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

Limites de Geração de Colunas para o Problema de Alocação Dinâmica de Berços

Lucas Guilhon, Letícia Caldas e Rafael Martinelli
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Rua Marquês de São Vicente, 225 - Gávea, Rio de Janeiro / RJ - 22.451-900
{lucasxg, leticiacaldas}@aluno.puc-rio.br, martinelli@puc-rio.br

RESUMO

O Problema de Alocação Dinâmica de Berços (DBAP) é fundamental na logística marítima e na gestão portuária, envolvendo a atribuição eficiente de berços aos navios que chegam ao porto. Este estudo explora o DBAP utilizando uma metodologia de geração de colunas, demonstrando sua eficácia na resolução do problema. A abordagem proposta estabelece novos limites inferiores para instâncias ainda não resolvidas na literatura e produz resultados em um tempo significativamente menor comparado ao modelo indexado por tempo. Especificamente, a geração de colunas identificou três novos limites inferiores para essas instâncias, e igual ou o melhor valor conhecido em 76,4% das instâncias restantes. A geração de colunas emerge como uma metodologia promissora para futuras resoluções, conforme evidenciado pelos resultados.

PALAVRAS CHAVE. Alocação Dinâmica de Berços, Geração de Colunas, Logística Marítima.

Tópicos. OC – Otimização Combinatória, PM – Programação Matemática.

ABSTRACT

The Dynamic Berth Allocation Problem (DBAP) is crucial in maritime logistics and port management, involving the efficient assignment of berths to arriving ships at a port. This study investigates DBAP using a column generation methodology, demonstrating its effectiveness in solving the problem. The proposed approach establishes new lower bounds for instances unsolved in the literature and produces results in a significantly shorter time compared to the time-indexed model. Specifically, column generation identified three new lower bounds for these instances and matched the best known value in 76.4% of the remaining instances. Column generation emerges as a promising methodology for future resolutions, as evidenced by the results.

KEYWORDS. Dynamic Berth Allocation. Column Generation. Maritime Logistics.

Paper topics. CO – Combinatorial Optimization, MP – Mathematical Programming.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

1. Introdução

O Problema de Alocação de Berços (do inglês, *Berth Allocation Problem* - BAP) é um problema de otimização comum na logística marítima e na gestão portuária. Envolve determinar a atribuição ideal de berços aos navios que chegam a um porto, considerando diversas restrições e objetivos. O objetivo principal é maximizar a eficiência geral do porto e, ao mesmo tempo, atender às restrições operacionais.

No porto, a disponibilidade de berços é limitada, cada um com suas próprias capacidades e características. Os diferentes navios apresentam uma variedade de requisitos e restrições, incluindo tamanho, calado e tipo de carga transportada. Além disso, há várias restrições operacionais, como tempo, marés e outras considerações, que influenciam a programação e a duração da permanência de um navio no cais. É crucial garantir o cumprimento dos regulamentos de segurança e proteção, pois eles podem afetar diretamente a alocação de berços. A alocação eficiente é fundamental para minimizar o tempo que um navio passa no cais, o que permite um retorno mais rápido e aumenta a produtividade portuária. Para isso, é essencial otimizar a alocação, aproveitando ao máximo os recursos e infraestrutura disponíveis.

A solução do problema de alocação de berços normalmente envolve o uso de técnicas e algoritmos de otimização. Estes podem incluir modelos de programação matemática, métodos heurísticos, algoritmos meta-heurísticos e abordagens de simulação. Vários modelos matemáticos, como formulações de programação linear inteira (do inglês, *Integer Linear Programming* - ILP), são usados para representar formalmente o problema de alocação de berços. No entanto, devido à complexidade e ao tamanho das instâncias do mundo real, métodos exatos podem apresentar dificuldades para encontrar soluções ideais em um período de tempo razoável.

As variações do BAP são conhecidas como problemas NP-difíceis [Monaco e Sammarra, 2007]. Considerando os trabalhos que utilizaram uma abordagem exata para o problema, Monaco e Sammarra [2007] revisaram e compararam cinco diferentes modelos para a formulação dinâmica do problema, considerando os trabalhos de Imai et al. [2001], Cordeau et al. [2005] e Christensen e Holst [2008]. Uma formulação de particionamento de conjuntos generalizada (do inglês, *Generalized Set Partitioning Problems* - GSPP), proposta por Christensen e Holst [2008], se mostrou superior a todas as outras para as instâncias de Cordeau et al. [2005].

Embora a abordagem GSPP seja capaz de resolver problemas de tamanho relativamente grande, a principal desvantagem é a explosão no número de atribuições viáveis de embarcações com o aumento no tamanho do problema. Essa característica faz com que a metodologia de otimização fique sem memória [Saadaoui, 2016].

Diversos trabalhos utilizando abordagens heurísticas, meta-heurísticas e matheurísticas foram publicados. Entre eles, destaca-se Lalla-Ruiz et al. [2012], que utilizaram uma metaheurística híbrida que combina Busca Tabu com religamento de caminhos (do inglês, *Path Relinking* - PR). Lin et al. [2014], propuseram um algoritmo guloso iterado (do inglês, *Iterated Greedy Algorithm* - IG), enquanto Lin e Ting [2014], utilizaram abordagens baseadas no *Simulated Annealing* (SA). LallaRuiz et al. [2015], propuseram uma metaheurística cooperativa descentralizada (do inglês, *Decentralized Cooperative Metaheuristic* - DCM). Por fim, Nishi et al. [2020] propuseram uma matheurística baseada em programação dinâmica.

Considerando os trabalhos que utilizaram Geração de Colunas (GC) em sua abordagem, Robenek et al. [2014] estudaram o problema integrado de alocação de berços e atribuição de pátios no contexto de portos graneleiros. Um algoritmo de solução exato baseado em *branch-and-price* e uma meta-heurística baseada em busca crítica de vizinhança foram propostas para resolver o problema em larga escala. Os autores destacaram que a principal fonte da complexidade do tempo foi a solução dos subproblemas na estrutura de GC.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

Guo et al. [2023] consideraram o BAP com múltiplos portos (do inglês, *Multi-port Berth Allocation Problem* - MPBAP) juntamente com o problema de estabilidade da cooperação portuária (do inglês, *Port Cooperation Stability Problem* - PCSP). O MPBAP considera o desvio de navios com tempos de espera excessivos para portos vizinhos e o PCSP, investiga como agrupar múltiplos portos vizinhos. Para todos os grupos de portos possíveis, um modelo de programação inteira mista (do inglês, Mixed-integer Programming - MIP) para o MPBAP é proposto, e uma abordagem de GC é desenvolvida para resolvê-lo. Com base em soluções ótimas do MPBAP, o PCSP pode ser formulado como um modelo de programação binária para determinar grupos de portos ideais. O algoritmo de GC proposto foi capaz de resolver o modelo MIP para instâncias com quatro portos e 160 embarcações em três minutos.

Saadaoui [2016] abordou a limitação da abordagem GSPP e apresentou uma estrutura de GC onde atribuições são geradas dinamicamente para resolver grandes instâncias do BAP. A autora propôs um algoritmo baseado em geração de colunas e que pode ser facilmente adaptado para resolver qualquer variante do BAP. Experimentos computacionais em um conjunto de instâncias artificiais indicam que a metodologia proposta pode resolver problemas de tamanhos muito grandes até a otimização ou quase otimização em tempo computacional de apenas alguns minutos.

Mauri et al. [2008] propuseram um algoritmo de geração de colunas híbridas para resolver o modelo de Cordeau et al. [2005]. O problema principal foi resolvido pelo modelo de programação linear utilizando a técnica de GC. O subproblema foi resolvido usando uma metaheurística de base evolutiva, chamada algoritmo de treinamento populacional. A abordagem proposta não garante encontrar soluções ótimas para o BAP, porque o subproblema de geração de colunas foi resolvido através de uma heurística. No entanto, os resultados mostram soluções de boa qualidade e obtidas em tempos de processamento razoáveis, em comparação com a Busca Tabu e o CPLEX.

A literatura apresenta os diversos esforços em desenvolver abordagens capazes de solucionar problemas reais, isto é, grandes instâncias, com diversas restrições, em razoável tempo computacional. Esses esforços são justificados pois o BAP é crucial no contexto da gestão portuária, pois impacta diretamente a eficiência, a relação custo-benefício e o desempenho geral das operações logísticas marítimas. A alocação eficiente de berços contribui para reduzir os tempos de espera dos navios, aumentar a produtividade e melhorar a produtividade geral do porto.

Embora várias abordagens de Geração de Colunas (GC) já tenham sido propostas na literatura, nosso trabalho se destaca por apresentar uma metodologia aprimorada para uma variação do BAP, o problema de alocação dinâmica de berços (do inglês, Dynamic Berth Allocation Problem - DBAP). A inovação deste trabalho reside em uma formulação eficiente do subproblema de *pricing* e na utilização de uma relaxação de rotas, que juntas ajudam a reduzir o tempo de resolução total da geração de colunas. Nossa abordagem oferece vantagens em termos de eficiência computacional, especialmente para instâncias de grande porte, proporcionando soluções viáveis em tempos de processamento menores.

Na Seção 2, é apresentada a definição do problema, a formulação de tempo indexada, seguida da geração de colunas desenvolvida. Resultados computacionais e instâncias utilizadas são apresentados na Seção 3 e as conclusões e trabalhos futuros na Seção 4.

2. Metodologia

2.1. Definição do Problema

Diferentes variações do BAP foram propostas na literatura. As restrições no BAP dependem da formulação específica do problema e dos atributos considerados. A representação formal pode ser ajustada com base em características e requisitos específicos do problema. De acordo com Imai et al. [2005] algumas restrições comuns incluem restrições espaciais, como o número e tamanho dos berços e restrições de profundidade da água. De acordo com as posições de atracação

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

viáveis, o problema pode ser classificado como discreto, onde o cais é particionado em uma série de seções, chamadas de berços. No traçado contínuo, não há compartimentação do porto e, quando navios de grande porte podem ocupar mais de um berço ou pequenas embarcações podem compartilhar um berço, temos o layout híbrido.

As restrições temporais podem incluir horários de chegada e partida dos navios, datas de vencimento e janelas de tempo para serviço. As restrições de tempo de manuseio podem considerar tempos de manuseio fixos ou dependentes da posição. De acordo com Imai et al. [2001] e considerando o horário de chegada dos navios, o caso estático assume que os navios já aguardam no porto e podem atracar imediatamente e, para o caso de chegada dinâmica, os navios chegam ao longo do horizonte de tempo.

A variação mais referenciada do problema é a DBAP [Kramer et al., 2019], proposta por Imai et al. [2001]. O DBAP pode ser formalmente definido como um problema de otimização que visa determinar a atribuição mais eficiente de berços em um porto para navios que chegam, sujeito a diversas restrições e objetivos. Uma representação formal do problema considera um conjunto de berços B, e um conjunto de navios N. Cada navio $i \in N$ está disponível para ser atendido em uma dada janela de tempo $[s_i, f_i]$, onde s_i e f_i representam o tempo de chegada e partida do navio, respectivamente. Além disso, cada berço $b \in B$ está disponível para atender navios em um período restrito $[s_b, f_b]$. Cada navio i possui um tempo de serviço ρ_{ib}, que depende do berço de atendimento b, e uma prioridade p_i. O objetivo geral do problema é minimizar o tempo total de fluxo ponderado para atender os navios que chegam, ou seja, o tempo decorrido entre a chegada dos navios ao terminal e a conclusão das operações associadas multiplicado pelos seus valores de prioridade. Observe que uma vez que um navio começou a ser atendido por um berço, seu processamento não pode ser interrompido e reiniciado posteriormente no mesmo ou em outro berço.

2.2. Formulação Indexada no Tempo

A formulação indexada no tempo (do inglês *Time-Indexed* - TI) considera o DBAP como um problema de máquinas paralelas, incorporando datas de início de disponibilidade e datas de entrega para minimizar o tempo total de fluxo ponderado. A disponibilidade dos berços e a compatibilidade entre navios e berços também são consideradas. Segundo Kramer et al. [2019], define-se $u_{ib} = min(f_i, f_b) - \rho_{ib}$, como o tempo limite para a permanência do navio i no berço b, e $l_{ib} = max(s_i, s_b)$, como o início da janela de tempo em que o navio i pode ser atendido no berço b. A formulação TI do DBAP é apresentada a seguir, sendo a variável de decisão x_{ibt} igual a 1 se o navio i é alocado ao berço b no instante de tempo t, e 0 caso contrário.

$$\min \sum_{i \in N} \sum_{b \in B} \sum_{t=l_{ib}}^{u_{ib}} p_i x_{ibt}(t + \rho_{ib} - s_i) \tag{1}$$

sujeito a

$$\sum_{b \in B} \sum_{t=l_{ib}}^{u_{ib}} x_{ibt} = 1 \qquad \forall i \in N \tag{2}$$

$$\sum_{i \in N} \sum_{r=\max(l_{ib}, t+1-\rho_{ib})}^{\min(t, u_{ib})} x_{ibr} \leq 1 \qquad \forall b \in B, t \in [s_b, f_b - 1] \tag{3}$$

$$x_{ibt} \in \{0, 1\} \qquad \forall i \in N, b \in B, t \in [l_{ib}, u_{ib}] \tag{4}$$

A Função Objetivo (1) busca minimizar o tempo total de fluxo dos navios ponderado por suas prioridades, no qual o tempo total de fluxo é calculado como o tempo decorrido entre a chegada

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

dos navios ao berço e a conclusão das operações associadas. Essa modelagem contém um número pseudo-polinomial de variáveis $O(|N| \cdot T)$ e restrições $O(|N| + T)$, em que $T = \sum_{b \in B}(f_b - s_b)$

As Restrições em (2) garantem que cada navio é alocado a exatamente um berço em um instante de tempo. As Restrições em (3) garantem que cada berço possui no máximo um navio alocado em um instante de tempo. Por fim, as Restrições (4) descrevem os domínios das variáveis.

A performance dessa formulação foi testada no trabalho de Kramer et al. [2019]. Para instâncias que possuem até 120 navios, a formulação TI conseguiu encontrar resultados ótimos em uma média de 23 minutos. No entanto, quando se trata de instâncias com mais de 150 navios, o desempenho os resolvedores se deteriora, não conseguindo encontrar soluções ótimas para a maioria das instâncias em um tempo limite de duas horas. Por este motivo, com a intenção da obtenção de melhores limites para as instâncias grandes do problema, o método de geração de colunas da próxima seção é apresentado.

2.3. Geração de Colunas

A fim de se obter melhores limites para o DBAP, uma geração de colunas pode ser desenvolvida para o problema. Seja Ω_b o conjunto de todas as programações viáveis para o berço $b \in B$. O problema principal da geração de colunas utiliza uma formulação do problema de partição de conjuntos (do inglês, *Set Paritioning Problem* - SPP) para encontrar a melhor programação considerando todos os berços. É utilizado um conjunto de variáveis binárias $\lambda_p^b, b \in B, p \in \Omega_b$, onde $\lambda_p^b = 1$ representa a utilização da programação p no berço b, ou zero caso contrário. Além disso, o custo total de uma programação $p \in \Omega_b$ é representado por c_p^b e a constante $\bar{y}_{ip}^b$ indica quantas vezes o navio $i \in N$ é atribuído ao berço b na programação p. Por fim, a formulação do SPP é apresentada a seguir.

$$\min \sum_{b \in B} \sum_{p \in \Omega_b} c_p^b \lambda_p^b \tag{5}$$

sujeito a

$$\sum_{b \in B} \sum_{p \in \Omega_b} \bar{y}_{ip}^b \lambda_p^b = 1 \qquad \forall i \in N \tag{6}$$

$$\sum_{p \in \Omega_b} \lambda_p^b = 1 \qquad \forall b \in B \tag{7}$$

$$\lambda_p^b \in \{0, 1\} \qquad \forall b \in B, p \in \Omega_b \tag{8}$$

$$\tag{9}$$

A Função Objetivo (5) minimiza o custo total das programações utilizadas nos berços. As Restrições (6) garantem que todos os navios são programados exatamente em um berço. As Restrições (7) forçam a utilização de apenas uma programação por berço. Por fim, as Restrições (8) indicam o domínio das variáveis.

A formulação (5)-(8) claramente possui um número exponencial de variáveis, visto a possibilidade de combinações de navios nos berços. Por este motivo, para a resolução do problema principal, um algoritmo de geração de colunas é utilizado, onde apenas um subconjunto das programações de cada berço Ω_b é considerado durante a resolução. A cada iteração do algoritmo, um subproblema de *pricing* é resolvido para cada berço, gerando novas programações e, por consequência, novas variáveis λ_p^b para o problema principal.

Sejam β_i e γ_b as variáveis duais associadas às restrições (6) e (7) respectivamente. O custo reduzido $\bar{c}_p^b$ de uma variável λ_p^b é apresentado na Equação (10).

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

$$\bar{c}_p^b = c_p^b - \gamma_b - \sum_{i \in N} \bar{y}_{ip}^b \beta_i \tag{10}$$

O subproblema de *pricing* pode ser resolvido utilizando uma programação dinâmica, onde o estado de uma programação parcial P de um berço b pode representado pelo rótulo $\mathcal{L}_b(P) = (t(P), \bar{c}(P), \Pi(P))$, sendo $t(P)$ o último instante de tempo da programação parcial P, $\bar{c}(P)$ o custo reduzido acumulado na programação parcial P, e $\Pi(P)$ o conjunto dos navios presentes na programação parcial P. O algoritmo então inicia com a programação parcial trivial vazia $L_b(P_i) = (-\gamma_b, \varnothing)$, e efetua extensões para novas programações parciais, desde que estas sejam viáveis. Uma extensão viável de uma programação parcial P_1 para um navio i gera uma nova programação parcial P_2 definida na Equação (11), caso $t(P_1) + \rho_{ib} < \min(f_i, f_b)$ e $i \notin \Pi(P_1)$.

$$\mathcal{L}_b(P_2) = (max(t(P_1), t_i) + \rho_{ib}, \bar{c}(P_1) + p_i(t(P) + \rho_{ib} - t_i) - \beta_i, \Pi(P_1) \cup \{i\}) \tag{11}$$

De forma a reduzir o número de rótulos gerados durante a execução da programação dinâmica, uma regra de dominância é utilizada, garantindo que um rótulo $\mathcal{L}_b(P_2)$ que pode ser estendido para os mesmos estados que outro rótulo $\mathcal{L}_b(P_1)$, porém com um custo reduzido maior, seja descartado durante a execução do algoritmo. A regra de dominância completa para $\mathcal{L}_b(P_1) \preceq \mathcal{L}_b(P_2)$ é apresentada na Equação (12).

$$\mathcal{L}_b(P_1) \preceq \mathcal{L}_b(P_2) \text{ sse } \bar{c}(P_1) \leq \bar{c}(P_2) \wedge t(P_1) \leq t(P_2) \wedge \Pi(P_1) \subseteq \Pi(P_2) \tag{12}$$

A regra de dominância acima é utilizada para rótulos terminando no mesmo instante de tempo, i.e., rótulos com o mesmo valor de $t(P)$. Esta política pode ser facilmente implementada com a utilização de um vetor contendo uma lista de rótulos para cada possível instante de tempo do horizonte de planejamento. A programação dinâmica então efetua todas as extensões possíveis, seguindo a ordem dos instantes de tempo, utilizando a regra de dominância apresentada, até que nenhum novo rótulo é gerado.

Uma outra forma de melhorar o desempenho do subproblema de *pricing* é a utilização de uma relaxação de rotas, como a relaxação conhecida como *ng*-rotas [Baldacci et al., 2011]. Nesta relaxação, a memória de uma programação parcial P, definida anteriormente como $\Pi(P)$ guarda apenas um subconjunto dos navios visitados seguindo a regra descrita a seguir. Seja uma programação parcial $P = \{v_1, v_2, \ldots, v_p\}$ com p navios visitados, e um subconjunto de navios N_i associado a cada navio $i \in N$, definido como a "memória" do navio i. A memória ng da programação parcial P é então definida na equação (13). Os conjuntos N_i são construídos baseados na proximidade das janelas de tempos dos navios.

$$\Pi(P) = \left\{ v_k \in P \setminus \{v_p\} : v_k \in \bigcap_{s=k+1}^{p} N_{v_s} \right\} \cup \{v_p\} \tag{13}$$

Com o intuito de reduzir ainda mais o tempo de resolução da geração de colunas, uma versão heurística simplificada do algoritmo de *pricing* é utilizada, onde, em vez de armazenar uma lista de rótulos não-dominados para cada instante de tempo no vetor da programação dinâmica, é armazenado apenas o rótulo com o menor valor de custo reduzido, assim não havendo a necessidade da utilização da dominância nesta versão heurística do *pricing* [Martinelli et al., 2014].

O algoritmo de geração de colunas inicia gerando uma solução primal utilizando um algoritmo construtivo simples, e as programações encontradas para cada berço são adicionadas na

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

formulação de partição de conjuntos. A cada iteração, a solução dual é obtida da formulação, e o subproblema de *pricing* heurístico é resolvido para cada berço, gerando um novo conjunto de programações com custo reduzido negativo, que são adicionadas à formulação, iniciando assim uma nova iteração do método principal. Quando nenhuma nova programação de custo reduzido negativo é encontrada para todos os berços, o subproblema de *pricing* exato é resolvido para todos os berços. Caso alguma nova programação seja encontrada, na próxima iteração o método retorna à utilização do *pricing* heurístico. Caso contrário, o algoritmo termina, encontrando assim o valor ótimo para a relaxação linear do problema.

3. Resultados Computacionais

Esta seção apresenta os experimentos computacionais realizados com os modelos TI e de partição de conjuntos resolvido por geração de colunas. Para permitir um comparativo das soluções fracionárias da geração de colunas, o modelo TI será rodado tanto na formulação original quanto com as restrições de integralidade relaxada. Ambos os modelos foram implementados na linguagem de programação Julia 1.10.3 e resolvidos utilizando o solver Gurobi na versão 11.0.2. A execução ocorreu em uma única *thread*, com um tempo limite de uma hora (3.600 segundos), em um sistema equipado com processador Intel Core i7-8700K CPU @ 3.70GHz e 64 GB de RAM.

3.1. Instâncias de Teste

Neste trabalho, consideramos um subconjunto das instâncias propostas pelos trabalhos de Cordeau et al. [2005], Lalla-Ruiz et al. [2012], Nishi et al. [2020] e Kramer et al. [2019]. Essas instâncias variam entre 30 e 250 navios e entre 3 e 20 berços, totalizando 110 instâncias consideradas. Dentre essas instâncias, 20 não possuem solução ótima conhecida até o momento. A tabela 1 apresenta o número de instâncias de teste para cada combinação de navios e berços:

Navios	Berços	Quantidade
200	15	10
250	20	10
30	3	10
	5	10
40	5	10
	7	10
55	5	10
	7	10
	10	10
60	5	10
	7	10

Tabela 1: Número de instâncias de teste para cada combinação de navios e berços.

3.2. Comparação dos Resultados

3.2.1. Instâncias com ótimo conhecido

Os resultados indicam que tanto a geração de colunas quanto o modelo TI relaxado apresentam resultados próximos ao ótimo para as instâncias menores, onde a solução ótima já é conhecida. A Figura 1 apresenta o Gap, que consiste na diferença percentual entre a solução ótima conhecida e a solução obtida pelo modelo, logo, um Gap positivo indica que a solução obtida é inferior à solução ótima. Os valores representam a média dos gaps das instâncias agrupadas por número de navios e berços. No geral, o método de GC apresentou um gap médio de 0.08% e o TI relaxado um gap médio de 0.11%.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

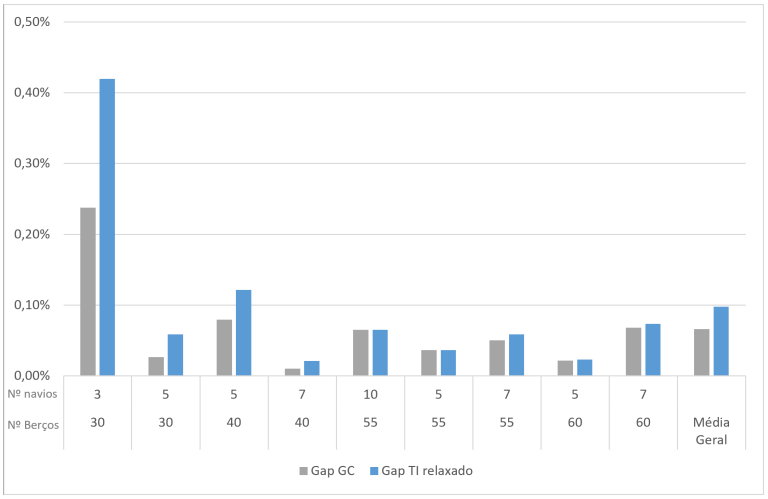

Figura 1: Diferença percentual entre a solução ótima conhecida e a solução obtida pelo modelo
Fonte: Autores, 2024.

Na Figura 2, observa-se o tempo total de execução dos modelos testados. Os valores representam a média do tempo das instâncias também agrupadas por número de navios e berços. No geral, o método de GC apresentou um tempo de médio total $\sim 50\%$ inferior ao tempo médio do modelo TI relaxado.

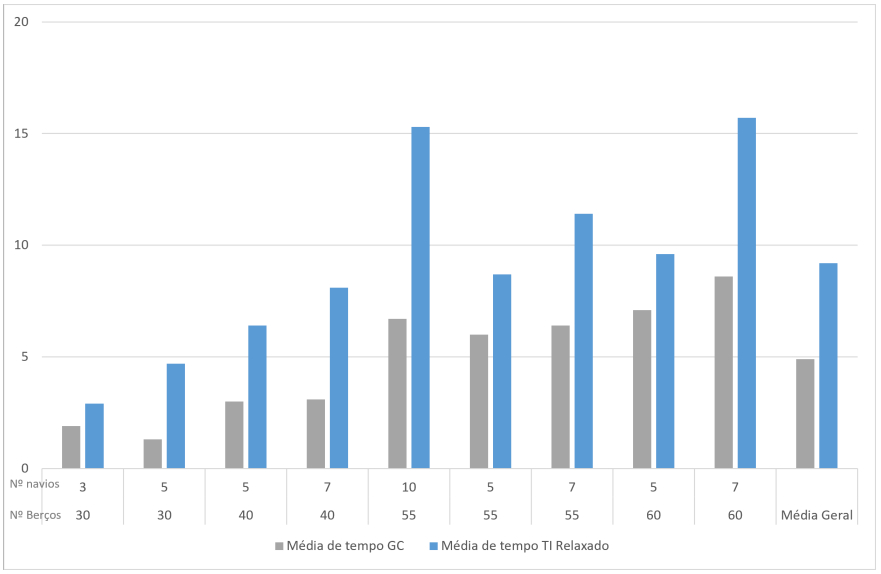

Figura 2: Tempo médio de execução por grupo de instâncias
Fonte: Autores, 2024.

Ambos os métodos apresentaram resultados próximos ao ótimo, o que indica que essas relaxações são fortes. Ademais, a geração de colunas mostrou um desempenho superior ao modelo TI relaxado, alcançando um gap médio menor em todos os grupos, e um tempo médio inferior em todos os grupos.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

3.2.2. Instâncias em aberto

Para as instâncias em aberto, a Tabela 2 é apresentada. Esta tabela reúne os resultados da literatura, dos modelos TI (MIP/relaxado) e da geração de colunas. Como é possível observar, a geração de colunas obteve resultados próximos ao limite inferior da literatura, com um tempo de execução significativamente menor. Em comparação com o modelo TI relaxado, a geração de colunas apresentou resultados de melhor qualidade e, do mesmo modo que também obteve tempos de execução inferiores. Ademais, destacamos na tabela em negrito os resultados em que as abordagens conseguiram atingir[1] os limites da literatura, e sublinhamos aqueles que superaram os limites.

No total, foram três novos limites inferiores encontrados pela geração de colunas, e uma nova instância resolvida pelo modelo TI inteiro (f200x15-05)[2]. Esses novos limites inferiores, e tempos claramente menores, concretizam a superioridade do modelo de partição de conjuntos resolvido por geração de colunas em relação ao modelo indexado pelo tempo relaxado.

Instance	Literatura		TI MIP			TI Relaxado		Geração de Colunas	
	LB	UB	LB	UB	Tempo	Root	Tempo	Root	Tempo
f200x15-01	12604	12609	**12604**	12663	3600.0	**12603.29**	197.2	**12603.29**	34.6
f200x15-02	10319	10319	10318	10354	3600.0	10317.60	218.6	10317.60	35.7
f200x15-03	11296	11355	**11305**	11364	3600.0	11288.14	144.2	**11304.35**	57.2
f200x15-04	15441	15441	15435	15463	3600.0	15431.80	193.7	15435.22	60.8
f200x15-05	18166	18352	**18170**	**18170**	3600.0	18157.25	235.1	**18165.61**	51.0
f200x15-06	16869	16869	**16869**	**16869**	1625.8	16867.05	209.3	**16868.14**	46.8
f200x15-07	13025	13226	**13026**	**13099**	3600.0	13023.01	149.4	**13025.85**	66.2
f200x15-08	14182	14259	**14193**	14330	3600.0	14156.33	191.0	**14192.11**	45.1
f200x15-09	18118	18118	18117	18172	3600.0	18115.79	215.2	18116.30	52.1
f200x15-10	17102	17118	**17103**	17181	3600.0	17093.79	208.0	**17102.67**	42.7
f250x20-01	15633	15769	**15633**	**15765**	3600.0	**15632.16**	344.9	**15632.31**	97.9
f250x20-02	15776	15915	**15776**	15936	3600.0	15774.94	813.3	**15775.14**	112.2
f250x20-03	16519	16606	**16519**	16635	3600.0	**16518.81**	384.7	**16518.81**	96.0
f250x20-04	16423	16481	**16423**	**16457**	3600.0	**16422.64**	446.8	**16422.64**	83.1
f250x20-05	15661	15837	**15661**	15909	3600.0	**15660.97**	578.4	**15660.97**	86.1
f250x20-06	20060	20060	**20060**	**20060**	929.6	**20060.00**	492.8	**20060.00**	93.6
f250x20-07	14284	14362	**14284**	14424	3600.0	**14283.31**	328.2	**14283.31**	101.8
f250x20-08	16305	16383	16304	16483	3600.0	16303.70	448.5	16303.70	94.2
f250x20-09	15864	15917	**15864**	15934	3600.0	**15863.52**	425.0	**15863.81**	94.8
f250x20-10	16283	16371	**16283**	16441	3600.0	**16282.48**	363.3	**16282.48**	92.2

Tabela 2: Comparação entre todos os modelos para instâncias em aberto

4. Conclusões e Trabalhos Futuros

Neste trabalho, abordamos o Problema de Alocação Dinâmica de Berços utilizando uma metodologia de geração de colunas. Os resultados computacionais comprovam a eficácia da geração de colunas para a resolução do DBAP, estabelecendo novos limites inferiores para instâncias ainda em aberto na literatura e obtendo resultados em um tempo consideravelmente menor do que o modelo indexado por tempo.

[1]Os limites inferiores, quando arredondados para o inteiro acima, são iguais ao limite superior da literatura, foram considerados como atingidos.

[2]O modelo TI inteiro conseguiu resolver esta instância devido a melhoria nos resolvedores comerciais desde a última execução dos modelos da literatura.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

Nossa pesquisa demonstra que a geração de colunas é uma ferramenta poderosa para o DBAP, com o potencial de resolver problemas que antes eram intratáveis. Considerando as 20 instâncias em aberto na literatura, a geração de colunas foi capaz de identificar 3 novos limites inferiores e igualou o melhor valor conhecido em 13 das 17 instâncias restantes.

Como um trabalho futuro, recomendamos a exploração de uma abordagem de *Branch-and-Price* para alcançar soluções inteiras para o DBAP. Esta metodologia surge como promissora, uma vez que a geração de colunas produz soluções com uma pequena diferença entre o limite inferior alcançado e o limite superior conhecido, o que é uma característica altamente desejável para a etapa inicial da árvore de *Branch-and-Price*.

Referências

Baldacci, R., Mingozzi, A., e Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. Operations research, 59(5):1269–1283.

Christensen, C. e Holst, C. (2008). Berth allocation in container terminals master's thesis department of informatics and mathematical modelling technical university of denmark.

Cordeau, J.-F., Laporte, G., Legato, P., e Moccia, L. (2005). Models and tabu search heuristics for the berth-allocation problem. Transportation science, 39(4):526–538.

Guo, L., Zheng, J., Liang, J., e Wang, S. (2023). Column generation for the multi-port berth allocation problem with port cooperation stability. Transportation Research Part B: Methodological, 171:3–28.

Imai, A., Nishimura, E., e Papadimitriou, S. (2001). The dynamic berth allocation problem for a container port. Transportation Research Part B: Methodological, 35(4):401–417.

Imai, A., Sun, X., Nishimura, E., e Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. Transportation Research Part B: Methodological, 39 (3):199–221.

Kramer, A., Lalla-Ruiz, E., Iori, M., e Voß, S. (2019). Novel formulations and modeling enhancements for the dynamic berth allocation problem. European Journal of Operational Research, 278 (1):170–185.

Lalla-Ruiz, E., Melián-Batista, B., e Marcos Moreno-Vega, J. (2012). Artificial intelligence hybrid heuristic based on tabu search for the dynamic berth allocation problem. Engineering applications of artificial intelligence, 25(6):1132–1141. ISSN 0952-1976.

LallaRuiz, E., Izquierdo, C. E., Batista, B. M., e MorenoVega, J. M. (2015). Decentralized cooperative metaheuristic for the dynamic berth allocation problem. Inteligencia Artificial, 18(55): 1–11.

Lin, S.-W. e Ting, C.-J. (2014). Solving the dynamic berth allocation problem by simulated annealing. Engineering optimization, 46(3):308–327.

Lin, S.-W., Ying, K.-C., Wan, S.-Y., et al. (2014). Minimizing the total service time of discrete dynamic berth allocation problem by an iterated greedy heuristic. The Scientific World Journal, 2014.

Martinelli, R., Pecin, D., e Poggi, M. (2014). Efficient elementary and restricted non-elementary route pricing. European Journal of Operational Research, 239(1):102–111.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br

Mauri, G. R., Oliveira, A. C., e Lorena, L. A. N. (2008). A hybrid column generation approach for the berth allocation problem. In European Conference on Evolutionary Computation in Combinatorial Optimization, p. 110–122. Springer.

Monaco, M. F. e Sammarra, M. (2007). The berth allocation problem: a strong formulation solved by a lagrangean approach. Transportation Science, 41(2):265–280.

Nishi, T., Okura, T., Lalla-Ruiz, E., e Voß, S. (2020). A dynamic programming-based matheuristic for the dynamic berth allocation problem. Annals of operations research, p. 1–20. ISSN 0254-5330.

Robenek, T., Umang, N., Bierlaire, M., e Ropke, S. (2014). A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. European Journal of Operational Research, 235(2):399–411.

Saadaoui, Y. (2016). The berth allocation problem at port terminals: a column generation framework.

DOI: 10.59254/sbpo-2024-193371
https://proceedings.science/p/193371?lang=pt-br