LVI SBPO SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
DE 04 ATÉ 07 NOVEMBRO 2024
FORTALEZA | CEARÁ

Uma heurística para o problema de programação da produção em máquinas paralelas não relacionadas com formação de lotes, famílias de tarefas e tempos de *setup*

Gustavo Alencar Rolim, Caio Paziani Tomazella, Marcelo Seido Nagano
Universidade de São Paulo
Av. Trabalhador São-carlense 400, 13566-590, São Carlos, SP
gustavo.rolim@usp.br, caio.tomazella@usp.br, drnagano@usp.br

RESUMO

Este trabalho é motivado pelas operações de difusão necessárias para a fabricação de semicondutores. Placas de silício são agrupadas em diferentes famílias de produtos, geralmente incompatíveis, e processadas em lotes em máquinas paralelas. O objetivo é a minimização da soma dos tempos de término das tarefas (tempo de fluxo), considerando a existência de *setups* na transição entre as diferentes famílias. Apresenta-se então uma definição formal para o problema e, devido ao fato de ser considerado fortemente NP-difícil, propõe-se uma heurística construtiva. A heurística é comparada com outros algoritmos reportados para problemas semelhantes através da distância com relação a soluções ótimas em instâncias de pequeno porte. Os resultados mostram que a heurística proposta atinge melhores soluções, entretanto, possui um grau de complexidade mais elevado.

PALAVRAS CHAVE. Programação da produção, Sequenciamento, Heurísticas.

AD&GP - PO na Administração e Gestão da Produção, OC – Otimização Combinatória

ABSTRACT

This work is motivated by the diffusion operations necessary for semiconductor manufacturing. Silicon wafers are grouped into different, often incompatible, product families and processed in batches on parallel machines. The objective is to minimize the sum of completion times (flow time), considering the presence of setups during transitions between different families. A formal definition for the problem is then presented, and due to its strong NP-hardness, a constructive heuristic is proposed. The heuristic is compared with other algorithms reported for similar problems through the distance from optimal solutions on small-scale instances. The results show that the proposed heuristic achieves better solutions; however, it has a higher degree of complexity.

KEYWORDS. Production scheduling, Sequencing, Heuristics.

AD&GP – OR in Administration and Production Management, OC – Combinatorial Optimization

https://proceedings.science/p/193906?lang=pt-br

1. Introdução

Estratégias de controle da produção baseadas em regras de despacho são muito comuns no setor de semicondutores. Essas regras geralmente possuem um custo computacional baixo, mas, em contrapartida, oferecem soluções de baixa qualidade. Devido ao alto grau de automação desse setor e ao crescente poder de processamento dos computadores modernos, profissionais da indústria e pesquisadores têm trabalhado no desenvolvimento de heurísticas mais eficientes para o controle e planejamento da produção. Nesse contexto, este trabalho apresenta um novo problema de programação da produção num ambiente de máquinas paralelas não relacionadas em que as tarefas estão segmentadas em famílias incompatíveis, isto é, deve-se formar lotes com tarefas da mesma família e sequencia-los em cada máquina com o objetivo de minimizar a soma total dos tempos de término das tarefas. Considera-se tempos de *setup* distintos dos tempos de processamento das tarefas e dependentes do sequenciamento das famílias nas máquinas. Usando a famosa notação três campos de para problemas de programação da produção e as notações definidas por Rolim e Nagano [2020], o problema estudado pode ser classificado como $R|\mathbf{B}, f, \mathrm{ST_{sd,b}}|\sum C_j$. Vale ressaltar que o caso mais específico de máquina única já é considerado fortemente NP-difícil, portanto, a generalização aqui apresentada também é NP-difícil [Yang et al., 2022].

2. Definição do problema

Deve-se programar um conjunto de tarefas $\mathcal{J} = \{1, \ldots, n\}$ que pertencem a uma de f famílias. O conjunto de famílias é denominado $\mathcal{F} = \{1, \ldots, f\}$ de tal forma que cada família $k \in \mathcal{F}$ consiste num subconjunto $\mathcal{J}_k \subseteq \mathcal{J}$. Mais especificamente, assume-se que $\cup_{k=1}^{f} \mathcal{J}_k = \mathcal{J}$ e $\mathcal{J}_k \cap \mathcal{J}_l = \varnothing$ para cada duas famílias $k, l \in \mathcal{F}$ diferentes $k \neq l$. Cada tarefa $j \in \mathcal{J}$ deve ser processada exatamente em uma máquina de um conjunto $\mathcal{M} = \{1, \ldots, m\}$ de máquinas não relacionadas. Todas as tarefas estão disponíveis em tempo zero e preempções não são permitidas. Além disso, cada tarefa tem um tempo de processamento associado $p_{ij} \in \mathcal{P}$ na máquina $i \in \mathcal{M}$ e um volume $v_j \in \mathcal{V}$. Cada máquina pode processar mais de uma tarefa simultaneamente desde que a capacidade $v_{\max}$ seja respeitada. Assume-se que $0 \leq v_j \leq v_{\max}$.

Um lote factível pode ser definido como um subconjunto de tarefas $\mathcal{B} \subseteq \mathcal{J}$ que podem ser processadas simultaneamente, se e somente se, todas as tarefas pertencerem a mesma família, por exemplo, $\mathcal{B} \subseteq \mathcal{J}_k$ para qualquer $k \in \mathcal{F}$ e o volume total do lote não exceder o volume máximo permitido $\sum_{j \in \mathcal{B}} v_j \leq v_{\max}$. O tempo de processamento de uma tarefa atribuída a uma máquina é definido como $\max_{j \in \mathcal{B}} p_{ij}$ de tal forma que todas as tarefas do lote comecem e terminem o processamento no mesmo instante. Além disso, um tempo de *setup* $s_{ikl} \in \mathcal{S}$ ocorre quando um lote da família l precede imediatamente um lote da família k na máquina i. Também considera-se um *setup* s_{i0k} (tempo de preparação) se um lote da família k é o primeiro a ser processado na máquina i. Uma solução consiste na formação de lotes, na alocação e no sequenciamento dos lotes nas máquinas. Se o tempo de término da tarefa j é C_j, então o objetivo é encontrar uma programação Π que minimize a soma total dos tempos de processamento $\sum_{j \in \mathcal{J}} C_j$.

3. Heurística construtiva

Uma estratégia para gerar boas soluções consiste em verificar duas ações para uma tarefa não programada: a inserção em um lote já existente e a criação de um novo lote. O algoritmo 1 demonstra em detalhes a lógica da heurística proposta. O procedimento começa com a ordenação de cada tarefa $j \in \mathcal{J}$ de acordo com a seguinte regra $\min_{i \in \mathcal{M}}\{p_{ij}\}$. Então, avalia-se a inserção de um novo lote $\mathcal{B}^{\text{new}}$ contendo a tarefa candidata j' (linha 10) ou a inserção de j' em lote já existente na posição $[x]$ (linha 12). Nota-se que as tarefas candidatas são as tarefas com maior prioridade de cada família. O procedimento então salva a tarefa, daqui por diante denominada de j^*, a máquina, a posição e a ação que produzem o menor incremento na função objetivo. Se o custo Z^{job} de inserir j^* em um lote já existente for menor que o custo Z^{batch} de se criar um novo lote, então

j^* é inserido no lote $\mathcal{B}_{[x^*]}$ que gera o menor custo. Caso contrário, um novo lote $\mathcal{B}^*$ é criado e inserido na melhor posição da sequência da melhor máquina Π_{i^*}. A complexidade computacional do algoritmo 1 é limitada por $O(f \cdot m \cdot n^2)$. Entretanto, assumindo que o custo de avaliar cada inserção é executado em $O(n)$ e dois movimentos são executados, então a complexidade total é estimada em $O(2 \cdot f \cdot m \cdot n^3)$.

Algoritmo 1: Construção de solução

Entrada: $\mathcal{F}$, $\mathcal{J}$, $\mathcal{M}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{V}$, $v_{\max}$
Saída : Uma programação Π, $\sum C_j$

$\Pi_i \leftarrow \varnothing \quad i \in \mathcal{M}$;
$\mathcal{B}^{\text{new}} \leftarrow \varnothing$;
Ordenar $\mathcal{J}$ de acordo com $\min_{i \in \mathcal{M}}\{p_{ij}\}, j \in \mathcal{J}$;
/* Encontrar j^* e a ação que produzem o menor incremento em $\sum C_j$ */
enquanto $\mathcal{J} \neq \varnothing$ **faça**
  para $k \in \mathcal{F}$ **faça**
    j' é a tarefa com maior prioridade da família k;
    $\mathcal{B}^{\text{new}} \leftarrow \mathcal{B}^{\text{new}} \cup \{j'\}$;
    para $i \in \mathcal{M}$ **faça**
      para $x = 1$ **até** $|\Pi_i| + 1$ **faça**
        Calcular o custo de $\Pi_i \cup \{\mathcal{B}^{\text{new}}\}$;
        se $v_{j'} + \sum_{j \in \mathcal{B}_{[x]}} v_j \leq v_{\max} \wedge F(j') = F(\mathcal{B}_{[x]})$ **então**
          Calcular o custo de $\mathcal{B}_{[x]} \cup \{j'\}$;
        fim
      fim
    fim
    $\mathcal{B}^{\text{new}} \leftarrow \mathcal{B}^{\text{new}} \setminus \{j'\}$;
  fim
  /* Inserir j^* em um lote existente ou criar um novo */
  se $Z^{job} < Z^{batch}$ **então**
    $\mathcal{B}_{[x^*]} \leftarrow \mathcal{B}_{[x^*]} \cup \{j^*\}$
  senão
    $\mathcal{B}^* \leftarrow \mathcal{B}^* \cup \{j^*\}$;
    $\Pi_{i^*} \leftarrow \Pi_{i^*} \cup \{\mathcal{B}^*\}$
  fim
  $\mathcal{J} \leftarrow \mathcal{J} \setminus \{j^*\}$;
fim

4. Resultados

Os resultados da heurística proposta na Seção 3 são comparados com as heurísticas construtivas FF e IS reportadas em Cakici et al. [2013] e Arroyo et al. [2019], respectivamente. A heurística representada no algoritmo 1 é denominada, daqui por diante, por RTN. Utiliza-se o *benchmark* de Rolim et al. [2024] e a categoria de instâncias C_1. Os algoritmos foram implementados em Julia 1.9 e executados em um MacBook Air com processador ARM M1 e 8GB de memória RAM. As soluções ótimas são obtidas através de um modelo com variáveis posicionais adaptado de Unlu e Mason [2010]. A figura 1 mostra os resultados das heurísticas com relação ao ótimo global

atingido pelo *solver* nas instâncias separadas por número de famílias de tarefas para $f \in \{2, 4, 6\}$. Pode-se observar que a heurística RTN produz os melhores resultados para o conjunto de instâncias.

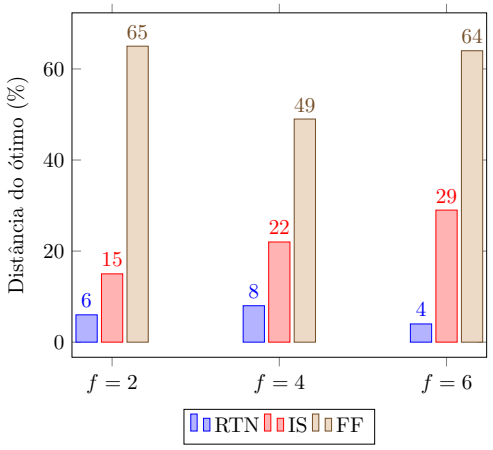


Figura 1: Distância das soluções ótimas nas instâncias do tipo C_1.

5. Conclusões

Este trabalho propôs uma heurística construtiva para o problema de sequenciamento de máquinas paralelas com formação de lotes, famílias de tarefas e tempos de *setup*. A heurística RTN apresenta os melhores resultados com relação ao ótimo global num conjunto de instâncias pequenas ($n = 10$). Isso se deve ao fato de que o método proposto considera tanto a formação de novos lotes quanto a inserção de tarefas em lotes compatíveis. Apesar dos bons resultados, vale ressaltar que heurística RTN também possui um grau de complexidade maior em relação aos algoritmos de comparação FF $O(n \log n)$ e IS $O(m \cdot n^2)$ o que pode resultar em tempos computacionais mais elevados.

Referências

Arroyo, J. E. C., Leung, J. Y. T., e Tavares, R. G. (2019). An iterated greedy algorithm for total flow time minimization in unrelated parallel batch machines with unequal job release times. *Engineering Applications of Artificial Intelligence*, 77:239–254.

Cakici, E., Mason, S. J., Fowler, J. W., e Geismar, H. N. (2013). Batch scheduling on parallel machines with dynamic job arrivals and incompatible job families. *International Journal of Production Research*, 51(8):2462–2477.

Rolim, G. A. e Nagano, M. S. (2020). Structural properties and algorithms for earliness and tardiness scheduling against common due dates and windows: A review. *Computers & Industrial Engineering*, 149:Artigo 106803. ISSN 0360-8352.

Rolim, G. A., Tomazella, C. P., e Nagano, M. S. (2024). Problem instances for scheduling unrelated parallel batch machines with family setups. `https://data.mendeley.com/datasets/nmfsh6ygtk/1`. [Acessado em 7 de maio de 2024].

Unlu, Y. e Mason, S. J. (2010). Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems. *Computers & Industrial Engineering*, 58(4): 785–800.

Yang, F., Davari, M., Wei, W., Hermans, B., e Leus, R. (2022). Scheduling a single parallel-batching machine with non-identical job sizes and incompatible job families. *European Journal of Operational Research*, 303(2):602–615.

https://proceedings.science/p/193906?lang=pt-br