os valores do caminho crítico em uma solução, mostrando que a complexidade pode ser reduzida de $O(p^2)$ para O ($log\ p + n + m$), sendo p, n e m o número de operações, *jobs* e máquinas, respectivamente

2. Referencial Teórico

Uma forma eficiente de tratar o JSSP consiste em agendar todas as operações o mais cedo possível, sem violar as restrições impostas. Isso faz com que os problemas sejam modelados como um problema de caminho mais longo – chamados de caminhos críticos - e isso vem sendo explorado com sucesso relativo (Bierwirth & Kuhpfahl, 2017). O comprimento do caminho crítico é igual ao valor do makespan, sendo assim o problema de minimizar o makespan pode ser definido em como encontrar um agendamento das operações em cada máquina que minimize o comprimento do caminho mais longo, ou seja, do

caminho crítico (Pinedo, 2016).

Yamada e Nakano (1997) definem o caminho crítico como um elemento fundamental de uma solução, sendo que cada operação presente em um caminho crítico é chamada de operação crítica, e uma máquina que possui uma sequência de operações críticas consecutivas, é considerada uma máquina crítica. Como exemplo, na Figura 1 ilustra-se a configuração de um caminho crítico. Pode-se observar que o comprimento do caminho mais longo, destacado pelas setas em negrito, é de 19 unidades de tempo e composto pelas operações {O_{11}, O_{21}, O_{23}, O_{13}, O*33*} representadas pelos nós {1, 4, 5, 3, 9}. Na figura, a operação do job *i* na máquina *j* é denotada por O*ij*, e possui um tempo de duração associado. A operações *0* e *F* são fictícias e possuem duração igual a zero.

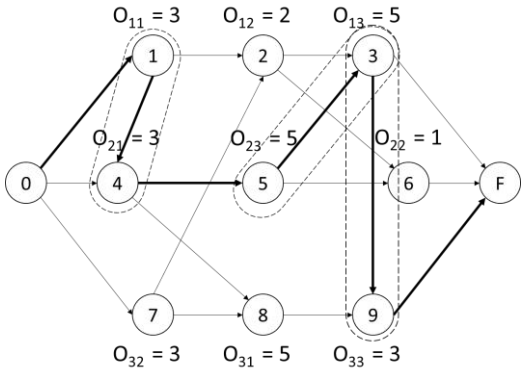
Adicionalmente, pode-se definir uma divisão do caminho crítico de uma solução em blocos críticos, os quais são sequências de operações adjacentes processadas em uma mesma máquina. Na Figura 1, por exemplo, observa-se dois blocos críticos, o primeiro composto pelas operações {O_{11}, O_{21}} processadas na máquina 1 e o segundo bloco {O_{23}, O_{13}, O_{33}} contendo as operações executadas na máquina 3.

As estratégias de movimentação em métodos de busca local atuam, em geral, permutando operações em blocos críticos segundo uma regra de vizinhança estabelecida, conforme descrito a seguir.

2.1 Estratégias de movimentação das operações críticas

A definição de quais e como as operações críticas serão permutadas, depende da estratégia da estrutura da vizinhança adotada. Entre as diversas estratégias que podem ser aplicadas, (Kuhpfahl & Bierwirth, 2016; Błazewicz et al., 1996) definem seis principais sendo elas conhecidas como N1, N2, ..., N6 sendo a N5 a mais eficiente (Kuhpfahl & Bierwirth, 2016; Jain et al., 2000; Grabowski & Wodecki, 2005).

Figura 1. Representação do caminho crítico de uma solução e seus respectivos blocos críticos. Os arcos horizontais, chamados conjuntivos, representam as operações de um mesmo conjunto de tarefas (*job*), enquanto os arcos diagonais (disjuntivos) conectam operações em uma mesma máquina na ordem determinada pela solução.



Fonte: Autores.

- N1: Proposta por Van Laarhoven *et al*. (1992) foi a primeira estrutura de vizinhança bem-sucedida para o problema de *job shop*, ela é gerada trocando um par de operações críticas em uma mesma máquina e com base nos seguintes princípios:
  - a permutação de operações não críticas não pode melhorar uma solução e pode gerar uma solução não factível;
  - dada uma solução factível, a permutação de duas operações críticas não pode gerar uma solução não factível;
- N2: A vizinhança proposta por Dell'Amico e Trubian (1993) pode reverter mais de um arco no caminho crítico. Supondo que *i* e *j* são duas operações consecutivas do mesmo bloco e uma delas está em um ponto extremo do

bloco, o predecessor de *i* e o sucessor de *j* também podem estar sujeitos a reversão com seus predecessores e sucessores, respectivamente;

- N3: Uma sequência de três operações no caminho crítico pode ser revertida, desde que tal reversão não leve a nenhum *loop*. Como extensão da vizinhança N1, a estrutura N3 não se limita à reversão apenas de três operações simultâneas, inclui também a permutação de um par de operações (Dell'Amico & Trubian, 1993);
- N4: Na vizinhança N4, cada operação de um bloco pode se mover para qualquer posição do bloco crítico, desde que nenhum ciclo seja criado. Ao contrário das outras três estruturas de vizinhança, que são baseadas em permutações de operações adjacentes. Uma operação salta sobre várias outras operações à esquerda ou à direita no bloco crítico. Portanto, para que a vizinhança N4 pudesse ser considerada uma expansão de todas as estruturas anteriores (Dell'Amico & Trubian, 1993);
- N5: A vizinhança definida por Nowicki e Smutnicki (1996) e adotada nesta dissertação, segue o mesmo princípio de permutação de operações adjacentes em uma mesma máquina da estrutura N1, porém, o objetivo da estrutura N5 é realizar menos movimentos que a vizinhança proposta por Van Laarhoven *et al*. (1992, que em alguns casos poderia gerar inúmeras permutações sem alcançar alguma melhoria. A permutação realizada na estrutura N5, ocorre apenas entre as duas primeiras e as duas últimas operações de cada bloco de operações no caminho crítico, sendo que no caso do primeiro bloco, apenas as duas últimas operações são permutadas, já no último bloco, a permutação ocorre apenas nas duas primeiras operações. Se algum bloco crítico for composto por apenas uma operação, nenhuma ação é realizada e se um bloco contiver somente duas operações, apenas uma permutação será realizada;
- N6: Por fim, a estrutura N6 de Balas e Vazacopoulos (1998), permite que cada operação de um bloco possa se mover para o início (antes da primeira operação) ou fim do bloco (depois da última operação), podendo gerar um ciclo.

Independente da estrutura de vizinhança escolhida, sua aplicação pressupõe a geração de uma solução por algum método de busca e posterior cálculo do caminho crítico. Nesse contexto, as metaheurísticas têm sido a escolha na maioria das abordagens propostas nos anos recentes com destaque, neste trabalho, para o Algoritmo Genético.

2.2 Algoritmo Genético

Os processos de sobrevivência e reprodução natural das espécies foram a base para o desenvolvimento do Algoritmo Genético (AG) apresentado por Holland (1975). Segundo este, a sobrevivência é mantida pelos organismos que melhor se adaptam ao ambiente e que possuem maior capacidade de reprodução e de transmissão da sua hereditariedade para as próximas gerações.

Sendo os operadores genéticos - seleção, cruzamento e mutação - conceitos inerentes aos algoritmos evolutivos, o AG sobressai-se em função da possibilidade de substituir as variáveis originais do problema por parâmetros codificáveis; associar uma função de aptidão para cada indivíduo; evoluir de um conjunto de potenciais soluções e não somente de uma solução inicial; buscar novas soluções através de um processo estocástico e encontrar a solução ótima dirigida por uma função objetivo. De acordo com Goldberg (1989), em função da sua simplicidade, flexibilidade e robustez, o AG tem sido usado para resolver problemas em que outras técnicas de otimização enfrentam obstáculos.

Semelhante à biologia, o AG possui um conjunto de indivíduos denominado como população e que representa as possíveis soluções do problema. A primeira população pode ser selecionada estocasticamente. Cada indivíduo é chamado de cromossomo e cada gene é uma variável do problema. A codificação do cromossomo pode variar de acordo com o problema

estudado: geralmente é uma *string* de tamanho fixo, que pode ter uma representação de números binários, inteiros ou reais. (Michalewicz, 1996).

Exemplos de codificação de cromossomo:

(a) [1 2 5 9 7 8 6 4 3] lista de prioridade representada por números inteiros

(b) [0011 0101 1100 1010 0110] representação por números binários

(c) [2,4 3,1 5,7 9,8 7,6] tempo de produção em números reais

Normalmente, a representação da solução, baseada em listas de números inteiros, é utilizada para solucionar problemas de otimização combinatória, como o sequenciamento de ordens de produção em *job shop*.

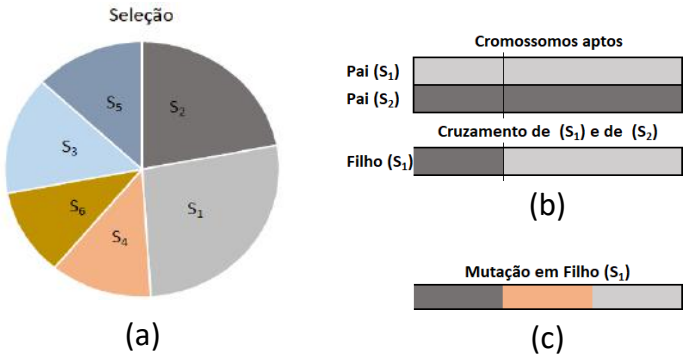
A cada iteração, chamada de geração, os indivíduos da população são submetidos a uma avaliação realizada pela função de *fitness*, cujo resultado - valor de aptidão – representa quão boa é essa solução, em relação à população atual, e a classifica para possível operação de reprodução. Por exemplo, se a otimização do AG está minimizando uma medida de desempenho, então, quanto menor o valor de aptidão, maior a probabilidade dessa solução ser selecionada. A função de *fitness* pode ser uma função matemática, um experimento, um modelo de simulação ou um metamodelo.

O aspecto fundamental para a evolução dos indivíduos ocorre na sua reprodução. O cruzamento, ou crossover, recombina os pais mais aptos, selecionados anteriormente, para a produção de novos cromossomos que, eventualmente, passam por um processo chamado mutação. O objetivo da mutação é garantir a diversidade da população.

Assim, a operação de seleção visa permitir que os cromossomos mais aptos dentro da população tenham mais oportunidades de produzir novos descendentes, enquanto os operadores genéticos de crossover e mutação garantem, respectivamente, a interação entre os cromossomos da população e a permanência de diversidade dentro do conjunto de soluções potenciais para o problema. Dessa forma, é possível que os descendentes tenham atributos distintos de seus antepassados e que o indivíduo gerado tenha maior capacidade de adaptação ao ambiente em que vive (Mitchell, 1997).

A Figura 2 traz um exemplo de seleção, cruzamento e mutação:

Figura 2. Representação das operações de seleção, cruzamento e mutação.

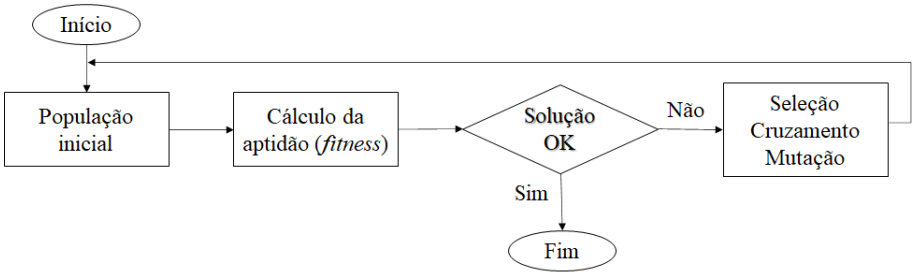


Fonte: Autores.

Observe que em (2a) ilustra-se a seleção dos cromossomos S_1 até S_6 com probabilidade proporcional à aptidão do indivíduo. Em (2b) é feito o cruzamento dos dois cromossomos S_1 e S_2 que possuem maior aptidão para a reprodução, eles são cruzados e depois recebem uma mutação em (2c).

Uma representação simplificada do algoritmo genético pode ser vista no diagrama da Figura 3.

Figura 3. Representação simplificada do funcionamento do Algoritmo Genético.



Fonte: Autores.

A população inicial é avaliada, se a solução foi encontrada ou a condição de parada foi atingida, as iterações cessam, caso contrário são aplicados os operadores genéticos e cada novo indivíduo é então, avaliado.

3. Materiais, Métodos e Procedimentos

Este trabalho realizou um estudo do ambiente de produção *job shop* a fim de verificar se há correlação entre as características estatísticas presente no problema como as operações críticas dos agendamentos realizados (Mirshekarian E sormaz, 2016). Para os procedimentos técnicos desse artigo foram adotados métodos compatíveis com pesquisa bibliográfica e experimental.

Foi realizado um levantamento bibliográfico sobre o problema e variáveis relacionadas, por meio de sistemas de busca em periódicos disponíveis no portal CAPES e IEEE. A estratégia de busca consistiu no uso das versões em inglês dos termos: agendamento, técnicas metaheurísticas, algoritmo genético, busca local, caminho crítico e operações críticas, além de *job shop*.

Na etapa de experimentação o algoritmo genético utilizado foi o presente na biblioteca numérica GAlib escrita por Wall (2007), do *Massachusetts Institute of Technology-MIT*, disponível em <http://lancet.mit.edu/ga/ GAlib.html>. Os parâmetros de otimização inseridos no AG para as instâncias do problema segue a seguinte configuração, obtida experimentalmente: população de 500 indivíduos, taxa de mutação em 5%, taxa de cruzamento de 90% e 50 iterações para cada execução.

Como estratégia de intensificação um algoritmo de busca local baseado no método de descida de encosta foi implementando e utilizou como operador de movimento a estratégia proposta por Nowicki e Smutnicki (1996) denominada como N5.

Para realizar a coleta de dados, foram utilizados os grupos de instâncias de LA01 ao LA10, LA16 ao LA20 e por fim, as instâncias LA36 ao LA40, propostas por Lawrence (1984), disponíveis na OR-Library (Beasley, 1990) especialmente desenvolvidas para o ambiente JSSP. Trata-se de problemas clássicos com valor do *makespan* ótimo conhecido conforme encontra-se na literatura (Asadzadeh, 2015; Pongchairerks, 2019). Ressalta-se que cada instância foi executada dez vezes e todas as operações críticas que, quando permutadas, promoveram melhoria na solução foram armazenadas para análise posterior. Essas operações foram agrupadas por máquinas, com o objetivo de verificar se há máquina(s) específica(s) que concentra(m) um maior número de operações críticas. Foram realizados testes de hipóteses para avaliar se as diferenças observadas no número de operações críticas nas máquinas poderiam ser consideradas estatisticamente significativas. Nesse caso, uma análise de variância (ANOVA) foi executada seguida da avaliação de todos os seus pressupostos.

O número de operações críticas em cada máquina foi correlacionado utilizando estatísticas das características do problema. Em caso de correlação significativa tais operações, ou as máquinas nas quais essas operações estão concentradas, foram estimadas a partir das características do problema. Esse processo pode ser visto com mais detalhes na etapa de resultados.

Para realizar os experimentos computacionais foi utilizado um computador com processador Intel® Core™ i7-4500U CPU 1.80GHz e memória RAM – 8,00 GB. Os testes estatísticos foram realizados com o auxílio da ferramenta de análise de dados do Excel. O gráfico e tabelas apresentadas foram criados com o *software* Minitab.

4. Resultados

Os dados coletados para as operações críticas indicam as máquinas e os *jobs* para os quais uma permutação resultou em uma redução no *makespan.*

Inicialmente, as operações críticas foram agrupadas por máquina com vistas a identificar a possibilidade de classificar as máquinas quanto essa criticidade. A identificação de máquinas que concentrem um maior volume de operações críticas pode ser usada para direcionar a geração de soluções pelo método de busca. Adicionalmente, o estudo da correlação dessa caraterística com dados oriundos do problema pode indicar máquinas críticas sem a necessidade do cálculo repetitivo do caminho crítico a cada solução. Os resultados correspondentes são apresentados na Tabela 1

Os valores da Tabela 1 se referem ao número total de operações críticas observadas em cada máquina nas 10 replicações para cada problema. Observa-se que para algumas instâncias há uma ou mais máquinas nas quais nota-se uma concentração de operações críticas como, por exemplo, para a LA06 na qual a máquina M1 apresenta cerca de três vezes mais operações críticas que o número médio de operações nas demais 4 máquinas, igual a 16,75. Resultado semelhante pode ser observada para a instância LA07.

Para a instância LA03 nota-se uma maior concentração de operações críticas nas máquinas M1 e M2 com cerca de 2 vezes o número de operações nas demais máquinas. Trata-se de um resultado especialmente interessante visto que essa é, entre todas as instâncias testadas, a mais difícil de ser resolvida. Assim, um método de solução poderia ser programado para concentrar as suas buscas em soluções obtidas por permutações das operações nessas máquinas. Por outro lado, outras instâncias apresentam um equilíbrio nessa distribuição como é o caso da LA04. Uma representação gráfica dessa concentração de operações críticas é ilustrada na Figura 4.

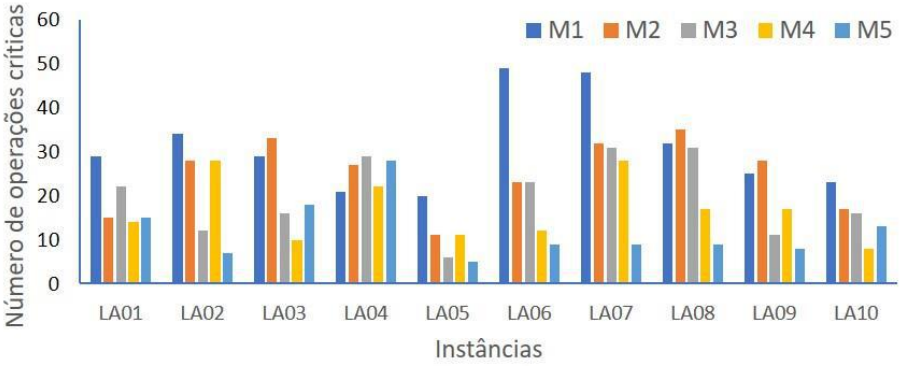
Além das dez primeiras instâncias investigadas, mais dois grupos de instâncias foram utilizados para os próximos testes: LA16 ao LA20 e LA36 ao LA40. As instâncias para os problemas LA16 ao LA20 possuem dimensão 10 *jobs* por 10 máquinas, enquanto o grupo com os problemas LA36 ao LA40 tem dimensão de 15 *jobs* por 15 máquinas (15x15). Ambos, portanto, são considerados problemas quadrados de difícil solução, em geral.

Tabela 1. Concentração de operações críticas nas máquinas.

Instância	***M1***	***M2***	***M3***	***M4***	***M5***
LA01	29	15	22	14	15
LA02	34	28	12	28	7
LA03	**29**	**33**	16	10	18
LA04	21	27	29	22	28
LA05	20	11	6	11	5
LA06	**49**	23	23	12	9
LA07	**48**	32	31	28	9
LA08	32	35	31	17	9
LA09	25	28	11	17	8
LA10	23	17	16	8	13

Fonte: Autores.

Figura 4. Número de operações críticas nas máquinas para as diferentes instâncias.



Fonte: Autores.

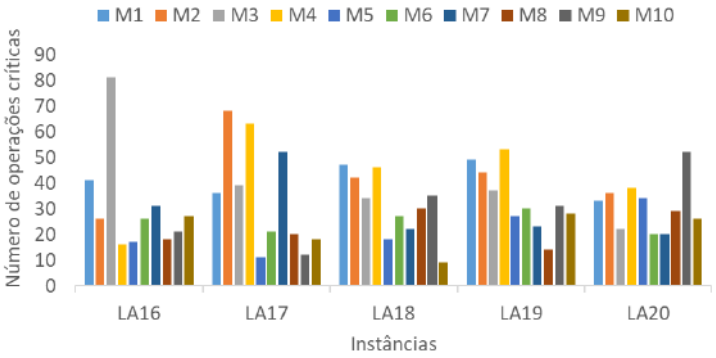
As concentrações de operações críticas para os dois novos grupos são exibidas respectivamente nas tabelas 2 e 3, e ilustradas graficamente nas Figuras 5 e 6.

Tabela 2. Concentração de operações críticas LA16-LA20.

Instância	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
LA16	41	26	81	16	17	26	31	18	21	27
LA17	36	68	39	63	11	21	52	20	12	18
LA18	47	42	34	46	18	27	22	30	35	9
LA19	49	44	37	53	27	30	23	14	31	28
LA20	33	36	22	38	34	20	20	29	52	26

Fonte: Autores.

Figura 5. Número de operações críticas nas máquinas para as instâncias LA16-LA20.



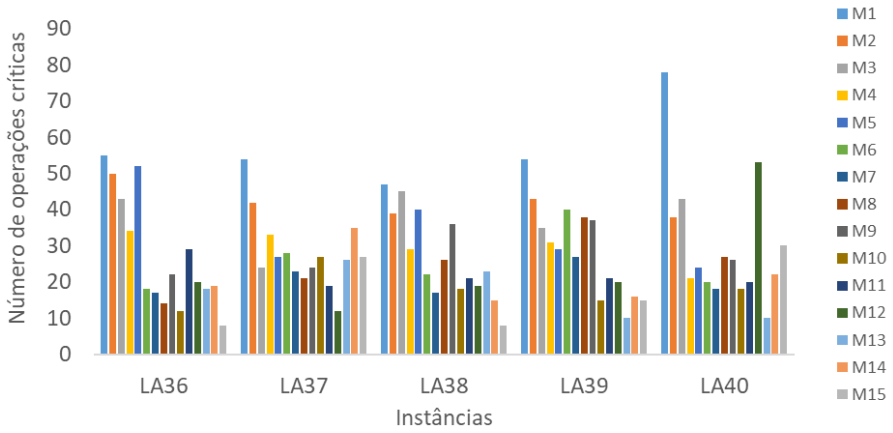
Fonte: Autores.

Tabela 3. Concentração de operações críticas LA36-LA40.

Instância	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
LA36	55	50	43	34	52	18	17	14	22	12	29	20	18	19	8
LA37	54	42	24	33	27	28	23	21	24	27	19	12	26	35	27
LA38	47	39	45	29	40	22	17	26	36	18	21	19	23	15	8
LA39	54	43	35	31	29	40	27	38	37	15	21	20	10	16	15
LA40	78	38	43	21	24	20	18	27	26	18	20	53	10	22	30

Fonte: Autores.

Figura 6. Número de operações críticas nas máquinas para as instâncias LA36-LA40.



Fonte: Autores.

É possível notar que o cenário visto na Tabela 1 e Figura 4 se repete para as Tabelas 2 e 3 e Figuras 5 e 6, nas quais ao menos uma máquina possui uma concentração elevada de operações críticas em relação as demais, em geral as primeiras máquinas concentraram mais operações críticas, em especial as máquinas M1 e M2.

O número de operações críticas nas máquinas obtido para cada uma das replicações também foi avaliado para verificar se as diferenças observadas são estaticamente significativas. Nesse caso foi realizada uma Análise de Variância fator único seguido de um teste de análise por pares nos casos para os quais a igualdade foi descartada. As hipóteses testadas nesse caso, para cada instância, são definidas com base na média do número de operações críticas para cada máquina Mi, i =1, ..., 5, em cada replicação, e descritas conforme equação (1).

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \quad (1)$$
$$H_1 : \textit{ao menos uma média é diferente}$$

O resultado da Análise de Variância obtido para cada uma das 10 instâncias é demonstrado na Tabela 4, na qual um *p-valor* maior que 0,05 indica a aceitação de H_0, hipótese nula em que não há diferenças na concentração de operações críticas nas máquinas. Nesse caso 30 replicações (#) foram realizadas para cada instância.

Tabela 4. Resultados da ANOVA para as 20 instâncias testadas.

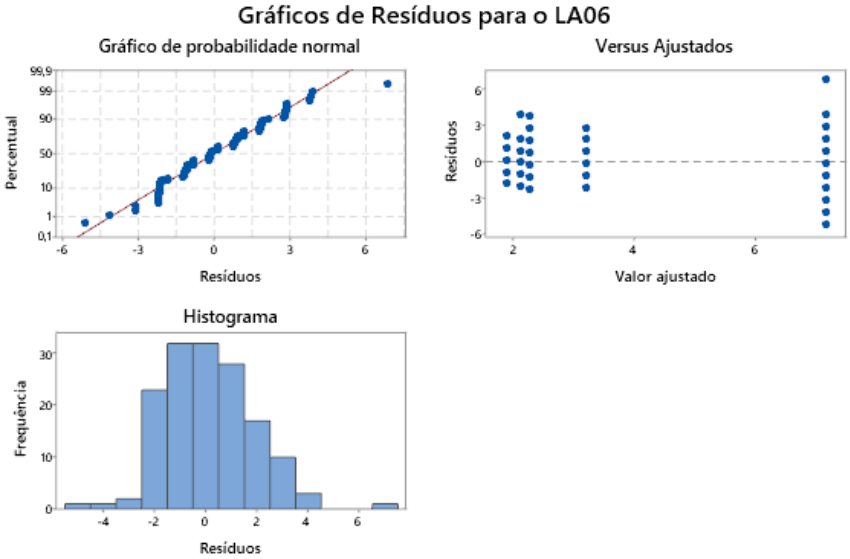
Instância	*p-valor*	Aceita H_0?	Instância	*p-valor*	Aceita H_0?
LA01	0,001708	Não	LA16	1,75207E-38	Não
LA02	8,57E-17	Não	LA17	1,33478E-22	Não
LA03	7,02E-22	Não	LA18	2,19688E-18	Não
LA04	3,99E-05	Não	LA19	5,54E-09	Não
LA05	8,44E-17	Não	LA20	1,91178E-06	Não
LA06	6,34E-25	Não	LA36	2,2771E-41	Não
LA07	1,30E-31	Não	LA37	8,88073E-20	Não
LA08	3,70E-11	Não	LA38	1,78492E-22	Não
LA09	2,80E-07	Não	LA39	2,28336E-34	Não
LA10	6,80E-12	Não	LA40	4,21534E-41	Não

Fonte: Autores.

Para validar os resultados obtidos pela ANOVA, foram gerados três gráficos de resíduos para as vinte instâncias analisadas. A Figura 7 ilustra os dados obtidos para a instância LA06. Vale ressaltar que apesar do gráfico dos Resíduos versus Ajustes indicar que há uma diferença na variância, esse resultado não gerou impacto nos demais.

Nos casos para os quais a hipótese nula é rejeitada, ou seja, em que há evidências nos dados de que o número médio de operações críticas nas máquinas é estatisticamente diferente, o que ocorreu para todas as instâncias, foram realizados testes de Tukey, em que cada par de máquina é verificado com o objetivo de identificar em quais máquinas de fato existem diferenças. Essa verificação é importante para classificar as máquinas em função de sua criticidade (maior ocorrência de operações críticas) e usar essa informação para desenvolver métodos de otimização que concentrem sua busca no espaço das soluções obtidas por permutações de operações nessas máquinas, por exemplo.

Figura 7. Avaliação dos pressupostos para a ANOVA.



Fonte: Autores.

A avaliação por pares realizada por meio do teste de Tukey, é feita pelo cálculo do intervalo de confiança (95%) para as diferenças entre as médias de cada par de máquinas e verifica se o valor zero está contido no intervalo, o que indica a possibilidade de igualdade. Os resultados dessa análise estão apresentados na Tabela 5, na qual os valores em negrito indicam os pares de máquinas para as quais não há diferença significativa entre as médias.

É importante notar na Tabela 4 que há diferença significativa em todos os casos. Isso indica que em todos os vinte problemas existe ao menos uma máquina que possui uma concentração de operações críticas diferente das demais. As máquinas nas quais essa diferença de fato ocorre são apresentadas na Tabela 5. Nesse caso, vale destacar que, em geral, as duas primeiras máquinas apresentam uma maior concentração das operações críticas, em especial para as instâncias de LA06 à LA10. Nessas instâncias a diferença entre os pares de médias foi bastante significativa sempre entre as máquinas M1 e M2 e as demais, especialmente a primeira (resultado destacado em cinza na Tabela 5).

Tabela 5. Resultado da análise por pares de média para as instâncias testadas. Destaques em negrito para os pares estatisticamente iguais (intervalos que incluem o zero).

Pares	LA01		LA02		LA03		LA04		LA05	
M1-M2	**-0.12**	**2.45**	**-0.11**	**2.64**	3.12	5.94	**-0.78**	**1.12**	1.33	3.00
M1-M3	**-0.59**	**1.59**	3.66	5.74	4.79	7.14	**-1.03**	**1.03**	1.02	2.78
M1-M4	0.89	3.51	0.07	2.66	3.75	6.45	1.39	3.21	2.01	3.39
M1-M5	**-0.67**	**1.81**	3.86	5.87	4.25	6.55	**-1.68**	**1.15**	2.77	4.17
M2-M3	**-1.63**	**0.29**	2.24	4.63	0.59	2.28	**-1.12**	**0.78**	**-1.06**	**0.53**
M2-M4	0.02	2.05	**-1.16**	**1.36**	**-0.37**	**1.50**	1.14	3.12	**-0.05**	**1.12**
M2-M5	**-1.79**	**0.59**	2.75	4.45	**-0.35**	**2.08**	**-1.65**	**0.78**	0.67	1.93
M3-M4	0.63	2.77	-4.72	-1.95	**-1.74**	**0.00**	1.28	3.32	0.08	1.52
M3-M5	**-1.10**	**1.23**	**-0.55**	**0.88**	**-1.54**	**0.40**	**-1.36**	**0.83**	1.03	2.10
M4-M5	-2.62	-0.65	2.30	4.70	**-0.74**	**1.34**	**-3.70**	**-1.43**	0.25	1.28

Fonte: Autores.

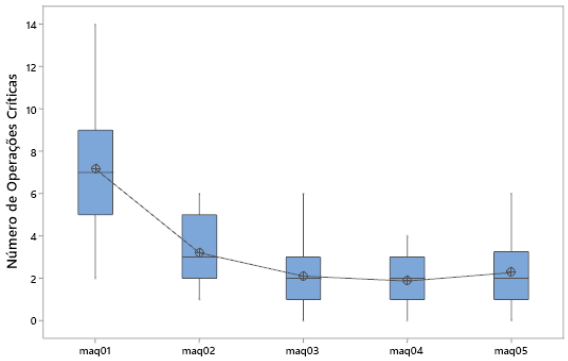
Tabela . Resultado da análise por pares de média para as instâncias testadas. Destaques em negrito para os pares estatisticamente iguais (intervalos que incluem o zero).

Pares	LA01		LA02		LA03		LA04		LA05	
M1-M2	2.79	5.14	6.58	9.89	**-1.44**	**1.24**	**-0.03**	**2.56**	-2.36	-0.30
M1-M3	4.09	6.05	7.41	10.66	0.63	3.64	1.36	3.71	0.18	1.89
M1-M4	4.22	6.38	7.81	11.19	1.59	4.35	1.68	3.98	0.96	2.64
M1-M5	3.78	6.02	9.19	12.81	2.64	5.36	1.75	3.99	0.52	2.21
M2-M3	0.46	1.74	**-0.79**	**2.39**	1.14	3.32	0.28	2.25	1.46	3.27
M2-M4	0.65	2.01	**-0.12**	**2.65**	2.14	4.00	0.37	2.76	2.38	3.89
M2-M5	0.11	1.75	1.62	3.91	2.86	5.34	0.36	2.84	1.85	3.55
M3-M4	**-0.34**	**0.80**	**-0.90**	**1.83**	**-0.37**	**2.03**	**-0.53**	**1.13**	0.15	1.38
M3-M5	**-0.81**	**0.48**	0.71	3.22	0.72	3.01	**-0.54**	**1.21**	**-0.28**	**0.94**
M4-M5	**-1.09**	**0.29**	0.46	2.54	**-0.17**	**2.23**	**-0.93**	**1.00**	**-0.93**	**0.06**

Fonte: Autores.

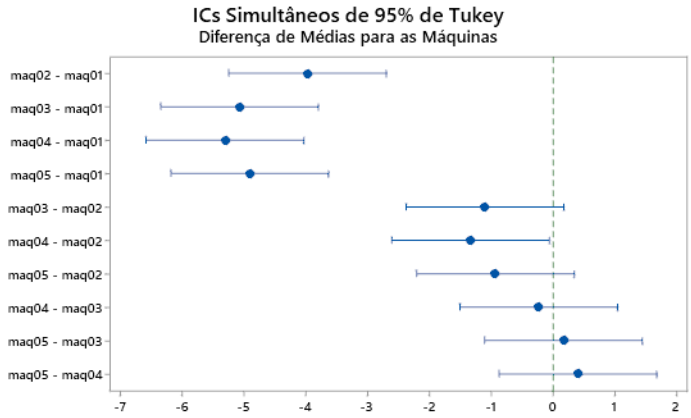
Na Figura 8 ilustra-se a distribuição das operações crítica nas máquinas na forma de um gráfico boxplot. Apenas os resultados para a instância LA06 são exibidos, sendo semelhantes para os demais casos. O resultado da figura indica uma diferença significativa entre a máquina M1 e as demais, o que é comprovado numericamente pelo teste de Tukey apresentado na Figura 9.

Figura 8. Distribuição do número de operações críticas para a instância LA06.



Fonte: Autores.

Figura 9. Teste de Tukey para a instância LA06. Intervalos que contém o *zero* indicam valores do par estatisticamente igual.



Fonte: Autores.

Os valores do número médio de operações críticas em cada máquina para o primeiro grupo de problemas podem ser observados na Tabela 6, indicando de fato uma maior concentração nas duas primeiras máquinas.

Tabela 6. Número médio de operações críticas por máquina, em 30 replicações.

Instância	Número médio de operações					Instância	Número médio de operações				
	M1	***M2***	***M3***	***M4***	***M5***		***M1***	***M2***	***M3***	***M4***	***M5***
LA01	4.67	3.50	4.17	2.47	4.10	LA06	7.17	3.20	2.10	1.87	2.27
LA02	7.67	6.40	2.97	6.30	2.80	LA07	13.97	5.73	4.93	4.47	2.97
LA03	8.97	4.43	3.00	3.87	3.57	LA08	6.97	7.07	4.83	4.00	2.97
LA04	4.93	4.77	4.93	2.63	5.20	LA09	5.70	4.43	3.17	2.87	2.83
LA05	4.43	2.27	2.53	1.73	0.97	LA10	3.33	4.67	2.30	1.53	1.97

Fonte: Autores.

Adicionalmente, foi realizado o estudo de correlação entre as estatísticas definidas na Tabela 7 e os resultados das concentrações de operações críticas da Tabela 1. Os valores das estatísticas para as dez instâncias do primeiro grupo são apresentados na Tabela 8.

Tabela 7. Notação e descrição das estatísticas consideradas nos testes.

Estatística	Notação	Definição
Média dos tempos	*Mean*	Média dos tempos de processamento das operações em cada máquina
Desvio-padrão dos tempos	Desv-Pad	Desvio-padrão dos tempos de processamento das operações em cada máquina
Soma dos intervalos entre chegadas	SumIC	Intervalo entre chegadas dos *Jobs* nas máquinas

Fonte: Autores.

Tabela 8. Estatísticas calculadas para as instâncias de teste.

Instância	*Mean*	Desv-Pad	SumIC	Instância	*Mean*	Desv-Pad	SumIC
LA01	60,90	29,97	4	LA06	61,73	28,31	4
	53,60	23,57	4		49,33	27,20	4
	53,00	24,65	4		52,33	24,58	4
	50,80	21,70	4		48,40	21,29	4
	66,60	29,35	4		54,33	31,76	4
LA02	59,70	27,23	4	LA07	57,93	23,41	3
	46,30	22,55	4		46,27	20,92	4
	45,30	29,40	4		49,00	22,62	4
	63,50	27,63	3		53,33	19,37	4
	49,50	29,23	4		43,13	24,26	4
LA03	51,50	31,02	2	LA08	50,27	27,18	4
	58,80	23,11	4		43,93	30,27	4
	44,40	25,99	4		51,73	31,26	4
	38,00	22,44	4		51,53	32,88	4
	45,60	23,78	3		57,53	33,88	4
LA04	53,60	31,26	4	LA09	51,27	24,05	4
	45,80	31,29	4		63,40	29,83	4
	52,50	28,89	4		48,73	24,93	4
	45,10	35,74	4		61,47	25,90	4
	53,70	39,25	3		59,33	27,04	4
LA05	59,30	23,72	4	LA10	43,80	30,49	4
	46,30	18,61	4		63,87	28,16	4
	53,20	25,69	4		58,73	23,65	4
	39,10	23,61	4		51,20	27,56	4
	30,40	28,41	4		50,40	30,59	3

Fonte: Autores.

Os valores de correlação estão apresentados na Tabela 9 para as instâncias LA01 à LA10, já a Tabela 10, conta com os valores de correlação para os grupos de problemas quadrados (LA16-LA20 e LA36-LA40). Pode-se notar que para as instâncias LA02, LA03 e LA05 há uma correlação positiva de moderada a forte entre os tempos médios das operações (*Mean*) e a concentração de operações críticas nas máquinas, em especial para o LA03 que apresenta um coeficiente de correlação ρ igual a 0,98. Esse resultado indica que quanto maior os tempos médio de processamento das operações nas máquinas maior a concentração de operações críticas nessas máquinas. Ressalta-se que o LA03 é, das instâncias testadas, uma das que representa a maior dificuldade para solução em função do número de ótimos globais nessa instância (Pérez et al., 2012).

Tabela 9. Coeficientes de correlação ρ entre as estatísticas e as operações críticas. O símbolo † denota a impossibilidade de cálculo do coeficiente.

Instância	*Mean*	Desv-Pad	*Sum*IC	Instância	*Mean*	Desv-Pad	*Sum*IC
LA01	0,196	0,570	†	LA06	0,773	0,123	†
LA02	0,607	-0,600	-0,298	LA07	0,838	-0,166	-0,740
LA03	0,979	0,393	-0,364	LA08	-0,827	-0,801	†
LA04	0,202	-0,048	-0,398	LA09	0,240	0,287	†
LA05	0,652	-0,439	†	LA10	-0,172	0,219	0,244

Fonte: Autores.

Tabela 10. Coeficientes de correlação ρ entre as estatísticas e as operações críticas para as instâncias LA16-LA20 e LA36-LA40.

Instância	*Mean*	Desv-Pad	Instância	*Mean*	Desv-Pad
LA16	0,479	0,082	LA36	0,718	-0,155
LA17	0,719	-0,034	LA37	-0,129	0,285
LA18	0,586	0,075	LA38	0,250	0,047
LA19	-0,233	0,338	LA39	0,407	-0,012
LA20	0,384	-0,085	LA40	0,133	-0,261

Fonte: Autores.

Também foram observadas correlações significativas entre os tempos médios e a ocorrência de operações críticas nas máquinas para as instâncias LA06, LA07 e LA08 (Tabela 9), LA16, LA17, LA18 e LA36 (Tabela 10). É importante notar que para a instância LA08 o coeficiente de correlação é negativo indicando que a concentração de operações críticas é inversamente proporcional à estatística *Mean*. Esse resultado, aparentemente contraditório, pode estar associado aos valores de desvio-padrão do tempo de processamento visto que, nesse caso (LA08) também foi observado uma correlação negativa significativa com a estatística Desv-Pad (ρ = -0,801).

Por fim, apresenta-se na Tabela 11 os valores de *makespan* alcançados após a realização dos experimentos, em que a coluna “BKS” se refere ao melhor valor de *makespan* conhecido na literatura (*Best Known Solution*). Vale notar que, apenas os valores de *makespan* dos dez primeiros problemas foram alcançados, com exceção da instância LA03, que apesar de sua dimensão 10x5, é de difícil resolução.

Tabela11. Comparação entre a melhor solução conhecida com o *makespan* encontrado.

Instância	Tamanho	*BKS*	*Makespan*	Instância	Tamanho	*BKS*	*Makespan*
LA01	10x05	**666**	**666**	LA16	10x10	945	973
LA02	10x05	**655**	**655**	LA17	10x10	784	789
LA03	10x05	597	603	LA18	10x10	848	860
LA04	10x05	**590**	**590**	LA19	10x10	842	866
LA05	10x05	**593**	**593**	LA20	10x10	902	907
LA06	15x05	**926**	**926**	LA36	15x15	1268	1432
LA07	15x05	**890**	**890**	LA37	15x15	1397	1588
LA08	15x05	**863**	**863**	LA38	15x15	1196	1425
LA09	15x05	**951**	**951**	LA39	15x15	1233	1437
LA10	15x05	**958**	**958**	LA40	15x15	1222	1374

Fonte: Autores.

5. Conclusão

O presente trabalho teve como objetivo avaliar a distribuição das operações críticas nas máquinas para verificar se existe um padrão de concentração dessas operações, possibilitando a criação futura de métodos de busca local mais eficientes em termos computacionais.

Foram definidas três estatísticas em relação aos tempos de processamento e ao intervalo de chegada de *jobs* nas máquinas. Como resultado, observou-se que para algumas instâncias há uma ou mais máquinas nas quais nota-se uma concentração de operações críticas, com uma maior concentração de operações críticas nas máquinas M1 e M2 para as instâncias LA01 à LA10, com cerca de 2 vezes o número de operações nas demais máquinas em alguns casos. Para os problemas quadrados, instâncias LA16 à LA20 e LA36 à LA40, a concentração de operações críticas foi observada em sua grande maioria para as quatro primeiras máquinas.

Quando se considera a concentração de operações nas máquinas em múltiplas replicações, há diferença significativa em todos os casos. Isso indica que em todos os problemas testados existe ao menos uma máquina que possui uma concentração de operações críticas diferente das demais.

Considerando os testes de correlação, foi observada uma correlação positiva de moderada a forte entre os tempos médios das operações (*Mean*) e a concentração de operações críticas nas máquinas, em especial para o LA03 que apresenta um coeficiente de correlação ρ igual a 0,98. Resultado semelhante foi obtido para os problemas maiores. Esse resultado indica que quanto maior os tempos médios de processamento das operações nas máquinas, maior a concentração de operações críticas nessas máquinas. A correlação com respeito as outras duas medidas estatísticas não foram, em geral, significativas. Sendo assim, os experimentos indicam que é possível criar métodos de solução programados para concentrar as suas buscas em soluções obtidas por permutações das operações nas máquinas com maior concentração de operações críticas. Logo, os métodos de busca podem concentrar as permutações em operações nessas máquinas.

Como sugestão para pesquisas futuras, pode-se realizar novas análises considerando o agrupamento das operações críticas por *jobs* e por *jobs* e máquinas de maneira conjunta, bem como um aprofundamento da análise dos resultados considerando problemas nos quais o número de *jobs* é igual ao número de máquinas.

Agradecimentos

Os autores agradecem a Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e ao Programa de Pós graduação em Informática e Gestão do Conhecimento da Universidade Nove de Julho – PPGI/UNINOVE pelo apoio financeiro.

Referências

Aarts, E., & Lenstra, J. K. (Eds.). (1993). *Local Search in Combinatorial Optimization.* Princeton University Press. https://apps.crossref.org/coaccess/coaccess.html?doi=10.2307%2Fj.ctv346t9c

Abdullahi, M., Ngadi, M., & Abdulhamid, S. (2016). Symbiotic organism search optimization based task scheduling in cloud computing environment. *Future Generation Computer Systems, 56*, pp. 640-650. https://doi.org/10.1016/j.future.2015.08.006

Akram, K., Kamal, K., & Z. A. (2016). Fast simulated annealing hybridized with quenching for solving job shop scheduling problem. *Applied Soft Computing, 49*, pp. 510-523. https://doi:10.1016/j.asoc.2016.08.037

Amirghasemi, M., & Zamani, R. R. (2015). An effective asexual genetic algorithm for solving the job shop scheduling problem. *Computers & Industrial Engineering, 83*, pp. 123-138. https://doi:10.1016/j.cie.2015.02.011

Asadzadeh, L. (2015). A local search genetic algorithm for the job shop scheduling problem with intelligent agents. *Computers & Industrial Engineering, 85*, pp. 123-138. https://doi:10.1016/j.cie.2015.04.006

Balas E, V. A. (1998). Guided local search with shifting bottleneck for job shop scheduling. *Management Science, 44*, pp. 262-275. Fonte: https://www.jstor.org/stable/2634500

Beasley, J. E. (1990). OR-Library. http://people.brunel.ac.uk/~mastjjb/jeb/info.html

Bierwirth, C., & J., K. (2017). Extended GRASP for the Job Shop Scheduling Problem with Total Weighted Tardiness Objective. *European Journal of Operational Research, 261(3)*, pp. 835-848. https://doi:10.1016/j.ejor.2017.03.030

Blazewicz, J., Domschke, W., & Pesch, E. (1996). The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research, 1, N.93*, pp. 1-33. https://doi.org/10.1016/0377-2217(95)00362-2

Bueno, A., Godinho Filho, M., & Frank, A. G. (2020). Bueno, A.; Godinho Filho, M.; Frank, A. G. (2020). Smart production planning and control in the Industry 4.0 context: A systematic literature review. *Computers & Industrial, 149*. https://doi:10.1016/j.cie.2020.106774.

Burgy, R., & Bulbul, K. (2018). The job shop scheduling problem with convex costs. *European Journal of Operational Research., 268 (1)*, pp. 82-100. https://doi:10.1016/j.ejor.2018.01.027

Çalis, B., & Bulkan, S. (2015). A research survey: review of AI solution strategies of job shop scheduling problem. *Journal of Intelligent Manufacturing, 26(5)*, pp. 961-973. https://doi:10.1007/s10845-013-0837-8

Chan, F. T., Wong, T. C., & Chan, L. Y. (2008). A Genetic algorithm-based approach to job shop scheduling problem with assembly stage. *In 2008 IEEE International Conference on Industrial Engineering and Engineering Management* (pp. 331-335). Singapore: IEEE. https://doi:10.1109/IEEM.2008.4737885

Cruz-Chávez, M. A. (2014). Neighbourhood Generation mechanism applied in simulated annealing to job shop scheduling problems. *International Journal of Systems Science*, pp. 2673-2685. https://doi:10.1080/00207721.2013.876679

Dell'Amico, M., & Trubian, M. (s.d.). Applying tabu-search to job-shop scheduling problem. *41*, pp. 231-252. https://doi.org/10.1007/BF02023076

Freitag, M., & Hildebrandt, T. (2016). Automatic design of scheduling rules for complex manufacturing systems by multi-objective simulation-based optimization. *CIRP Annals-Manufacturing Technology, 65 (1)*, pp. 433-436. https://doi.org/10.1016/j.cirp.2016.04.066.

Fuchigami, H. Y., Moura, M. A., & Branco, F. J. (2017). Modelos Matemáticos para Programação Job Shop com Tempos de Configuração Independentes de Sequência. *Revista Produção Online, 17 (1)*, pp. 245-267. https://doi:10.14488/1676-1901.v17i1.2504.

Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley.

Gonçalves, J. F., Mendes, J. J., & Resende, M. G. (2005). A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research,, 167*, pp. 77-95. https://doi:10.1016/j.ejor.2004.03.012

Grabowski, J., & Wodecki, M. (2005). A very fast tabu search algorithm for job shop problem. In book: Metaheuristic optimization via memory and evolution; tabu search and scatter search. Em A. B. Rego C, & A. Rego C. (Ed.), *Metaheuristic optimization via memory and evolution; tabu search and scatter*. Kluwer Academic Publishers. https://doi:10.13140/2.1.2986.6880

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

Jain, A. S., Rangaswamy, B., & Meeran, S. (2000). New and "stronger" job-shop neighbourhoods: A focus on the method of Nowicki and Smutnicki (1996). *Journal of Heuristcs, 6*, pp. 457-480. https://doi:10.1023/A:1009617209268

Kong, W., Lei, Y., & Ma, J. (2016). Virtual machine resource scheduling algorithm for cloud computing based on auction mechanism. *Optik-International Journal for Light and Electron Optics, 127 (12)*, pp. 5099-5104. https://doi:10.1016/j.ijleo.2016.02.061

Kuhpfahl, J., & Bierwirth, C. (2016). A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective. *omputers & Operations Research, 66*, pp. 44-57. https://doi:10.1016/j.cor.2015.07.011

Lawrence, S. (1984). Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement). *Tese (Doutorado em Administração Industrial) - Carnegie-Mellon University, Pittsburgh*.

Lenstra, J. K., Rinnooy Kan, A. H., & Brucker, P. (1977). Complexity of machine. Em E. J. P.L. Hammer (Ed.), *Annals of Discrete Mathematics. 1*, pp. 343-362. Elsevier.

Li, K., Zhang, X., Leung, J.-T., & Yang, S. (2016). Parallel machine scheduling problems in green manufacturing industry. *Journal of Manufacturing Systems, 38*, pp. 98-106. https://doi:10.1016/j.jmsy.2015.11.006

Li, X., Peng, Z., Du, B., Jun, G., Wenxiang, X., & Zhuang, K. (2017). Hybrid artificial bee colony algorithm with a rescheduling strategy for solving flexible job shop scheduling problems. *Computers & Industrial Engineering, 113*, pp. 10-26. https://doi:10.1016/j.cie.2017.09.005

Liu, W., Liang, Z., Ye, Z., & Liu, L. (2016). The optimal decision of customer order decoupling point for order insertion scheduling in logistics service supply chain. *International Journal of Production Economic, 175*, pp. 50-60. https://doi:10.1016/j.ijpe.2016.01.021

Lu, H., Shi, J., Fei, Z., Zhou, Q., & Mao, K. (2018). Analysis of the similarities and differences of job-based scheduling problems. *European Journal of Operational Research, 270 (3)*, pp. 809-825. https://doi:10.1016/j.ejor.2018.01.051

Mahmud, S., Abbasi, A., Chakrabortty, R. K., & Ryan, M. J. (2021). Multi-operator communication based differential evolution with sequential Tabu Search approach for job shop scheduling problems. *Applied Soft Computing, 108*, pp. 1-15. https://doi:10.1016/j.asoc.2021.107470

Mati, Y., Dauzre-Prs, S., & Lahlou, C. (2011). A general approach for optimizing regular criteria in the job-shop scheduling problem. *European Journal of Operational Research*, 212, pp. 33-42. https://doi:10.1080/00207721.2013.876679

Matsuo, H. D., Suh, C., & Sullivan, R. (1988). Controlled search simulated annealing method for general job shop scheduling problem. *Working Paper, Department of Management, The University of Texas at Austin, Austin, TX*.

Mattfeld, D. C., & Bierwirth, C. (2004). n efficient genetic algorithm for job shop scheduling with tardiness objectives. *European Journal of Operational Research, 155 (3)*, pp. 616-630. https://doi:10.1016/S0377-2217(03)00016-X

Michalewicz, Z. (1996) Genetic Algorithms + Data Structures = Evolution Programs. 3rd revised and extended. Berlim, Springer-Verlag.

Mirshekarian, S., & Sormaz, D. N. (2016). Correlation of Job-Shop Scheduling Problem Features with Scheduling EfÞciency. *Expert Systems With Applications, 62*, pp. 131-147. https://doi:10.1016/j.eswa.2016.06.014

Mitchell, T. M. (1997). *Machine Learning* (16 ed.). New York: McGraw-Hill.

Muminu, O. A., & A., A. (2016). inimizing the weighted number of tardy jobs on multiple machines: A review. *Journal of Industrial & Management Optimization, 12 (4)*, pp. 1465-1493. https://doi:10.3934/jimo.2016.12.1465

Nowicki, E., & Smutnicki, C. (1996). A Fast Taboo Search Algorithm for the Job Shop Problem. *Management Science, 42 (6)*, pp. 797-813. https://www.jstor.org/stable/2634595

Parente, M., Figueira, G., Amorim, P., & Marques, A. (2020). roduction scheduling in the context of Industry 4.0: review and trends. *International Journal of Production Research, 58 (17)*, pp. 5401-5431. https://doi:10.1080/00207543.2020.1718794

Pérez, E., Posada, M., & Herrera, F. (2012). Analysis of new niching genetic algorithms for finding multiple solutions in the job shop scheduling. *Journal of Intelligent Manufacturing, 23*, pp. 341-356. https://doi:10.1007/s10845-010-0385-4

Pinedo, M. L. (2016). *Scheduling: Theory, Algoritms, and Systems* (5a ed.). Springer. https;//doi:10.1007/978-3-319-26580-3

Pongchairerks, P. A. (2019). A Two-Level Metaheuristic Algorithm for the Job-Shop Scheduling Problem. *Complexity, 2019*, pp. 1-11. https://doi:10.1155/2019/8683472

Ponsich, A., & Coello, C. A. (2013). A Hybrid Differential Evolution Tabu Search Algorithm for the Solution of Job-Shop Scheduling Problems. *Applied Soft Computing, 13 (1)*, pp. 462-474. https://doi:10.1016/j.asoc.2012.07.034

Tamssaouet, K., Dauzère-Pérès, S., & Yugma, C. (2018). Metaheuristics for the Job-Shop Scheduling Problem with Machine Availability Constraints. *Computers and Industrial Engineering, 125*, pp. 1-8. https://doi:10.1016/j.cie.2018.08.008

Tang, H., Chen, R., Li, Y., Peng, Z., Guo, S., & Du, Y. (2019). Flexible job-shop scheduling with tolerated time interval and limited starting time interval based on hybrid discrete PSO-SA: An application from a casting workshop. *Applied Soft Computing, 78*, pp. 176-194. https://doi:10.1016/j.asoc.2019.02.011

Van Laarhoven, P. E. (1992). Job shop scheduling by simulate annealing. *Institute for Operations Research and the Management Sciences, 40, n.1*, pp. 113-125. https://www.jstor.org/stable/171189

Wall, M. (1996). GAlib: A C++ library of genetic algorithm components. *Mechanical Engineering Departament, Massachussetts Institute of Technology*. Fonte: http://lancet.mit.edu/ga/dist/

Wang, B., Wang, X., Lan, F., & Pan, Q. (2018). A hybrid local-search algorithm for robust job-shop scheduling under scenarios. *Applied Soft Computing, 62*, pp. 259-271. https://doi:10.1016/j.asoc.2017.10.020

Wang, X., Wang, B., Zhang, X., Xia, X., & Pan, Q. (2021). wo-objective Robust Job-Shop Scheduling with Two Problem-Specific Neighborhood Structures. *Swarm and Evolutionary Computation, 61*, p. 100805. https://doi:10.1016/j.swevo.2020.100805

Wu, Z., Sun, S., & Yu, S. (2020). Optimizing makespan and stability risks in job shop scheduling. *Computers and Operations Research, 122*, p. 104963. https://doi:10.1016/j.cor.2020.104963

Yamada, T. (2003). Studies on metaheuristics for jobshop and flowshop scheduling problems. *PhD dissertação Kyoto University*. http://www.kecl.ntt.co.jp/as/members/yamada/YamadaThesis.pdf

Yamada, T., & Nakano, R. (1997). Job shop scheduling. *IEE Control Engineering Series, 55*, pp. 134-160. https://www.kecl.ntt.co.jp/as/members/yamada/galbk.pdf

Yang, J., Sun, L., Lee, H. P., Qian, Y., & Liang, Y. C. (2008). Clonal Selection Based Memetic Algorithm for Job Shop Scheduling Problems. *Journal of Bionic Engineering, 5 (2)*, pp. 111–119. https://doi:10.1016/S1672-6529(08)60014-1

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under Industry 4.0. *Journal of Intelligent Manufacturing, 30*, pp. 1809–1830. https://doi:10.1007/s10845-017-1350-2

Zhang, R., & Wu, C. (2011). An artificial bee colony algorithm for the job shop scheduling problem with random processing times. *Entropy, 13 (9)*, pp. 1708–1729. https://doi:10.3390/e13091708

Zhang, R., Song, S., & Wu, C. (2013). A hybrid differential evolution algorithm for job shop scheduling problems with expected total tardiness criterion. *Applied Soft Computing*, pp. 1448–1458. https://doi:10.1016/j.asoc.2012.02.024

Zhang, R., Song, S., & Wu, C. (2013(b)). A hybrid artificial bee colony algorithm for the job shop scheduling problem. *International Journal of Production Economics, 141*((1)), pp. 167-178. https://doi:10.1016/j.ijpe.2012.03.035