DeepL

Assine o DeepL Pro para traduzir arquivos maiores.
Mais informações em www.DeepL.com/pro.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Recebido em 24 de julho de 2020, aceito em 8 de agosto de 2020, data de publicação em 11 de agosto de 2020, data da versão atual em 20 de agosto de 2020.

Identificador de objeto digital 10.1109/ACCESS.2020.3015796

Meta-heurística para resolver o problema de agendamento multiobjetivo de job shop em uma célula robótica

XIAOHUI LI [1], (Membro, IEEE), XI YANG [1], YI ZHAO [1], (Membro, IEEE), YING TENG [2] e YUAN DONG [1]

[1]Escola de Engenharia Eletrônica e de Controle, Universidade de Chang'an, Xi'an 710054, China

[2]Escola de Administração e Economia, Universidade de Ciência e Tecnologia Eletrônica da China, Chengdu 610051, China

Autor correspondente: Ying Teng (tengy@uestc.edu.cn)

Este trabalho foi apoiado em parte pelos Fundos de Pesquisa Fundamental para o Projeto Chave de Inovação e Integração Integrada da Internet das Coisas Nacional sob o Subsídio 2018-470, em parte pelo Projeto Chave do Plano de Pesquisa e Desenvolvimento de Shaanxi sob o Subsídio 2019ZDLGY03-01 e em parte pelo Projeto de Ciência e Tecnologia de Xi'an sob o Subsídio 201805045YD23CG29.

RESUMO Este artigo trata do problema de programação multiobjetivo de job shop em uma célula robótica (MOJRCSP). Todos os trabalhos são processados de acordo com sua ordem de operação nas estações de trabalho. Diferentemente do problema clássico de programação de job shop, o problema estudado considera que o transporte dos trabalhos é feito por um robô. Além disso, espera-se que os trabalhos sejam concluídos em uma janela de tempo, em vez de uma data de vencimento constante. Um modelo de programação inteira mista (MIP) é proposto para formular esse problema. Devido às características especiais do problema estudado e à sua complexidade computacional NP-hard, foi proposta uma metaheurística baseada no algoritmo TLBO (Teaching Learning Based Optimization). O algoritmo proposto determina simultaneamente as atribuições das operações nas estações de trabalho, as atribuições dos robôs para as operações de transporte e a sequência de movimentação dos robôs. O objetivo é minimizar o makespan e o total de adiantamentos e atrasos. Os resultados computacionais validaram ainda mais a eficácia e a robustez de nosso algoritmo proposto.

INDEX TERMS Célula robótica, job shop, otimização multiobjetivo, busca local, otimização baseada em ensino-aprendizagem.

I. INTRODUÇÃO

Os equipamentos de produção inteligente têm sido aplicados gradualmente no recente setor de manufatura. Como unidade central da oficina inteligente, a célula robótica (RC) é composta por um sistema unificado de controle de informações, robôs de transporte de materiais e um conjunto de equipamentos de processamento. Os trabalhos podem ser transferidos entre a estação de trabalho e a estação de carga/descarga pelo robô. Além disso, a célula robótica pode ser a vários equipamentos de produção e armazenamento, e toda a unidade é controlada pelo sistema de controle de informações. Devido à alta eficiência e flexibilidade da célula robótica, ela pode ser amplamente utilizada em vários campos de produção industrial avançada e moderna [1]-[4]. Entretanto, a existência de robôs industriais aumenta a complexidade do ambiente de produção. Portanto, no processo de programação de job shop, o tomador de decisões não deve considerar apenas a organização do processamento dos trabalhos, mas também a alocação de tarefas e a movimento do robô na célula. Portanto, a programação tradicional de job shop não pode ser aplicada diretamente a esse tipo de problema de produção de manufatura inteligente avançada, de modo que um novo modo de produção e um novo esquema de otimização de programação são necessários para resolver o problema de programação da célula robótica. Assim, torna-se um importante tópico de pesquisa desenvolver um método eficaz de programação colaborativa ideal (para encontrar o plano ideal de processamento de trabalhos e o plano de manuseio do robô) para resolver o problema de programação da célula robótica. Yan *et al.* [5] tratam de um problema de programação dinâmica de job-shop em tempo real em uma célula robótica. O problema estudado pode ser modelado como uma loja de trabalho em que os trabalhos precisam ser transportados entre as máquinas por robôs. Elmi e Topaloglu [6] resolveram o problema de programação de célula robótica de job cíclico com vários robôs. Todos trabalhos são processados na ordem de suas operações em várias máquinas com tempos de processamento padrão e robôs de garra única. Os robôs realizam as operações de transporte dos trabalhos entre as estações.

Existem diferentes métodos de pesquisa para resolver o problema de programação de células robóticas e, principalmente, soluções comuns

O editor associado que coordenou a revisão deste manuscrito e o aprovou para publicação foi Muhammad Zakarya.

Este trabalho está licenciado sob uma Licença Creative Commons Attribution 4.0. Para obter mais informações, consulte https://creativecommons.org/licenses/by/4.0/

incluem algoritmos exatos e algoritmos aproximados. O algoritmo exato geralmente usa o método de modelagem matemática para resolver o problema. Caumond *et al.* [7] forneceram a formulação matemática e as soluções ótimas para o problema de programação de sistemas de manufatura flexíveis (FMSSP) com um veículo. Essa formulação linear é diferente dos trabalhos anteriores, pois leva em conta o número máximo de trabalhos permitidos no sistema, as capacidades limitadas de buffer de entrada/saída, as viagens de veículos vazios e as viagens sem movimento simultâneo. Che *et al.* [8] abordaram o problema de programação de 2 ciclos com vários robôs em uma célula robótica sem espera em que duas peças entram e saem da célula durante cada ciclo. Em seu trabalho, vários robôs são responsáveis pelo transporte de peças entre as máquinas. Eles desenvolveram um algoritmo polinomial para encontrar o número mínimo de robôs para todos os tempos de ciclo viáveis. Nattafa *et al.* [9] estudaram os problemas de programação com trabalhos de diferentes famílias em máquinas paralelas, em que nem todas as máquinas são qualificadas (elegíveis) para processar todos os trabalhos. Para resolver esse problema, são propostos um modelo de programação linear inteira e um modelo de programação com restrições.

Para o problema de programação, os algoritmos aproximados são mais utilizados. Zhang e Xing [10] abordaram pela primeira vez o problema de programação de flowshop de buffer limitado distribuído com critério de minimização de makespan. Eles propuseram sucessivamente duas heurísticas construtivas para gerar uma programação rápida e fornecer uma boa inicialização de meta-heurísticas. Lei [11] apresentou uma otimização de enxame de partículas para o de programação de job shop com vários objetivos. Seu objetivo é minimizar simultaneamente o makespan e o atraso total dos trabalhos. Wang *et al.* [12] investigaram o problema de agendamento de flow shop de permutação distribuída (DPFSP) com eficiência energética e com os objetivos de makespan e consumo de energia. Um algoritmo de enxame de baleias multiobjetivo (MOWSA) é proposto para resolver esse DPFSP com eficiência energética. Além disso, para aumentar a eficiência energética sem afetar a eficiência da produção, uma nova busca local dependente do problema é desenvolvida para melhorar a capacidade de exploração do MOWSA. Kurdi [13] propõe um algoritmo memético de modelo de ilha aprimorado com uma nova fase de cooperação naturalmente inspirada (IIMMA) para o problema de programação de job shop multiobjetivo. A nova fase de cooperação é usada principalmente para melhorar os recursos de exploração do algoritmo. Udomsakdigool e Khachitvichyanukul [14] apresentam um algoritmo de colônia de formigas para solucionar o problema de agendamento de trabalho com vários objetivos. Os objetivos considerados nesse estudo incluem a minimização do makespan, do tempo médio de fluxo e do atraso médio.

Para acelerar a convergência do algoritmo, muitos trabalhos combinaram o algoritmo de aproximação tradicional com a tecnologia de pesquisa de vizinhança, que pode ser mais eficiente na solução do problema de agendamento de FMS. Para prevenir a convergência prematura em um problema de otimização multiobjetivo, o algoritmo SA (Simulated annealing) é utilizado como pesquisa local nos trabalhos de Mokhtari e Hasani [15]. Ele é capaz de obter mais soluções ao escapar dos mínimos locais em um espaço de solução mais amplo. Arnold e Sorensen [16] combinaram três poderosos métodos de busca local técnicas e as implementou de eficiente. Em seu trabalho, foram feitos experimentos para determinar como a busca local pode ser combinada de forma eficaz com a perturbação e a poda e como orientar a busca para soluções melhores. Moghaddam *et al.* [17] propuseram um novo algoritmo de otimização de enxame de partículas (PSO) de arquivo de Pareto multiobjetivo combinado com operadores genéticos como pesquisa de vizinhança variável (VNS). Seus métodos forneceram uma solução melhor para instâncias de problemas de grande porte em um tempo de computação razoável. Moslehi e Mahnam [18] apresentam uma nova abordagem baseada em uma hibridização do algoritmo de enxame de partículas e de busca local para resolver o problema de agendamento flexível de job-shop com múltiplos objetivos. Zhang *et al.* [19] formularam o problema de programação do processo de tingimento têxtil como um modelo de otimização com dois objetivos, no qual um objetivo está relacionado ao custo do atraso, enquanto o outro objetivo reflete o nível de emissão de poluentes. E um algoritmo de otimização de enxame de partículas multiobjetivo aprimorado por técnicas de busca local específicas do problema (MO-PSO-L) para buscar soluções não dominadas de alta qualidade. Luo *et al.* propuseram um Distributed Flexible Job Shop Schedul- ing Problem with Transfers (DFJSPT), no qual as operações de um trabalho podem ser processadas em diferentes fábricas. A fim de expandir o espaço de busca e acelerar a velocidade de convergência da solução, um algoritmo memético eficiente (EMA) é proposto para resolver o DFJSPT com os objetivos de minimizar o makespan, a carga de trabalho máxima e o consumo total de energia das fábricas.

De fato, os trabalhos anteriores se concentram principalmente nos casos de problemas de programação de objetivo único com robôs ou de problemas de programação multiobjetivo sem robôs transferidos. Essas soluções propostas não conseguem lidar bem com o nosso problema estudado (MOJRCSP). Além disso, a maioria dos métodos de pesquisa se baseia em algoritmos exatos, e há poucos algoritmos aproximados disponíveis, especialmente para oficinas de manufatura multiobjetivo com células robóticas. Portanto, neste trabalho, nosso objetivo é o MOJRCSP com o algoritmo de otimização baseado no aprendizado do professor multiobjetivo aprimorado (IMOTLBO). Como o algoritmo TLBO precisa de menos parâmetros, ele é simples e tem grande capacidade de convergência e velocidade de convergência rápida, por isso tem sido aplicado em vários campos [21]-[24], [26]-[28]. Para o problema do algoritmo de otimização baseado no aprendizado do professor com múltiplos objetivos (MOTLBO), até o momento, a aplicação do MOTLBO tem sido muito limitada. Em [29], uma versão modificada do algoritmo TLBO é introduzida e aplicada para a otimização multiobjetivo de trocadores de calor por Rao *et al.* O trocador de calor de aleta de placa e o trocador de calor de casco e tubo são considerados para a otimização. A maximização da eficácia do trocador de calor e a minimização do custo total do trocador são consideradas as funções objetivas. Tog˘an [22] apresenta um procedimento que emprega uma técnica de otimização baseada em ensino-aprendizagem (TLBO) para otimização discreta de estruturas de aço planas. O algoritmo de projeto visa a obter estruturas de peso mínimo sujeitas aos requisitos de resistência e deslocamento impostos. Nesse

No nosso trabalho, nossa meta de otimização é a minimização do makespan e do total ponderado de antecipação e atraso. Para acelerar a convergência do algoritmo, combinamos o algoritmo TLBO com uma poderosa técnica de pesquisa local: descida de vizinhança variável (VND). O algoritmo determina a sequência ideal de atribuição de operações de trabalhos e o movimento do robô na programação. Finalmente, com diferentes instâncias, o IMOTLBO proposto é comparado com algoritmos multiobjetivo, *ou seja*, o Non-dominated Sorting Genetic Algorithm II (NSGA-II) [23], o Pareto-based Grouping Discrete Harmony search algorithm (PGDHS) [24] e a Otimização por Enxame de Partículas Multiobjetivo (MOPSO) [25]. Os resultados comprovaram a eficácia e a vantagem do algoritmo proposto.

A contribuição e a inovação deste estudo são destacadas a seguir:

- O problema de programação de células robóticas de job shop com múltiplos objetivos é formulado como uma programação inteira mista
  com transporte por robô de manuseio, são consideradas as janelas de tempo finalizadas dos trabalhos.
- Este estudo propõe uma nova metaheurística híbrida multiobjetivo que combina o algoritmo TLBO com uma poderosa técnica de busca local, o algoritmo de experiência.
  Os resultados mentais mostram que nosso algoritmo obtém soluções não dominadas melhores do que outros.

O restante deste documento está organizado da seguinte forma: A descrição do problema e o modelo matemático são apresentados na Seção 2. Na Seção 3, os detalhes da nossa proposta IMOTLBO são apresentados para resolver o MOJRCSP. Os resultados experimentais e a conclusão são apresentados na Seção 4.

II. DEFINIÇÃO DO PROBLEMA E FORMULAÇÃO MATEMÁTICA

A. DESCRIÇÃO DO PROBLEMA

Uma célula robótica típica consiste em M estações de trabalho, uma estação de carga/descarga e um robô de manuseio. A estrutura do sistema RC é ilustrada na Figura 1. Uma série de trabalhos $J = \{J_1, \ldots J_N\}$ chegam à estação de carga do sistema RC e cada um deles tem suas próprias sequências de processamento. Cada trabalho consiste em um número de $_{n(j)}$operações $O_{(j)\,(1)}, \ldots O_{(jn)\,(j)}$especificando a rota de visita do trabalho e diferentes tempos de processamento p_{ji}. O número de operações de cada trabalho $n_j$pode ser diferente e $n_j \leq M$.

Sem perda de generalidade, cada trabalho começa com a carga e termina na estação de descarga. Um robô de manuseio automatizado move os trabalhos de uma estação para outra. Uma tarefa de movimentação Tr_{ji}indica que o robô transporta o trabalho j da estação $\mu_{(j,i)\,(-)\,(1)}$para a estação μ_{ji}, em que μ_{ji}corresponde à estação processada estação de operação O_{ji}. Antes do movimento Tr_{ji}, se o robô não estiver na posição $\mu_{(j,i)\,(-)\,(1)}$, será necessário um movimento vazio $Tr^{'}_{ji}$para chegar à posição desejada. O transporte carregado do robô e O movimento vazio deve ser considerado no problema de programação. O tempo de carregamento do robô ε_ce o tempo de descarregamento $\varepsilon_d$são definidos como um valor constante.

Cada operação O_{ji}inicia o processo na estação μ_{ji}em seu horário de chegada s_{ji}e deixa a estação no horário t_{ji}, sua permanência

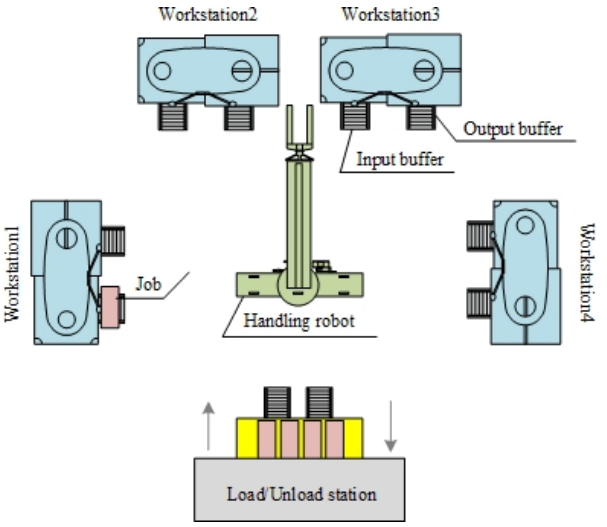


FIGURA 1. Uma célula robótica típica com um robô de manuseio de materiais.

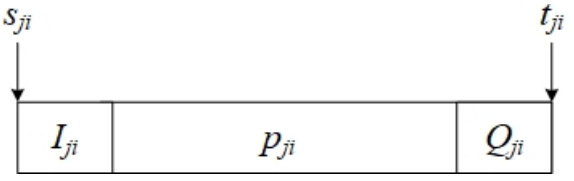


FIGURA 2. Variáveis indicadas na descrição de uma operação.

O tempo de espera pode ser dividido em três partes: p_{ji}representa o tempo de processamento de O_{ji}, I_{ji}e Q_{ji}correspondem ao tempo de espera de O_{ji}no buffer de entrada e no buffer de saída, respectivamente. Essas variáveis modelam o processo de operação na estação (mostrado na Figura 2).

O objetivo é minimizar o makespan e o atraso e a precocidade totais simultaneamente, já que o tempo de conclusão esperado de cada trabalho é confinado em janelas de tempo com os limites de tempo mínimo e máximo fornecidos.

As seguintes premissas são consideradas:

- Todos os trabalhos entram e saem do sistema RC por meio das estações de carga e descarga;
- a capacidade da estação de carga/descarga e do buffer de entrada/saída da estação de trabalho é ilimitada;
- A estação de trabalho e o robô podem processar no máximo um trabalho por vez;
- cada operação só pode ser processada em apenas uma estação de trabalho;
- as falhas de máquinas e robôs são ignoradas;
- no tempo zero, todos os trabalhos chegaram à estação de carregamento e todas as estações de trabalho e o robô estão disponíveis;
- nenhuma preempção é permitida.

B. NOTAÇÕES

1) NOTAÇÕES PARA PARÂMETROS

J número de trabalhos, que são indexados como trabalho j, j e $[1, J]$

M número de estações, $1, \ldots . M$ indica máquinas, 0 e $M+1$ apresentam as estações de carga e descarga

n_j número de operações do trabalho j, 0 e n_j+ 1 apresentam a operação na carga e descarga estações
O_{ji} a operação i^th do trabalho j
p_{ji} tempo de processamento de O_{ji}
a_j limite inferior da data de vencimento do trabalho j
b_j limite superior da data de vencimento do trabalho j
τ_{rl} duração da viagem carregada da estação r até a estação l
σ_{rl} duração da viagem em vazio da estação r até a estação l
ε_c tempo de carregamento para um trabalho
ε_d tempo de descarregamento para um trabalho
μ_{ji} estação em que O_{ji}é processado
Tr_{ji} Tarefa de transporte carregada entre $O_{(j,i)\,(-)\,(1)}$ e $O_{(j,i)\,(}$de $\mu_{(j,i)\,(-)\,(1)}$para μ_{ji})
Tr^{r}_{ji} tarefa de movimentação vazia antes de Tr_{ji}
H um número positivo grande

2) NOTAÇÕES PARA VARIÁVEIS

t_{ji} tempo de conclusão da operação O_{ji}
Q_{ji} tempo de espera da operação O_{ji}no buffer de saída da estação μ_{ji}
I_{ji} tempo de espera da operação O_{ji}no buffer de entrada da estação μ_{ji}
s_{ji} tempo de chegada de O_{ji}na estação μ_{ji}
C_j tempo de conclusão do trabalho j E_j precocidade do trabalho j
T_j atraso do trabalho j
U_{jik} = 1, se $O_{ji}$for processado antes de O_{jk}
= 0, caso contrário
$X_{(jij)\,ri^r}$ = 1, se $O_{ji}$for processado antes de $O_{(j)\,ri^r}$ na estação μji
= 0, caso contrário
$Z_{(jij)\,ri^r}$ = 1, se $Tr_{ji}$for processado antes de $Tr_{(j)\,ri^r}$
= 0, caso contrário

C. FUNÇÃO OBJETIVA E RESTRIÇÕES

A função objetiva é:

$$f_1 = min(C_{max}) \quad (1)$$

$$f_{(2} = min(\quad (E \qquad T)) \quad (2)$$

$$= \qquad j\,j+$$

$$j\in[1,J]$$

As seguintes restrições são consideradas:

$$t_{ji} = s_{ji} + I_{ji} + p_{ji} + Q_{ji} \quad (3)$$

onde $j \in [1, J]$, $i \in [1, {}_{n(j)}]$.

Essa restrição garante que o tempo de conclusão da operação seja o mesmo que o tempo de conclusão da operação.

O_{ji}é igual ao tempo de chegada mais o tempo de permanência na estação, em que o tempo de permanência consiste no tempo de espera no buffer de entrada I_{ji}, no tempo de processamento p_{ji}e no tempo de espera no buffer de saída Q_{ji}.

$$t_{ji} = t_{j,i(-)\,(1} + \tau\,(\mu_{j,i(-)(1)},\ \mu_{ji}) + \varepsilon_c + \varepsilon_d + I_{ji} + p_{ji} + Q_{ji} \quad (4)$$

onde $j \in [1, J]$, $i \in [1, {}_{n(j)}]$.

$$t_{j1} \geq \tau\,(0,\ \mu_{j1}) + \varepsilon_c + \varepsilon_d + I_{j1} + p_{j1} + Q_{j1} \quad (5)$$

$$t_{j,n_j+1} = t_{j,n} + \tau\,(\mu_{j,n},\ M + {}_{j}1) + \varepsilon_c + \varepsilon_d \quad (6)$$

$$C_j = t_{j,n_j+1} \quad (7)$$

onde $j \in [1, J]$.

Essas restrições garantem que as operações sejam realizadas de acordo com a sequência do trabalho, em que $O_{(j,i)\,(-)\,(1)}$é denotada como a operação imediatamente anterior de O_{ji}na sequência de trabalho.
sequência de tarefas. A restrição (4) calcula o tempo de conclusão t_{ji}de O_{ji}, que é igual à soma do tempo de conclusão da operação imediatamente anterior $O_{(j,i)\,(-)\,(1)}$, o tempo de transporte $\tau\,(\mu_{(j,i)\,(-)\,(1)},\ \mu_{ji})$, o tempo de carregamento ε_ce o tempo de descarregamento ε_c,
o tempo de permanência de O_{ji}.

As restrições (5) e (6) podem ser consideradas como um caso especial de (4). A restrição (5) calcula o tempo de conclusão do primeira operação de cada trabalho com $t_{(j)\,(0)} = 0$. As restrições (6) e (7) calculam o tempo de conclusão de cada trabalho, que é igual a o momento em que o trabalho é inserido na estação de descarregamento, onde $Ij_{,nj\,1+} = pj_{,nj\,1+} = Qj_{,nj\,1+} = 0$.

$$t_{ji} + \tau\,(\mu_{ji},\ \mu_{jk}) + \varepsilon_d + \varepsilon_c \leq s_{jk} + H\,(1 - U_{jik}) \quad (8)$$

onde $j \in [1, J]$, $i, k \in [1, {}_{n(j)}]$.

Essa restrição garante que duas operações do mesmo trabalho não podem ser realizadas ao mesmo tempo. Se $O_{ji}$for processada antes de $O_{(jk)\,(Ujik=1}$, a operação O_{jk}deve começar depois que o robô mover o trabalho da estação μ_{ji}para o buffer de entrada de μ_{jk}.

$$t_{(ji)\,-p(ji)\,-Q(ji)} \geq t_{(j)\,r,i^r} - Q_{(j)\,ri^r} - H\,(1 - X_{(jij)\,ri^r}) \quad (9)$$

$$X_{jij^ri^r} + X_{j^ri^rji} \leq 1 \quad (10)$$

em que $j, j^r \in [1, J]$, $i \in [1, {}_{n(j)}]$, $i^r \in [1, {}_{n(j)^r}]$, com $\mu_{ji} = {}_{\mu(j)\,ri^r}$

Essas restrições garantem que duas operações no mesmo estação não pode ser processada ao mesmo tempo. Se $O_{ji}$for processado antes de $O_{(j)\,ri^r}$ na estação $\mu_{(ji)\,(X(jij)\,ri^r} = 1)$, então $O_{(j)\,ri^r}$ iniciará seu processo depois que O_{ji}entrar no buffer de saída.

$$t_{ji} + \varepsilon_c + \tau\,(\mu_{ji},\ \mu_{j^r,i^r(-)\,(1)}) + \varepsilon_d + \sigma\,(\mu_{j^r,i^r(-)\,(1)},\ \mu_{j^ri^r})$$

$$\leq t_{(j)\,ri^r} + H\,(1 - Z_{(jij)\,ri^r}) \quad (11)$$

$$Z_{jij^ri^r} + Z_{j^ri^rji} \leq 1 \quad (12)$$

onde $j, j \in [1, J]$, $i \in [1, {}_{n(j)}]$, $i \in [1, {}_{n(j)^r}]$

Essas restrições garantem que dois movimentos carregados não possam ser executados ao mesmo tempo pelo robô. No caso de $Z_{(jij)\,ri^r} = 1$, o movimento carregado $Tr_{(j)\,ri^r}$ deve começar depois que o movimento carregado anterior $Tr_{ji}$for concluído. Quando o robô terminar o movimento
movimento anterior, e se sua posição atual $\mu_{ji}$for diferente da posição inicial $\mu_{(j)\,r(,i)\,r(-)\,(1)}$do movimento carregado Tr_{ji}, será necessário um movimento vazio.

$$t_{ji} \leq t_{j^ri^r} + H\,(1 - X_{jij^ri^r}) \quad (13)$$

$$t_{(ji)\,-Q(ji)} \leq t_{(j)\,ri^r} - Q_{(j)\,ri^r} + H\,(1 - X_{(jij)\,ri^r}) \quad (14)$$

$$t_{(ji)\,-Q(ji)\,-p(ji)} \leq t_{(j)\,ri^r} - Q_{(j)\,ri^r} - p_{(j)\,ri^r} + H\,(1 - X_{(jij)\,ri^r}) \quad (15)$$

$$t_{(ji)\,-Q(ji)\,-p(ji)\,-I(ji)} \leq t_{(j)\,ri^r} - Q_{(j)\,ri^r} - p_{(j)\,ri^r} - I_{(j)\,ri^r}$$

$$+ H\,(1 - X_{(jij)\,ri^r}) \quad (16)$$

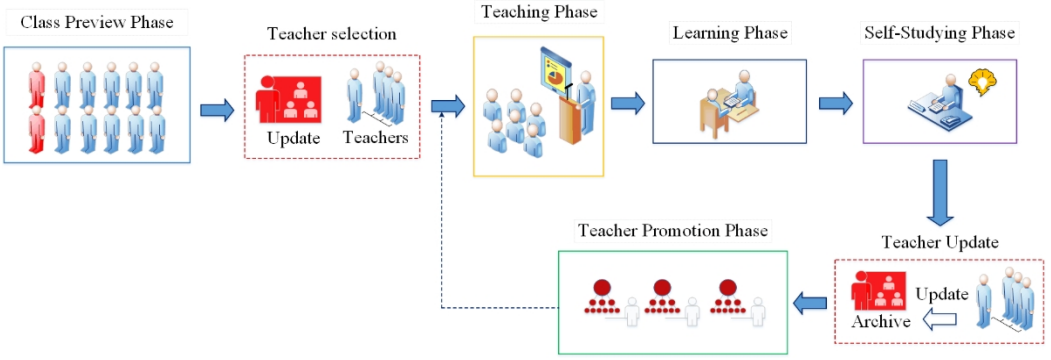


FIGURA 3. Estrutura do algoritmo IMOTLBO.

onde j, j^r e $[1, J]$, i e $[1, n_j]$, i^r e $[1, n_{(j)^r}]$, com $\mu_{ji} = \mu_{j^r i^r}$

Essas restrições garantem que o FIFO (First In First Out) A regra de gerenciamento de buffer é aplicada no sistema RC estudado, então todas as operações sucessivas são processadas na mesma ordem na estação, incluindo buffer de entrada, máquina e buffer de saída.

$$C_{max} = Max(C_j) \tag{17}$$

$$E_j = Max(0, a_{(j)} - C_{(j)}) \tag{18}$$

$$T_j = Max(0, C_{(j)} - b_{(j)}) \tag{19}$$

onde j e $[1, J]$

As restrições (16), (17) e (18) calculam o makespan, a antecipação e o atraso de cada trabalho, respectivamente.

III. METAHEURÍSTICA PROPOSTA

A otimização baseada no ensino-aprendizagem (TLBO) é um novo algoritmo de otimização inteligente proposto por Rao *et al.* [29]. O algoritmo TLBO é um tipo de meta-heurística baseada em população que simula o processo de ensino-aprendizagem. No TLBO padrão, há dois estágios vitais que são a fase de ensino e a fase de aprendizado. O algoritmo pode ser simplesmente descrito como se os alunos aprendessem com o professor (o melhor aluno da população atual na fase de ensino). Ao mesmo tempo, eles aprendem com outros alunos na fase de aprendizado.

Neste trabalho, é proposto um algoritmo de Otimização Baseada em Ensino e Aprendizagem Multiobjetivo Aprimorado para resolver o problema estudado, em que a fase de visualização da aula, a fase de autoestudo e a fase de promoção do professor são incorporadas. A estrutura do algoritmo IMOTLBO é mostrada na Figura 3, e os detalhes são descritos a seguir.

A. ESQUEMA DE CODIFICAÇÃO

O esquema de codificação é um elemento essencial de qualquer algoritmo metaheurístico. Neste estudo, uma solução para o problema (aluno ou professor) é representada por uma matriz ($len \times 2$) em que len é a soma do número total de operações e do número de trabalhos (cada trabalho tem uma pseudooperação para mover o trabalho para a estação de descarga). A primeira linha representa o índice do trabalho, a linha

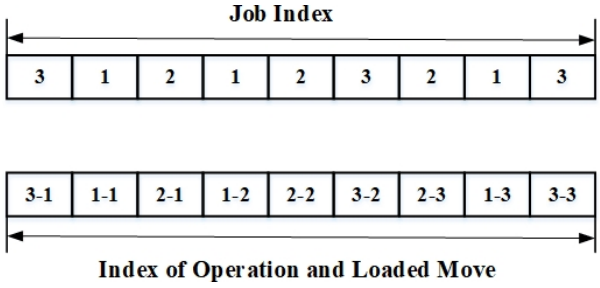


FIGURA 4. Representação individual.

o número de ocorrências do trabalho j é igual a $n_j + 1$. A segunda linha representa a sequência de programação em que cada elemento indica o índice da operação O_{ji} e da movimentação carregada Tr_{ji}.

Um exemplo do problema estudado com 3 trabalhos é mostrado na Figura 4. Nesse exemplo, uma solução é representada como [3 1 2 1 2 3 2 1 3] na primeira linha, em que cada trabalho tem duas operações de processamento. Na posição relativa da segunda linha, o primeiro número "3-1" representa a operação O_{31} e um movimento carregado Tr_{31} que transporta o trabalho 3 da estação de carga para sua primeira estação de processamento. O segundo número '1-1' representa a operação O_{11} e o movimento carregado Tr_{11}, uma vez que o robô de manuseio parou em M_2 na visualização carregada

mover Tr_{31}, um movimento vazio Tr^r_{11} é necessário antes de Tr_{11} a fim de mover o robô para a estação inicial de Tr_{11}.

O número "3-3" é uma pseudooperação que indica que o trabalho 3 retorna à estação de descarga. As informações na segunda linha não apenas representam a sequência de programação das operações, mas também indicam a sequência de transporte do robô de manuseio. Um gráfico de Gantt do exemplo apresentado é mostrado na Figura 5.

B. INICIALIZAÇÃO DA FASE DE PRÉ-VISUALIZAÇÃO DA CLASSE-POPULAÇÃO

No processo de ensino, se os alunos tiverem um bom nível de conhecimento antes das atividades de ensino, isso será benéfico para a qualidade do ensino da turma. Essa parte é definida como a fase de visualização da turma, responsável por gerar a população inicial.

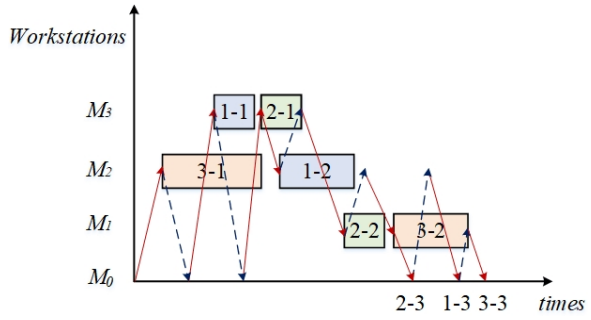


FIGURA 5: Gráfico de Gantt de um exemplo.

Neste estudo, uma conhecida heurística NEH (Nawaz, Enscore e Ham) [30] e a regra de prioridade de despacho SPT (Shortest Processing Time) [31] são aplicadas para gerar duas soluções iniciais, a fim de acelerar a convergência do algoritmo. As outras soluções na população inicial são geradas aleatoriamente.

C. SELEÇÃO DE PROFESSORES - CLASSIFICAÇÃO DO PARETO

Como nosso problema estudado é uma otimização com vários objetivos, mais de um professor é selecionado da classe de acordo com o nível de conhecimento pessoal e o desempenho. No algoritmo IMOTLBO, o nível de conhecimento pessoal e o desempenho referem-se aos valores objetivos e ao nível de classificação da solução. Cada professor tem suas próprias vantagens, ninguém pode superar completamente o outro. Esse processo é baseado na relação de dominância de Pareto. Um mesmo método de classificação de Pareto do NSGA-II é aplicado, todas as soluções são divididas em várias frentes de Pareto, a primeira frente de Pareto é considerada como os professores e são copiadas em um arquivo.

D. FASE DE ENSINO E FASE DE APRENDIZADO-CROSSOVER

Na etapa de visualização, os melhores indivíduos são identificados como professores. Os professores tentam melhorar o desempenho dos alunos por meio da comunicação de conhecimento. Os alunos obtêm conhecimento dos professores usando o operador de crossover. A técnica PTL [32] é aplicada nessa seção e pode produzir um par de permutações diferentes, mesmo a partir de dois indivíduos idênticos (veja a Figura 6). As novas soluções geradas são comparadas com a original, e a melhor (que domina as outras) será aceita. Se não houver uma relação dominante entre elas, uma solução escolhida aleatoriamente será considerada como o aluno.

Como a sequência de processamento das operações em um trabalho não pode ser alterada, a ordem das operações deve ser atualizada por um processo de reparo após a modificação. Em geral, enquanto a solução for modificada, a sequência de operações dos trabalhos deve ser verificada novamente pelo processo de reparo para garantir a viabilidade da solução.

Na fase de aprendizado, os alunos melhoram seu desempenho por meio da comunicação entre eles. Um aluno X interage com outro aluno y, que é escolhido aleatoriamente. Esse

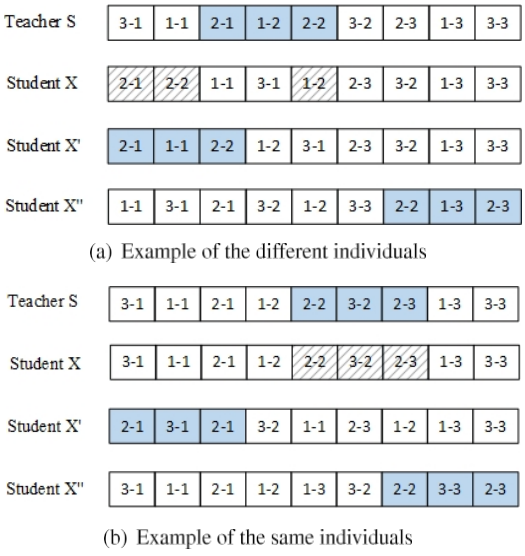


FIGURA 6. Técnica de crossover PTL.

O cruzamento é realizado entre os alunos da mesma forma que a mostrada na Figura 6.

E. MUTAÇÃO DE FASE DE AUTOESTUDO

Após a comunicação de conhecimento da classe, cada aluno precisa revisar seu conhecimento por meio de estudo autônomo. Nessa fase, os alunos realizam o autoaperfeiçoamento por meio de um operador de mutação. A cada vez, uma das estruturas de vizinhança mencionadas abaixo (veja a Figura 7) é selecionada aleatoriamente como o operador de mutação.

F. FASE DE PROMOÇÃO DE PROFESSORES - PESQUISA LOCAL

Nas atividades de ensino, os professores continuam a aprimorar seus conhecimentos e habilidades, o que é benéfico para o desempenho de toda a turma. O objetivo é melhorar ainda mais as soluções ideais atuais de toda a população e do arquivo. fase, um algoritmo de pesquisa local é aplicado para melhorar os indivíduos no arquivo ao final de cada iteração.

Normalmente, o método de pesquisa local recebe uma solução como entrada e retorna uma solução como saída. Neste trabalho, todos os indivíduos do arquivo são considerados como entrada da pesquisa local e retornam um conjunto de soluções não dominadas. Isso é feito para manter a diversidade do algoritmo. O IMOTLBO implementa uma técnica de busca local com base no algoritmo VND (variable neighbor hood descent) [33], que move continuamente uma solução para outra posição no espaço de busca. Em nosso implementação, usamos l_{max}= 4 estruturas de vizinhança diferentes (consulte a Figura 7), essas estruturas de vizinhança são explicado resumidamente da seguinte forma:

- Troca: Essa estrutura de vizinhança troca a posição de duas operações.
- Reversão: Essa estrutura inverte a ordem das operações entre duas posições selecionadas aleatoriamente.
- Inserção de ponto único: Essa estrutura remove uma operação de sua posição atual e a realoca em uma posição aleatória.

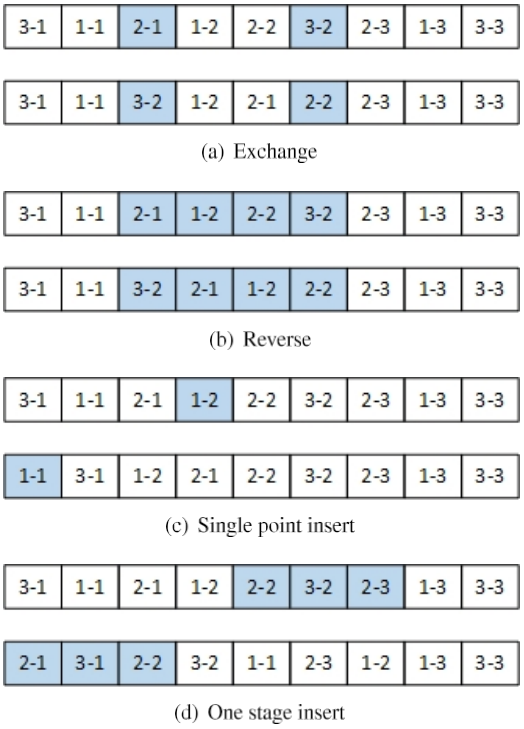


FIGURA 7. Exemplo de estruturas de vizinhança.

- One Stage Insert: nessa estrutura, uma parte das operações entre duas posições aleatórias é removida para outra posição.

Inspirado no algoritmo de busca local para o critério E/T de Gao *et al*. [24], é proposto um método de troca de antecipação e atraso para melhorar o indivíduo. De acordo com o tempo de conclusão e o intervalo de data de vencimento esperado dos trabalhos, os trabalhos são divididos em três grupos nesse método. Esses três grupos incluem os trabalhos concluídos mais cedo, os trabalhos concluídos mais tarde e os trabalhos concluídos dentro do prazo, respectivamente. Nesse método, duas operações adjacentes na mesma estação são consideradas como pares das duas primeiras operações. Para todas as operações em todas as estações, se a operação da esquerda no par pertencer ao trabalho do grupo um e a operação da direita no par pertencer ao trabalho do grupo dois, a posição do par de operações na estação será trocada. Por exemplo, conforme mostrado na Figura 5, suponha que o tempo de conclusão do trabalho 2 seja antes de sua primeira data de vencimento a_2, o tempo de conclusão do trabalho 3 seja depois de sua última data de vencimento b_3e o tempo de conclusão do trabalho 1 esteja entre a_1e b_1. As operações O_{22}e O_{32}trocarão suas posições.

O pseudocódigo da técnica de pesquisa local proposta é mostrado no Algoritmo 1. A busca local recebe os indivíduos no arquivo como entrada. A primeira etapa é inicializar as populações auxiliares *pop* e *localpop*. Em seguida, o processo de pesquisa local é aplicado a cada indivíduo no *Arch*. Diferentemente da técnica de busca local padrão, nossa busca local aprimorada é usada para gerar várias soluções não dominadas. No início de cada iteração, a $i_{ésima}$solução em *Arch* é selecionada e considerada como *s*, a população temporária *localpop* é inicializada com a solução *s*. Definimos *l* como 1 e

Algoritmo 1 Pseudocódigo do método de pesquisa local

Entrada: Os indivíduos no arquivo *Arch*
Output: Os descendentes Arquivo **Arch**
Início
  pop ▪ *φ*
  localpop ▪ *φ*
  for i= 1 to nArch *s* ▪
    Arch(*i*)
    localpop ▪ *s l*
    ▪ 1
    enquanto *l*< l_{max}**do**
      s^r' implement l_{th}neighborhood move(s)
      se *s* for dominado por s^r**, então**
        s ▪ s^r, *f* (*s*) ▪ *f* (s^r)
        update(*localpop, s*^{r})
        l ▪ 1
      elseif s e s' são soluções não dominadas **do**
        update(*localpop, s*^{r})
        l ▪ 1
      mais
        l ▪ *l*+ 1
      final
    finalizar enquanto
    s^r' *EarlineesTardinessSwap*(*s*)
    update(*localpop, s*^{r}) update(*pop, localpop*)
    localpop ▪ *φ*
  final
  Arco ▪ *pop*
  retornar *Arch*
Fim

iniciam o seguinte processo de pesquisa. Quando uma nova solução s^ré gerada pela implementação das estruturas de vizinhança. Ao comparar s^re a solução original *s*, há três possibilidades: 1) s^rdomina *s*, então *s* é substituída por s^re
localpop é atualizado, *l* é redefinido como 1 para reiniciar a pesquisa
processo com a primeira estrutura de vizinhança. 2) não há relação de dominância entre *s* e s^r, então, *localpop* é atualizado e *l* é redefinido para 1. 3) s^ré dominado por *s*, então definimos *l* *l*= + 1 para implementar a próxima estrutura de vizinhança. Quando a pesquisa mencionada acima for concluída, a antecipação-atraso

A troca é aplicada e atualiza *o localpop* com a solução gerada s^r. No final de cada iteração, a população auxiliar *pop* é atualizada por *localpop*, e *localpop* é apagada. Quando todos os
soluções em *arco* são selecionadas para realizar o processo de busca, o *pop* final será considerado como o arquivo de descendência.

G. PROCEDIMENTO DE IMOTLBO

Os detalhes do IMOTLBO proposto para resolver o MOJRCSP são apresentados no Algoritmo 2. O algoritmo começa com uma classe de alunos PS, em que PS é o número de alunos da classe. Cada aluno corresponde a uma solução do problema estudado e tem seu próprio nível de conhecimento.

Alguns deles terão melhor desempenho do que outros ao executar a fase de visualização da aula. Como há dois objetivos a serem considerados, a classificação de Pareto é aplicada para selecionar os professores (soluções não dominadas) que são salvos em uma equipe de professores (arquivo). Em seguida, todos os alunos começam a estudar continuamente até que o critério de parada seja satisfeito.

Em cada iteração, os alunos aprendem com um professor e se comunicam com outros alunos. Essa etapa é realizada por meio da fase de ensino e da fase de aprendizado, em que o cruzamento PTL é executado. Após a comunicação em sala de aula, cada aluno tenta melhorar seu desempenho usando a fase de autoestudo, em que quatro estruturas de vizinhança são fornecidas como operador de mutação. Quando todos os alunos tiverem concluído suas tarefas de aprendizagem nessa iteração, a equipe de professores será atualizada de acordo com a relação de dominância de Pareto.

Posteriormente, todos os professores iniciam seus estudos adicionais para melhorar ainda mais seu desempenho. Na fase de promoção do professor, a busca local proposta é implementada em todas as soluções do arquivo e gera um arquivo de descendentes com um conjunto de soluções não dominadas. Essa etapa pode acelerar a convergência do IMOTLBO. Quando o algoritmo é encerrado, as soluções do professor no arquivo são consideradas como a saída do algoritmo.

Algoritmo 2 Pseudocódigo do IMOTLBO

Entrada: Dados do problema
Output: Os melhores indivíduos
Início
Inicializar a população (fase de visualização da classe) Avaliar a aptidão (cálculo dos valores objetivos) Identificar as soluções não dominadas (seleção do professor)
tion)
Enquanto *o critério de parada não for satisfeito*, **DO**
Cruzamento entre alunos e professores (Ensino
fase)
Crossover entre alunos (fase de aprendizado) Mutação para alunos (fase de autoestudo) Atualização de professores
Pesquisa local de professores (promoção de professores)
fase)
End While
Saída dos professores
Fim

IV. RESULTADOS COMPUTACIONAIS E COMPARAÇÕES

A. CONFIGURAÇÃO EXPERIMENTAL

Para avaliar o desempenho do algoritmo IMOTLBO proposto, foram realizadas avaliações experimentais e comparações com outros métodos. Dois conjuntos de instâncias de problemas

são considerados:

- O primeiro conjunto de dados (instâncias 1-12) é fornecido por Cau- mond *et al.* [7].
- O segundo conjunto de dados (instâncias 13 a 30) é gerado de forma ranqueada pelo protocolo.

O conjunto de benchmark Caumond inclui 12 instâncias de tamanho pequeno, com tamanho variando de 3 trabalhos e 12 operações a 5 trabalhos e 18 operações, e o número de máquinas é igual a 5. O conjunto de dados gerado é composto por 18 instâncias de tamanho médio-grande, com tamanhos que variam de 3 trabalhos, 12 operações e 5 máquinas a 20 trabalhos, 110 operações e 10 máquinas. Neste estudo, o tempo de processamento p_{ji}é gerado aleatoriamente entre uma distribuição uniforme [1, 50]. A duração da viagem carregada τ_{ri}e a duração da viagem vazia σ_{ri}são geradas da mesma forma que em [7]. O tempo de carregamento ε_ce o tempo de descarregamento $\varepsilon_d$são definidos como 1. Inspirado por Gao *et al.* [24], as fórmulas a seguir são usadas para gerar a data de vencimento mais cedo e a data de vencimento mais tarde, respectivamente.

$$a_j = (1+\frac{T\times n}{m})\times \sum_{i=1}^{n_j} p_{(ji)} \quad (20)$$

$$b_j = (2+\frac{T\times n}{m})\times \sum_{i=1}^{n_j} p_{(ji)} \quad (21)$$

em que T é um parâmetro igual a 0,3, *n* é o número de trabalhos, *m* é o número de máquinas.

B. MÉTRICAS DE MEDIÇÃO

A comparação dos resultados da otimização multiobjetivo é diferente do método de objetivo único, porque há um conjunto de soluções não dominadas, em vez de uma única solução. Várias condições devem ser respeitadas para os resultados da pesquisa do método aproximado: a distância até a frente ideal absoluta de Pareto deve ser minimizada, a diversidade das soluções obtidas deve ser maximizada e a dispersão das soluções obtidas também deve ser maximizada. As seguintes medidas de desempenho amplamente usadas são aplicadas neste documento:

- Número de soluções não dominadas

Essa medida de desempenho conta o número total de soluções não dominadas dos algoritmos comparados.

- μ_d-distância de Dugardin *et al.* [34]

Essa medida calcula o tamanho da distância entre duas frentes não dominadas. Sejam A e B duas frentes não dominadas a serem comparadas e que são fornecidas por métodos diferentes. n_Ae $n_B$são o número de soluções não dominadas na frente A e na frente B, respectivamente. Então, a distância μ_d
é definido como:

$$\mu_{(d} = \frac{\frac{1}{nB}\sum_{i=0}^{nB} d_i}{\sqrt{(f_{1max} - f_{1min})^2 + (f_{2max} - f_{2min})^2}} \quad (22)$$

onde, $D = \sqrt{(f_{(1)(max)} - f_{(1)(min)})^2 + (f_{(2)(max)} - f_{(2)(min)})^2}$ é a cruz do retângulo correspondente, d_ié a distância entre a
solução de A e sua projeção ortogonal em B. O valor de d_ié negativo quando a solução *i* está abaixo da frente B (ou da frente extrapolada) e positivo caso contrário. Essa medida é a melhoria proporcional à diagonal do retângulo. Ela mostra se a frente A está mais próxima das soluções ideais do que a frente B. Quanto mais negativo for o valor de μ_d, mais A é melhor do que B.

- Medida de Zitzler [35]

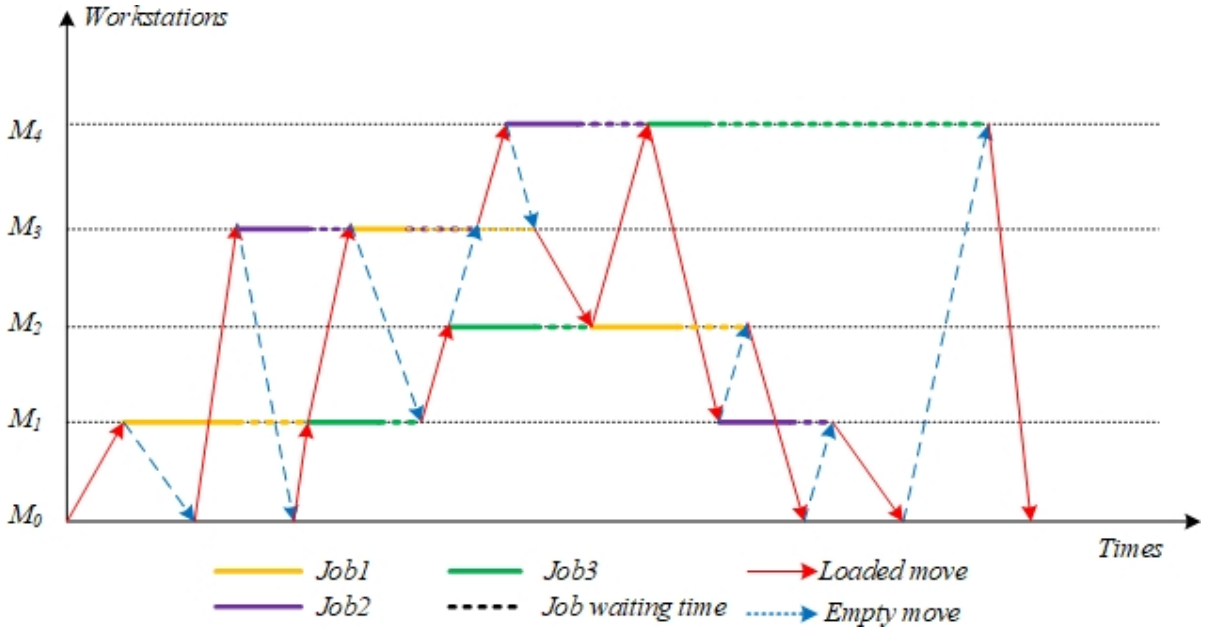


FIGURA 8. Exemplo de gráfico de Gantt para um problema de pequeno porte.

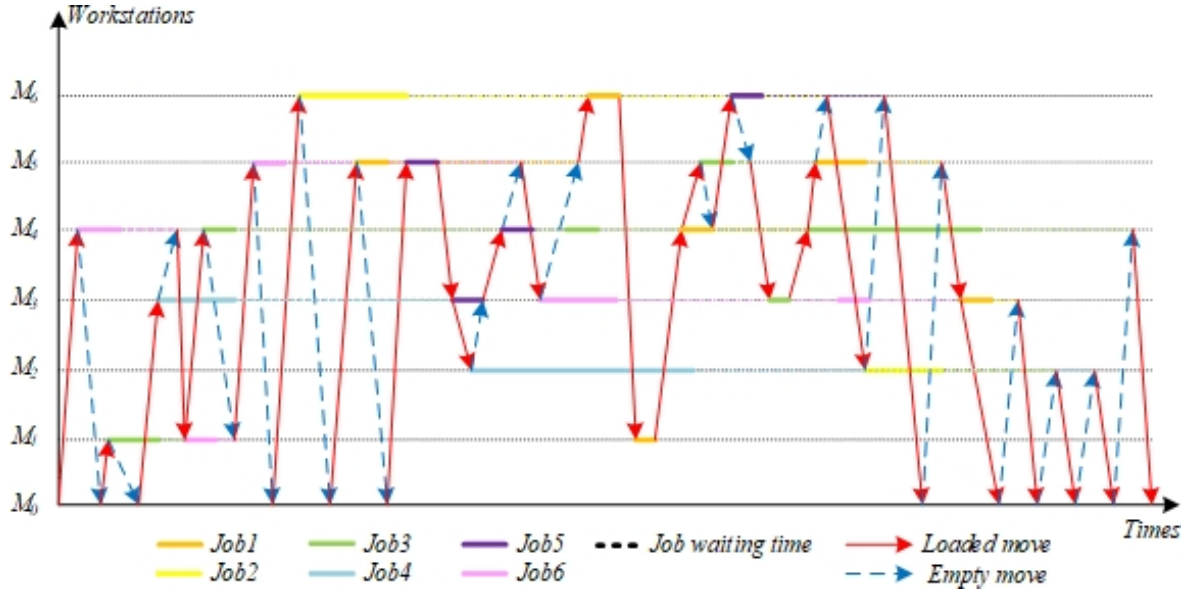


FIGURA 9. Exemplo de gráfico de Gantt para um problema de tamanho médio.

De acordo com a medida de Zitzler, C_1calcula a proporção de soluções em A que são dominadas por pelo menos uma solução em B e vice-versa. C_1é definido por

$$C_1 = \frac{|\{a \in A | \exists\ b \in B : b \succeq a\}|}{|A|} \tag{23}$$

onde a e b são as soluções não dominadas no conjunto A e B, respectivamente.

C. RESULTADOS E COMPARAÇÕES

Para demonstrar o desempenho do programa, o

O algoritmo IMOTLBO apresentado é comparado com vários algoritmos conhecidos e publicados recentemente, como NSGA-II apresentado por Deb *et al.* [23], a otimização por enxame de partículas multiobjetivo (MOPSO) apresentada por Sha e

Hsu [25] e o agrupamento baseado em Pareto de harmonia discreta (PGDHS) de Gao *et al.* [24]. Todos os algoritmos

TABELA 1. Comparação de desempenho do IMOTLBO e do NSGA-II.

Index	J	M	S	μ_d^*	$\mu_d^\#$	μ_d^b	C_1^*	$C_1^\#$	C_1^b	C_2^*	$C_2^\#$	C_2^b	n_1^*	n_2^*	CT_1	CT_2
1	3	5	12	-0.005	0.0001	-0.019	0.014	0.0006	0	0.337	0.014	0.526	22.3	16.2	3.2	4.2
2	4	5	15	-0.024	0.0001	-0.047	0.010	0.0006	0	0.816	0.0299	1	20.4	10.8	3.5	4.3
3	3	5	12	-0.038	0.0005	-0.086	0.029	0.0019	0	0.783	0.0210	1	29.1	9.6	3.4	4.5
4	3	5	12	-0.003	0.0001	-0.008	0.021	0.0007	0	0.251	0.0190	0.5	35.7	23.1	3.6	4.1
5	5	5	18	-0.090	0.0227	-0.490	0.113	0.0271	0	0.708	0.1045	1	5.7	2.8	3.4	4.6
6	4	5	15	-0.008	0.0001	-0.018	0.077	0.0050	0	0.482	0.0340	0.867	20	14.3	3.8	4.3
7	4	5	15	-0.028	0.0005	-0.085	0.055	0.0052	0	0.696	0.0275	1	16.2	9.8	3.5	4.4
8	4	5	15	-0.146	0.0140	-0.367	0.044	0.0107	0	0.796	0.0497	1	6.7	4.2	3.3	4.7
9	5	5	18	-0.010	0.0001	-0.019	0.079	0.0019	0	0.570	0.0168	0.846	31.7	15.0	3.9	4.5
10	5	5	18	-0.022	0.0003	-0.073	0.041	0.0045	0	0.767	0.0525	1	21.8	10.8	3.8	4.4
11	4	5	15	-0.073	0.0247	-0.458	0.089	0.0273	0	0.680	0.1250	1	5.2	4	3.7	4.6
12	5	5	18	0.004	0.0057	-0.031	0.022	0.0009	0	0.623	0.0288	0.941	34.4	18.9	3.7	4.8
13	3	5	12	-0.060	0.0057	-0.249	0.058	0.0102	0	0.813	0.0435	1	9.9	6.6	4.2	5.5
14	3	5	12	-0.033	0.0005	-0.074	0.015	0.0009	0	0.766	0.0349	1	37.2	9.9	4.4	5.5
15	4	5	15	-0.065	0.0011	-0.141	0.002	0.0001	0	0.901	0.0098	1	29.3	11	4.3	5.4
16	4	5	15	-0.056	0.0034	-0.180	0.061	0.0191	0	0.855	0.0585	1	28.8	9.1	4.3	5.3
17	5	5	18	-0.059	0.0008	-0.146	0.020	0.0014	0	0.873	0.0224	1	29.6	10.5	4.2	5.7
18	5	5	18	-0.048	0.0011	-0.139	0.051	0.0103	0	0.858	0.0242	1	26.3	10.2	4.3	5.6
19	6	5	23	-0.054	0.0011	-0.120	0.010	0.0004	0	0.941	0.0076	1	29.2	10.3	4.8	7.1
20	8	5	30	-0.076	0.0054	-0.232	0.008	0.0005	0	0.913	0.0256	1	32.1	11.9	4.9	7.4
21	8	5	30	-0.021	0.0001	-0.049	0.018	0.0012	0	0.650	0.0276	0.909	20.5	11.3	4.8	7.3
22	6	5	23	-0.011	0.0001	-0.028	0.007	0.0002	0	0.436	0.0366	0.765	31.0	16.4	5.2	7.4
23	6	7	31	-0.214	0.0141	-0.459	0.003	0.0002	0	0.988	0.0030	1	21.3	9.1	5.6	7.5
24	6	7	31	-0.017	0.0007	-0.096	0.077	0.0081	0	0.711	0.0210	1	12.8	6.2	6.6	7.8
25	10	7	47	-0.017	0.0002	-0.048	0.098	0.0086	0	0.648	0.0364	1	20.0	13.9	8.5	10.2
26	10	7	47	-0.026	0.0002	-0.059	0.046	0.0031	0	0.780	0.0356	1	25.5	13.3	8.7	10.5
27	10	7	56	-0.025	0.0002	-0.056	0.050	0.0479	0	0.870	0.0159	1	27.7	10.8	9.4	11.5
28	15	10	72	-0.031	0.0002	-0.077	0.023	0.0016	0	0.829	0.0114	1	22.2	13.7	9.9	12.0
29	15	10	83	-0.023	0.0001	-0.041	0.036	0.0030	0	0.760	0.0207	1	23.9	12.7	9.8	12.3
30	20	10	110	-0.042	0.0003	-0.090	0.015	0.0005	0	0.818	0.0131	1	34.0	11.8	10.0	12.7

Os algoritmos foram codificados em C++ e testados em um computador com CPU Intel I7-8700 de 3,2 GHz e 16 GB de RAM. Cada instância é executado para 20 replicações. Para garantir a imparcialidade das comparações.

Em seguida, o tamanho da população e o número de iterações dos quatro algoritmos são definidos como 40 e 100. Os resultados da comparação são mostrados na Tabela 1-3.

Nessas tabelas, J é o número de trabalhos, M representa o número de máquinas e S é o número de operações totais. da instância testada. Os valores de μ^*, $\mu^\#$e μ^b representa o valor médio, a variância e o melhor valor de μ_d-distância. Os próximos seis valores representam o valor médio, a variação e o melhor valor de C_1e C_2, respectivamente.

TABELA 2. Comparação do desempenho do IMOTLBO e do PGDHS.

Index	J	M	S	μ_d^*	$\mu_d^\#$	μ_d^b	C_1^*	$C_1^\#$	C_1^b	C_2^*	$C_2^\#$	C_2^b	n_1^*	n_2^*	CT_1	CT_2
1	3	5	12	-0.087	0.0056	-0.243	0.007	0.0003	0	0.656	0.0408	1	22.3	8.7	3.2	6.9
2	4	5	15	-0.090	0.0038	-0.234	0	0	0	0.841	0.0291	1	20.4	6.8	3.5	6.8
3	3	5	12	-0.115	0.0089	-0.450	0.012	0.0007	0	0.883	0.0242	1	29.1	7.0	3.4	7.0
4	3	5	12	-0.023	0.0014	-0.143	0.007	0.0001	0	0.420	0.0441	0.875	35.7	12.1	3.6	7.3
5	5	5	18	-0.669	0.0090	-0.857	0	0	0	1	0	1	5.7	2.9	3.4	7.1
6	4	5	15	-0.068	0.0046	-0.262	0.028	0.0023	0	0.723	0.0409	1	20.0	8.8	3.8	6.9
7	4	5	15	-0.111	0.0067	-0.285	0.028	0.0025	0	0.859	0.0363	1	16.2	6.8	3.5	7.4
8	4	5	15	-0.524	0.0577	-0.874	0	0	0	1	0	1	6.7	3.2	3.3	7.5
9	5	5	18	-0.043	0.0031	-0.236	0.031	0.0039	0	0.712	0.0520	1	31.7	9.9	3.9	7.3
10	5	5	18	-0.145	0.0154	-0.546	0.020	0.0037	0	0.915	0.0216	1	21.8	6.4	3.8	7.4
11	4	5	15	-0.621	0.0387	-0.861	0	0	0	1	0	1	5.2	3.4	3.7	7.2
12	5	5	18	-0.058	0.0026	-0.245	0.020	0.0017	0	0.711	0.0231	0.909	34.4	9.5	3.7	7.6
13	3	5	12	-0.342	0.0519	-0.664	0.033	0.0160	0	0.901	0.0593	1	9.9	4.0	4.2	10.1
14	3	5	12	-0.148	0.0157	-0.529	0.013	0.0010	0	0.883	0.0283	1	37.2	9.8	4.4	10.3
15	4	5	15	-0.108	0.0135	-0.569	0	0	0	0.971	0.0063	1	29.3	8.5	4.3	10.3
16	4	5	15	-0.169	0.0120	-0.480	0	0	0	0.969	0.0051	1	28.8	7.2	4.3	10.4
17	5	5	18	-0.142	0.0060	-0.330	0.006	0.0002	0	0.971	0.0035	1	29.6	7.1	4.2	10.2
18	5	5	18	-0.078	0.0036	-0.232	0.009	0.0008	0	0.959	0.0100	1	26.3	7.1	4.3	10.5
19	6	5	23	-0.065	0.0030	-0.168	0.006	0.0002	0	0.964	0.0059	1	29.2	7.5	4.8	10.8
20	8	5	30	-0.066	0.0045	-0.239	0.033	0.0098	0	0.932	0.0150	1	32.1	8.8	4.9	12.5
21	8	5	30	-0.132	0.0067	-0.324	0.002	0.0001	0	0.939	0.0064	1	20.5	7.1	4.8	12.7
22	6	5	23	-0.089	0.0038	-0.247	0.008	0.0004	0	0.758	0.0392	1	31.0	9.0	5.2	13.2
23	6	7	31	-0.206	0.0125	-0.472	0.003	0.0002	0	0.954	0.0099	1	21.3	5.4	5.6	13.8
24	6	7	31	-0.232	0.0164	-0.487	0.007	0.0010	0	0.973	0.0065	1	12.8	4.8	6.6	14.5
25	10	7	47	-0.068	0.0037	-0.233	0.030	0.0031	0	0.794	0.0210	1	20.0	9.0	8.5	18.4
26	10	7	47	-0.056	0.0048	-0.317	0.037	0.0020	0	0.775	0.0300	1	25.5	8.9	8.7	19.6
27	10	7	56	-0.067	0.0055	-0.324	0.007	0.0005	0	0.940	0.0065	1	27.7	8.6	9.4	20.5
28	15	10	72	-0.069	0.0017	-0.139	0.019	0.0010	0	0.882	0.0226	1	22.2	7.6	9.9	21.3
29	15	10	83	-0.061	0.0033	-0.248	0.031	0.0041	0	0.816	0.0420	1	23.9	8.5	9.8	22.1
30	20	10	110	-0.084	0.0034	-0.247	0.009	0.0003	0	0.959	0.0043	1	34.0	6.9	10.0	22.5

n_1^* e n_2^* são os valores médios dos números de não-dominados soluções de dois algoritmos comparados. CT_1e CT_2são médias valores de tempos de computação em segundos.

A primeira linha da Tabela 1 é explicada. O problema é definido com 3 trabalhos, 12 operações e 5 máquinas. O terceiro

As colunas adolescentes a seguir mostram os resultados da comparação entre IMOTLBO e NSGA-II. Como cada instância é testada por 20 vezes, o valor médio da distância μ_d e a variância $\mu_d^\#$ melhor do que ($\mu_d^* < 0$) o NSGA-II. Em uma situação ideal, temos $\mu_d^b = 0.019$. O resultado mostra que $\mu_d^* = -0{,}005$ $= 0.0001$ indicam que IMOTLBO é $= 0.014$, *ou seja,* há apenas 1,4% de soluções fornecidas pelo IMOTLBO

TABELA 3. Comparação do desempenho do IMOTLBO e do MOPSO.

Index	J	M	S	μ_d^*	$\mu_d^\#$	μ_d^b	C_1^*	$C_1^\#$	C_1^b	C_2^*	$C_2^\#$	C_2^b	n_1^*	n_2^*	CT_1	CT_2
1	3	5	12	-0.103	0.0091	-0.264	0.003	0.0001	0	0.879	0.0578	1	22.3	5.4	3.2	5.1
2	4	5	15	-0.135	0.0139	-0.340	0	0	0	0.972	0.0046	1	20.4	5.2	3.5	4.8
3	3	5	12	-0.196	0.0124	-0.381	0	0	0	1	0	1	29.1	6.7	3.4	5.0
4	3	5	12	-0.056	0.0028	-0.157	0	0	0	0.784	0.1062	1	35.7	5.6	3.6	4.6
5	5	5	18	-0.599	0.0321	-0.882	0	0	0	1	0	1	5.7	2.4	3.4	5.3
6	4	5	15	-0.082	0.0100	-0.357	0	0	0	0.767	0.0776	1	20.0	6.7	3.8	5.2
7	4	5	15	-0.093	0.0113	-0.365	0.003	0.0002	0	0.876	0.0267	1	16.2	6.1	3.5	5.2
8	4	5	15	-0.459	0.0444	-0.735	0	0	0	1	0	1	6.7	3.1	3.3	5.1
9	5	5	18	-0.113	0.0057	-0.273	0.005	0.0002	0	0.919	0.0156	1	31.7	7.0	3.9	4.9
10	5	5	18	-0.163	0.0098	-0.353	0.007	0.0006	0	0.964	0.0084	1	21.8	5.6	3.8	5.0
11	4	5	15	-0.375	0.0871	-0.826	0.013	0.0030	0	0.950	0.0225	1	5.2	3.0	3.7	5.2
12	5	5	18	-0.077	0.0050	-0.211	0.006	0.0002	0	0.795	0.0688	1	34.4	6.8	3.7	5.1
13	3	5	12	-0.284	0.0439	-0.644	0	0	0	0.9833	0.0053	1	9.9	3.1	4.2	7.4
14	3	5	12	-0.185	0.0244	-0.577	0.001	0.0001	0	0.992	0.0006	1	37.2	9.4	4.4	7.3
15	4	5	15	-0.167	0.0138	-0.499	0	0	0	0.996	0.0004	1	29.3	9.2	4.3	7.2
16	4	5	15	-0.216	0.0233	-0.561	0	0	0	1	0	1	28.8	8.4	4.3	6.9
17	5	5	18	-0.200	0.0064	-0.390	0	0	0	0.992	0.0011	1	29.6	10.1	4.2	7.0
18	5	5	18	-0.160	0.0138	-0.499	0	0	0	0.982	0.0031	1	26.3	8.1	4.3	7.4
19	6	5	23	-0.136	0.0141	-0.437	0	0	0	1	0	1	29.2	8.3	4.8	8.2
20	8	5	30	-0.072	0.0055	-0.275	0	0	0	0.979	0.0031	1	32.1	9.1	4.9	8.1
21	8	5	30	-0.170	0.0093	-0.364	0	0	0	1	0	1	20.5	4.2	4.8	8.4
22	6	5	23	-0.150	0.0154	-0.461	0	0	0	0.880	0.0591	1	31.0	4.6	5.2	9.0
23	6	7	31	-0.214	0.0141	-0.459	0.003	0.0002	0	0.988	0.0030	1	21.3	4.4	5.6	9.1
24	6	7	31	-0.279	0.0272	-0.516	0.008	0.0006	0	0.955	0.0145	1	12.8	4.6	6.6	9.3
25	10	7	47	-0.088	0.0064	-0.345	0.009	0.0015	0	0.950	0.0103	1	20.0	5.8	8.5	11.1
26	10	7	47	-0.078	0.0031	-0.239	0	0	0	0.945	0.0208	1	25.5	5.9	8.7	12.8
27	10	7	56	-0.130	0.0120	-0.369	0	0	0	0.995	0.0005	1	27.7	7.5	9.4	13.9
28	15	10	72	-0.086	0.0051	-0.291	0	0	0	0.996	0.0003	1	22.2	6.7	9.9	14.2
29	15	10	83	-0.134	0.0110	-0.389	0.005	0.0003	0	0.955	0.0122	1	23.9	6.9	9.8	14.5
30	20	10	110	-0.139	0.0080	-0.318	0	0	0	1	0	1	34.0	6.6	10.0	15.8

que são dominadas por pelo menos uma solução do NSGA-II. Como $C_{1,2}^{*}$ = 0,337 sugere que 37,7% das soluções obtidos pelo NSGA-II são dominados por pelo menos uma solução das frentes IMOTLBO. Os pequenos valores de variação de C_1e C_2também provam a conclusão ($C^{\#}$=0.0006 e $C_2^{\#}$ = 0.014). Os números de soluções não dominadas mostram que o IMOTLBO obteve mais soluções não dominadas do que o NSGA-II, com valores médios respectivos de 22,3 e 16.2. Ter CT_1menor que CT_2significa que o tempo computacional do IMOTLBO é menor que o do NSGA-II (3,2 segundos) e 4,2 segundos). O significado de cada coluna em cada com-A comparação é a mesma.

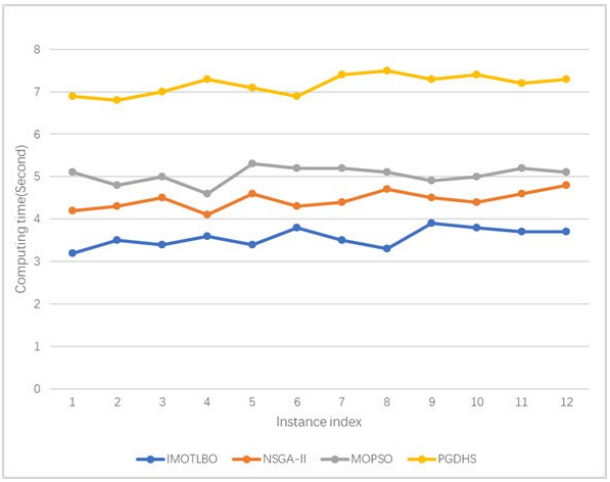


FIGURA 10. Tempos de computação para as instâncias (1)-(12).

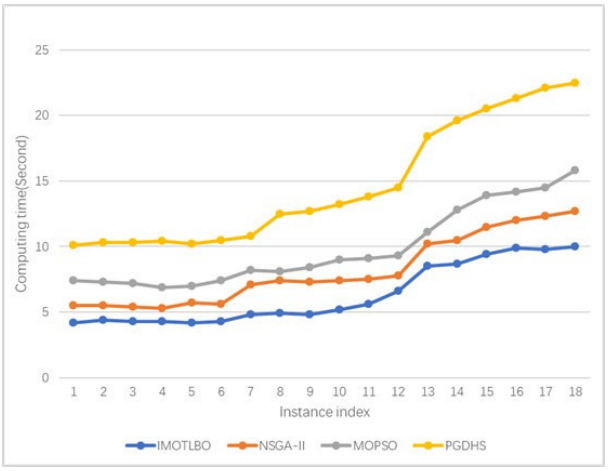


FIGURA 11. Tempos de computação para as instâncias (13)-(30).

Na Tabela 1, podemos chegar à mesma conclusão por meio dos dados de outras linhas, e descobrimos que nosso algoritmo IMOTLBO proposto supera o NSGA-II. Além disso, um caso especial aparece em linha 12, quando$_d\mu^*$=0004, é porque as duas frentes estão muito próximas uma da outra (quando uma se sobrepõe a outra), a d_ié alternativamente positiva e negativa, a medida μ^* não é d significativa [34]). Nesse caso, a dose μ^* não retrata a posição relativa dos dois fatores. As medidas C_1/C_2podem ser usado para realizar a missão Comparison.

Na Tabela 2 e na Tabela 3. O $\mu_d$mostra que o IMOTLBO domina o PGDHS e o MOPSO. O par C_1/C_2mostra que menos soluções do IMOTLBO são dominadas por pelo menos uma solução do PGDHS e do MOPSO ($C_1<_{C2)}$). O IMOTLBO pode produzir mais soluções não dominadas e exige menos tempo de computação do que os outros algoritmos. Em comparação, o IMOTLBO é melhor do que o PGDHS e o MOPSO. Os resultados da comparação comprovam totalmente a eficácia, a robustez e a diversidade do algoritmo IMOTLBO.

O gráfico de Gantt nos ajuda a observar os resultados da programação de forma intuitiva, incluindo a programação de operações em estações e os movimentos do robô de manuseio. Dois exemplos são mostrados na Figura 8 e na Figura 9, correspondendo a um problema de pequeno porte e a um problema de médio porte, respectivamente. Nessas figuras, o eixo X representa os tempos e o eixo Y representa as estações ou máquinas.

Por fim, comparamos os tempos de computação de todos os algoritmos por gráficos. Os tempos de computação dos quatro algoritmos no benchmark Caumond são mostrados na Figura 10, e a Figura 11 mostra os tempos de computação em nossas instâncias geradas. Para todas as instâncias testadas, o algoritmo IMOTLBO mostra sua vantagem no tempo de computação.

V. CONCLUSÃO

Neste artigo, nosso foco é solucionar o problema de programação de células robóticas em lojas de trabalho com múltiplos objetivos (MOJRCSP), e um modelo de programação inteira mista é proposto para modelar o problema estudado. O objetivo é minimizar o makespan e o total de adiantamentos e atrasos simultaneamente. É proposto um algoritmo aprimorado de otimização com base no ensino-aprendizagem multiobjetivo, no qual cinco fases são implementadas para melhorar o desempenho do algoritmo. Por meio de estudos comparativos com outros algoritmos de otimização multiobjetivo, o método proposto supera os demais.

REFERÊNCIAS

[1] S. Gürel, H. Gultekin e V. E. Akhlaghi, "Programação consciente de energia de um robô de manuseio de materiais em uma célula de fabricação", *Robot. Comput.- Integr. Manuf.*, vol. 58, pp. 97-108, ago. 2019.

[2] R. Abaza, D. D. Eun, M. Gallucci, I. S. Gill, M. Menon, A. Mottrie e A. Shabsigh, "Robotic surgery for renal cell carcinoma with vena caval tumor thrombus", *Eur. Urol. Focus*, vol. 2, no. 6, pp. 601-607, dez. 2016.

[3] R. Gerbers, M. Mücke, F. Dietrich e K. Dröder, "Simplifying robot tools by taking advantage of sensor integration in human collaboration robots", *Procedia CIRP*, vol. 44, pp. 287-292, Mar. 2016.

[4] M. Foumani, A. Razeghi e K. Smith-Miles, "Stochastic optimization of two-machine flow shop robotic cells with controllable inspection times: Da teoria à prática", *Robot. Comput.-Integr. Manuf.*, vol. 61, fev. 2020, Art. no. 101822.

[5] P. Yan, S. Q. Liu, T. Sun e K. Ma, "Uma abordagem de programação dinâmica para otimizar as operações de manuseio de materiais em uma célula robótica", *Comput. Oper. Res.*, vol. 99, pp. 166-177, nov. 2018.

[6] A. Elmi e S. Topaloglu, "Cyclic job shop robotic cell scheduling prob lem: Ant colony optimization", *Comput. Ind. Eng.*, vol. 111, pp. 417-432, set. 2017.

[7] A. Caumond, P. Lacomme, A. Moukrim e N. Tchernev, "An MILP for scheduling problems in an FMS with one vehicle", *Eur. J. Oper. Res.*, vol. 199, no. 3, pp. 706-722, Dez. 2009.

[8] A. Che, H. Hu, M. Chabrol e M. Gourgand, "A polynomial algorithm for multi-robot 2-cyclic scheduling in a no-wait robotic cell", *Comput. Oper. Res.*, vol. 38, no. 9, pp. 1275-1285, set. 2011.

[9] M. Nattaf, S. Dauzère-Pérès, C. Yugma e C.-H. Wu, "Parallel machine scheduling with time constraints on machine qualifications", *Comput. Oper. Res.*, vol. 107, pp. 61-76, jul. 2019.

[10] G. Zhang e K. Xing, "Differential evolution metaheuristics for dis- tributed limited-buffer flowshop scheduling with makespan criterion", *Comput. Oper. Res.*, vol. 108, pp. 33-43, ago. 2019.

[11] D. Lei, "A Pareto archive particle swarm optimization for multi-objective job shop scheduling", *Comput. Ind. Eng.*, vol. 54, no. 4, pp. 960-971, maio de 2008.

[12] G. Wang, L. Gao, X. Li, P. Li e M. F. Tasgetiren, "Energy-efficient dis- tributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm", *Swarm Evol. Comput.*, vol. 57, Sep. 2020, Art. no. 100716.

[13] M. Kurdi, "An improved island model memetic algorithm with a new coop- eration phase for multi-objective job shop scheduling problem", *Comput. Ind. Eng.*, vol. 111, pp. 183-201, set. 2017.

[14] A. Udomsakdigool e V. Khachitvichyanukul, "Ant colony algorithm for multi-criteria job shop scheduling to minimize makespan, mean flow time and mean delayiness", *Int. J. Manage. Sci. Eng. Manage*, vol. 6, no. 2, pp. 116-122, Jun. 2011.

[15] H. Mokhtari e A. Hasani, "An energy-efficient multi-objective opti- mization for flexible job-shop scheduling problem", *Comput. Chem. Eng.*, vol. 104, pp. 339-352, set. 2017.

[16] F. Arnold e K. Sörensen, "Knowledge-guided local search for the vehi- cle routing problem", *Comput. Oper. Res.*, vol. 105, pp. 32-46, maio de 2019.

[17] R. Tavakkoli-Moghaddam, M. Azarkish e A. Sadeghnejad-Barkousaraie, "Um novo algoritmo PSO de arquivo de Pareto multiobjetivo híbrido para um problema de agendamento de job shop com dois objetivos", *Expert Syst. Appl.*, vol. 38, no. 9, pp. 10812-10821, set. 2011.

[18] G. Moslehi e M. Mahnam, "A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimiza- tion and local search", *Int. J. Prod. Econ.*, vol. 129, no. 1, pp. 14-22, Jan. 2011.

[19] R. Zhang, P.-C. Chang, S. Song e C. Wu, "Local search enhanced multi-objective PSO algorithm for scheduling textile production processes with environmental considerations", *Appl. Soft Comput.*, vol. 61, pp. 447-467, Dez. 2017.

[20] Q. Luo, Q. Deng, G. Gong, L. Zhang, W. Han e K. Li, "Um algoritmo memético eficiente para o problema de agendamento de job shop flexível distribuído com transferências", *Expert Syst. Appl.*, vol. 160, dezembro de 2020, Art. no. 113721.

[21] M. Kumar, M. L. Mittal, G. Soni e D. Joshi, "A hybrid TLBO-TS algorithm for integrated selection and scheduling of projects", *Comput. Ind. Eng.*, vol. 119, pp. 121-130, maio de 2018.

[22] V. Togˇan, "Design of planar steel frames using teaching-learning based optimization", *Eng. Struct.*, vol. 34, pp. 225-232, Jan. 2012.

[23] K. Deb, A. Pratap, S. Agarwal e T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182-197, agosto de 2002.

[24] K. Z. Gao, P. N. Suganthan, Q. K. Pan, T. J. Chua, T. X. Cai e C. S. Chong, "Algoritmo de busca harmônica discreta de agrupamento baseado em Pareto para agendamento flexível e multiobjetivo de job shop", *Inf. Sci.*, vol. 289, pp. 76-90, dezembro de 2014.

[25] D. Y. Sha e C.-Y. Hsu, "A new particle swarm optimization for the open shop scheduling problem", *Comput. Oper. Res.*, vol. 35, no. 10, pp. 3243-3261, outubro de 2008.

[26] V. K. Patel e V. J. Savsani, "A multi-objective improved teaching-learning based optimization algorithm (MO-ITLBO)", *Inf. Sci.*, vol. 357, pp. 182-200, agosto de 2016.

[27] Y. Xu, L. Wang, S.-Y. Wang e M. Liu, "An effective teaching-learning- based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time", *Neurocomputing*, vol. 148, pp. 260-268, Jan. 2015.

[28] S. Chinta, R. Kommadath e P. Kotecha, "A note on multi-objective improved teaching-learning based optimization algorithm (MO-ITLBO)", *Inf. Sci.*, vol. 373, pp. 337-350, Mar. 2011.

[29] R. V. Rao, V. J. Savsani e D. P. Vakharia, "Teaching-learning-based optimization: A novel method for constrained mechanical design opti- mization problems", *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303-315, Mar. 2011.

[30] M. Nawaz, E. E. Enscore, Jr. e I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem", *Omega*, vol. 11, no. 1, pp. 91-95, Jan. 1983.

[31] O. Holthaus e C. Rajendran, "Efficient dispatching rules for schedul- ing in a job shop", *Int. J. Prod. Econ.*, vol. 48, no. 1, pp. 87-105, jan. 1997.

[32] Q.-K. Pan, M. Fatih Tasgetiren e Y.-C. Liang, "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling prob lema", *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2807-2839, set. 2008.

[33] P. Hansen, N. Mladenović e J. A. M. Pérez, "Variable neighborhood search: Methods and applications", *4OR*, vol. 6, no. 4, pp. 319-360, Dez. 2008.

[34] F. Dugardin, F. Yalaoui e L. Amodeo, "New multi-objective method to solve reentrant hybrid flow shop scheduling problem", *Eur. J. Oper. Res.*, vol. 203, no. 1, pp. 22-31, maio de 2010.

[35] E. Zitzler e L. Thiele, "Multi-objective evolutionary algorithm: A com- parative case study and the strength Pareto approach", *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257-271, Nov. 1999.

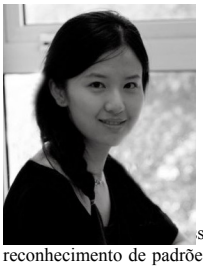
XIAOHUI LI (Membro, IEEE) nasceu em Xi'an, Shaanxi, China, em 1982. Recebeu o título de Bacharel em pela Universidade de Tecnologia de Xi'an, China, em 2004, o título de Mestre em Engenharia pela Universidade Paul Sabatier, França, em 2008, e o título de Doutor pela Universidade de Tecnologia de Troyes, França, em 2011. Desde 2012, ele é professor da Escola de Engenharia Eletrônica e de Controle da Universidade de Chang'an. Seus interesses de pesquisa incluem algoritmos de otimização.

ritmo, de programação, problema de roteamento de veículos, planejamento de caminho de UAV e otimização multiobjetivo.

XI YANG nasceu em Baoji, Shaanxi, China, em 1994. Recebeu o diploma de bacharel em tecnologia de medição e controle e da Universidade Aeronáutica de Xi'an, em 2016. Atualmente, está cursando o mestrado na Universidade de Chang'an. Seus principais interesses de pesquisa incluem algoritmo de otimização inteligente e problema de otimização de programação de oficinas.

YI ZHAO (Membro, IEEE) recebeu o título de M.Eng. da Universidade Pierre e Marie Curie (Universidade Paris VI), em 2009, e o título de Ph.D. da Universidade do Sul, Toulon- Var, em 2013. Desde 2014, ele trabalha Escola de Engenharia Eletrônica e de Controle da Universidade de Chang'an. Seus interesses de pesquisa atuais incluem aprendizado de máquina e redes de sensores.

YING TENG nasceu em Shehong, Sichuan, China, em 1967. Recebeu o título de mestre em economia e gestão pela University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, em 1989, e o título de Ph.D. em ciência e engenharia de gestão pela Southwest Jiaotong University, Chengdu, em 2006. De 2002 a 2008, ela foi professora associada da School of Economics and Management, UESTC, onde é professora ela é diretora do Centro de Estudos Israelenses da UESTC. sa incluem inovação de modelos de negócios no contexto da o de governança de plataforma, preços de acesso, discriminação de preços de terceiro grau com externalidades de rede e setor de telecomunicações.



YUAN DONG nasceu em Xi'an, Shaanxi, China, em 1987. Recebeu o título de bacharel em engenharia eletrônica e da informação pela North Western Polytechnical University, em 2009, e os títulos de mestrado e doutorado em ciência da computação pela Universidade de Tecnologia de Troyes, França, em 2013. Desde 2014, ela vem pesquisando e lecionando na Chang'an Univer- sity. É autora de mais de dez artigos técnicos em revistas especializadas e anais de conferências.

ses de pesquisa incluem aprendizado de máquina, reconhecimento de padrões, redes heterogêneas sem fio, especialmente no estudo de processamento de imagens e sistemas de decisão.