arXiv:2302.02356v1 [math.OC] 5 Feb 2023

多港口连续泊位分配问题的自适应大规模邻域搜索启发式

Bernardo Martin-Iradi[a], *, Dario Pacino[a], Stefan Ropke[a]

[a]丹麦技术大学 *DTU* 管理学院，*Akademivej 358*号楼，*2800* 孔斯灵比，丹麦

摘要

本文研究一个将船舶调度问题与泊位分配整合为协作问题的模型，称为多港口连续泊位分配问题（MCBAP）。该问题在多个港口同时优化一组船舶的泊位分配，同时考虑船舶在港口之间的航行速度。由于该问题具有高度组合性，精确方法难以扩展到大规模实例，因此有必要探索启发式方法。本文提出了 MCBAP 的混合整数问题公式，并介绍一种结合局部搜索过程的自适应大邻域搜索（ALNS）算法来求解该问题。计算结果表明，该方法适用于更大规模的实例，并能在较短计算时间内获得高质量的解。实践见解表明，燃料价格、港口的外部船舶以及连续岸线的建模方式，会以不同方式影响承运人和码头运营商的运营成本。

关键词：海运业中的运筹学、集装箱码头、泊位分配问题、航速优化、启发式算法

1. 引言

班轮运输业是国际货运的主要方式之一。根据联合国贸易和发展会议（UNCTAD, 2020）的报告，海运贸易和集装箱吞吐量一直稳步增长至2019年。尽管2020年新冠疫情造成了冲击，预计海运贸易将在2021年恢复并增长4.3%。报告还指出，世界船队规模正在扩大，不仅船舶数量增加（2021年增幅超过3%），船舶尺寸也在增大。过去十年间，超大型船舶承载的总运力占比从6%上升至40%。

需求的增长，加上国际海事组织提出的到2050年将航运排放减少50%的目标（国际海事组织，2018），要求集装箱码头提升运力，并提高运营效率和可持续性。当前船队规模和船舶尺寸的增长，直接影响集装箱码头最关键的作业之一，即泊位分配（Steenken 等人, 2004）。从数学角度看，这一问题称为泊位分配问题（BAP），旨在为到港船舶分配靠泊位置。泊位分配问题可采用离散或连续的码头岸线设定。在离散情形下，码头岸线被划分为若干位置，每个位置每次可供一艘船舶占用。在连续泊位分配问题中，船舶可以在码头岸线的任意位置靠泊，同时与其他船舶保持安全距离。

*通讯作者电子邮箱： bmair@dtu.dk(Bernardo Martin-Iradi), darpa@dtu.dk (Dario Pacino), ropke@dtu.dk(Stefan Ropke)

此外，泊位分配问题可以是动态的，也可以是静态的。静态泊位分配问题假设规划时所有船舶都已在港口，而动态版本则允许船舶在规划期内于不同时间到港。需要指出的是，动态泊位分配问题仍然是一个确定性问题。术语动态指的是各船舶到达时间不同，而非问题本身的性质（Cordeau 等人，2005），例如车辆路径问题中的情形。图1

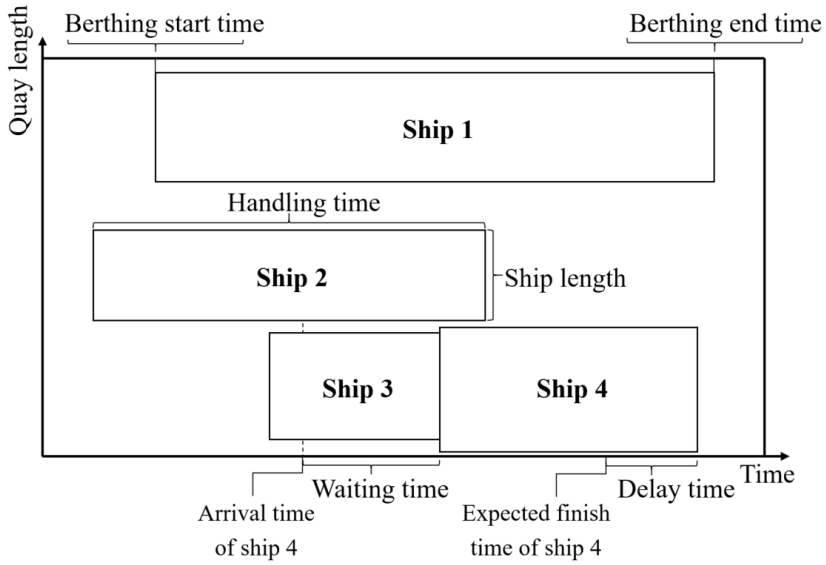


图1：连续型动态泊位分配问题的示例解

（续）港口码头，包含四艘船舶。展示了连续型动态泊位分配问题的一个示例解。

码头会优化其泊位分配，以最小化运营成本以及船舶在港口所需停留的时间，包括等待时间、装卸时间和各种延误。由于集装箱码头之间竞争激烈，它们通常不会共享超出严格必要范围的信息，并且独立于其他码头进行规划。一个潜在问题是，如果港口发生拥堵，受影响船舶的延误很容易传播到其航线上的后续港口。减少延误的一种方法是让船舶在港口之间航行时提高航速。然而，更快的航行会导致更高的燃油消耗。航运公司（即承运人）可以通过船舶航速优化问题（VSP）处理此类决策。VSP旨在优化船舶航线上相邻港口之间的航行速度（即航段）。大多数VSP研究旨在最小化船舶的燃油消耗、在港口的周转时间，以及确保给定航线班次频率所需的船舶数量。然而，VSP存在局限性。其中之一是对船舶在港口靠泊时间进行建模的方式过于简单。尽管一些研究对泊位分配进行了简化建模，但大多数研究并未将其纳入其中。未将泊位分配问题整合到VSP中，可能导致不现实甚至不可行的泊位分配，从而造成船舶延误，并使延误进一步传播。

Venturini 等（2017）首次提出了将泊位分配与船舶航速优化相结合的问题，并将其称为多港口泊位分配问题（MBAP）。该问题选取一组船舶及其航线所经过的一组港口，同时优化所有港口的泊位分配以及港口之间的航行时间。Ven-

turini 等（2017）研究了具有离散泊位位置集合的该问题版本。该问题需要联合优化承运人和码头的作业，并以相关船舶与港口事先达成一致为前提。Martin-Iradi等人（2022b）表明，这种协作能够为相关参与方（即航运承运人和码头运营商）节省成本，同时也有益于环境，因为它可以显著减少燃料排放。

如本节前文所述，连续泊位分配问题与离散泊位分配问题的主要区别在于靠泊位置的灵活性。离散泊位分配问题中的泊位位置集合是连续泊位分配问题中相应集合的子集。因此，可以认为，将码头岸线建模为连续岸线能够带来资源高效方案，因为连续泊位分配问题的最优解不劣于离散泊位分配问题的最优解。然而，这种解的质量提升潜力是以更高的复杂度为代价的，因为解空间会显著增大。

Martin-Iradi 等人（2022a）研究了具有连续岸线的多港口泊位分配问题（MCBAP），并指出其复杂度有所增加，因为他们提出的方法无法扩展到大型实例。本研究采用能够处理大型真实实例的启发式方法，以应对这一可扩展性问题。

本文作出以下四项贡献：

1. 我们为多港口连续泊位分配问题提出了一种新的混合整数规划（MIP）模型。2. 我们基于真实港口数据为多港口连续泊位分配问题构建实例生成器，并定义了一组公开可用的基准实例。3. 我们实现了一种针对多港口连续泊位分配问题设计的自适应大邻域搜索方法，并通过基于弹射链的局部搜索程序对其加以改进。4. 我们展示了自适应大邻域搜索方法在真实规模实例上的可行性；与基准商业求解器相比，该方法能够更快地找到高质量解。

本文其余部分安排如下。第 2 节全面回顾 MBAP 相关文献，以及其他涉及泊位分配或船舶调度的协作问题。第 3 节详细介绍 MCBAP，并给出其 MIP 公式。第 4 节介绍求解方法。第 5 节介绍实例生成器的详细信息以及计算研究。第 6 节总结全文并展望未来研究。

2. 文献综述

集装箱码头最重要的问题之一是泊位分配问题（BAP），二十多年来这一问题得到了广泛研究。Carlo 等人（2014）以及 Bierwirth 和 Meisel（2015）的综述汇总了其中大部分研究。Lim（1998）提出了最早的问题建模之一，并证明该问题属于 NP困难。由于连续岸线泊位分配问题变体具有更高的复杂性，文献中主要采用启发式方法。关于连续泊位分配问题的早期研究由 Kim 和 Moon（2003）以及 Imai 等人（2005）开展，他们提出了该问题的混合整数规划公式，并使用模拟退火等启发式和元启发式算法求解。Cordeau 等人（2005）研究了离散和连续泊位分配问题，并采用禁忌搜索求解。Guan 和 Cheung（2005）提出了复合启发式以及树搜索精确方法，并表明两者的表现均优于商业求解器。Kordi´c 等人（2016）研究了连续与离散泊位分配问题的混合变体，其中船舶只能在部分位置靠泊

。当泊位分配问题与其他码头作业相结合时，启发式方法得到了广泛应用。研究的主要问题之一是泊位分配与岸桥指派问题（Iris 和 Lam，2018）。Iris 等人（2017）提出了经过进一步改进的混合整数问题建模，并采用自适应大邻域搜索启发式求解；而 Cheimanoff 等人（2022）则采用变邻域搜索启发式算法。

船舶航速优化问题（VSP）也受到了文献界的广泛关注。Dulebenets 等人（2019）对该问题进行了全面综述，并指出协作与信息共享具有潜力，可成为未来的研究方向之一。据我们所知，Fagerholt（2001）首次提出了船舶航速优化问题的数学模型。Dulebenets（2018）指出，通过与码头运营商协商港口挂靠安排，承运人和码头运营商能够实现显著节约。Dulebenets（2019）提出了协作式船舶航速优化问题，其中码头运营商提供不同的港口停靠时长和装卸率，从而实现双赢。Fagerholt 等人（2010）旨在通过优化航运路线上的速度来最大限度地降低燃油消耗，并将其建模为最短路径问题。作者将各港口可能的到达时间离散化，以近似燃油消耗与航行速度之间的非线性关系。Du 等人（2011）和 Sun 等人（2018）通过考虑距离港口一定航行距离范围内的船舶，将船舶航速优化与泊位分配结合起来。

过去十年间，随着数据获取渠道的增加，研究需要不同利益相关方（如承运人和码头运营商）协作的问题变得愈发重要。Wang 等人（2015）提出了两种协作机制，鼓励承运人与码头运营商共享准确信息。Lalla-Ruiz 等人（2016）研究了离散泊位分配问题，并提出一种基于分组策略的协作搜索方法，组内成员只能在组内共享信息。Dulebenets 等人（2018）提出了协作式泊位分配问题（CBAP），其中，规划泊位分配的码头可以将超额需求转移至其他码头。Hellsten 等人（2020）提出了一种用于港口调度问题（PSP）的自适应大邻域搜索启发式，旨在为多码头港口安排支线船舶的船期。协作也已应用于扰动管理。Lyu 等人（2022）提出了泊位分配和岸桥分配重新规划的模型，并提出一种启发式方法进行求解。Guo 等人（2022）研究了泊位指派与分配问题，该问题将泊位分配问题与泊位指派与航线聚类问题相结合。Venturini 等人（2017）首次提出了多港口泊位分配问题（MBAP）的初始模型。该模型解决了多个港口中的动态离散泊位分配问题，同时优化船舶在港口之间的航行速度。Martin-Iradi 等人（2022b）针对同一问题提出了分支定价法，并运用合作博弈论研究了协作机制。Martin-Iradi 等人（2022a）将分支定价法扩展至具有连续岸线的多港口泊位分配问题，即本研究所关注的问题，并表明精确方法在小规模和中等规模实例上具有竞争力，但难以扩展至更大规模的实例。近期，Yu 等人（2022）提出了一种遗传算法，用于求解将泊位分配问题、航速优化和船舶服务差异化相结合的问题，以实现纵向和横向协作。

3. 问题描述

MCBAP整合了涉及码头运营商和航运公司的运营方面。我们考虑一组船舶和一组码头，每个码头都位于不同的港口，以优化其运营。每艘船舶都会沿其航线访问全部或部分港口。船舶访问港口的顺序可能各不相同。该问题旨在确定船舶在所访问的各个码头的靠泊位置和时间。每个码头的泊位空间有限，

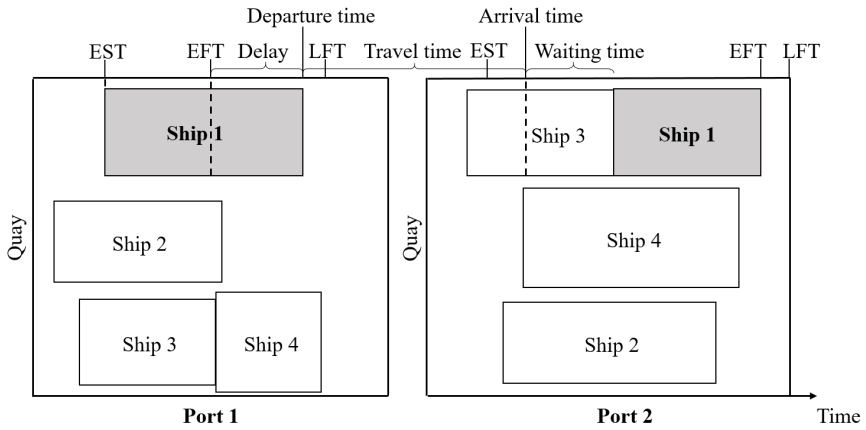


图2：四艘船舶访问两个码头时，MCBAP 解的示例表示。顶部定义了船舶1的作业时间线，其中 *EST*、*EFT* 和 *LFT* 分别表示该船舶在港口的最早开始时间、预计完成时间和最晚完成时间。

泊位空间由码头岸线的长度决定。船舶装卸货物所需的服务时间称为装卸时间，具体时长取决于靠泊位置。我们假设，装卸时间会随着与理想位置的偏差而线性增加。与大多数泊位分配问题研究类似，船舶的靠泊时间和位置受到一系列约束。船舶有一个接受服务的时间窗，也称为挂港；该计划提前制定，有助于算子分配泊位能力并避免过度拥堵。为应对延误，时间窗的结束时间并非严格限定，但延误会受到惩罚，因为延误需要动用意料之外的资源，例如增加工时。

众所周知，航行速度与燃油消耗之间的关系是非线性的。事实上，这种关系通常用三次函数近似，如公式（1）所示（Venturini 等，2017；Martin-Iradi 等，2022b）

$$F(s) = (\frac{s}{s_d})^3 F_d \tag{1}$$

其中，s 表示航行速度，s_d 表示船舶的设计航速，F_d 表示设计航速下的燃油消耗。对于我们的模型，我们将可选航行速度集合离散化，并假设船舶以其中一种速度航行于港口之间。给定可行航行速度集合后，我们可以计算相应的燃油消耗率集合。这一假设确保了问题可以采用线性形式。

图2展示了该问题的图形表示示例，突出了船舶（即船舶1）的主要运营特征。该船在最早开始时间之后才靠泊，但由于其靠泊位置对应的装卸时间较长，服务时间会在预计完成时间之后结束。超过预计完成时间的服务时间部分计为延误。船舶完成服务后，便可驶往下一个港口。船舶到达时，码头岸线已被占用，因此需要等待，直到3号船结束靠泊时段。此时的时间窗足够长，可以容纳等待时间，因此该船能够按时完成作业。

3.1. 混合整数规划公式

我们提出了一种针对多港口连续泊位分配问题的新混合整数规划公式。该公式基于 Kim 和 Moon（2003）提出的连续泊位分配问题公式，以及 Venturini 等（2017）提出的离散多泊位分配问题公式：

集合和参数：N 所有在任一港口靠泊的船舶集合。$N^* \subseteq N$ 我们要进行优化的船舶集合。$\bar{N} \subseteq N$ 被视为固定的外部船舶集合。P 港口集合。S 航速集合。L_p 港口 $p \in P$ 的码头岸线长度。$P_i \subseteq P$ 船舶 $i \in N$ 计划访问的港口集合，* 按访问顺序排序。$C_i = \{1, ..., c_i\}$ 船舶 $i \in N$ 的港口挂靠集合，每次港口访问对应一次挂靠。c_i 是最后一次港口访问，其值等于港口挂靠次数。ρ_i^c 船舶 $i \in N$ 在港口访问 $c \in C_i$ 时对应的港口 $p \in P$。$N_p \subseteq N$ 访问港口 $p \in P$ 的船舶集合。$C^p i \subseteq C_i$ 船舶 $i \in N$ 访问港口 $p \in P$ 时的港口挂靠位置。$x^i_{,0}{}^c$ 船舶 $i \in N^*$ 在港口访问 $c \in C_i$ 时的理想靠泊位置，以船舶最左侧位置为基准测量。ES T_i^c船舶 $i \in N^*$ 在港口访问 $c \in C_i$时的最早靠泊开始时间。EFT_i^c船舶 $i \in N^*$ 在港口访问 $c \in C_i$时的预计靠泊完成时间。LFT_i^c船舶 $i \in N^*$ 在港口访问 $c \in C_i$时的最晚完成时间。β 每单位距离偏离理想靠泊位置所导致的装卸时间相对增幅。$\Delta^{p,p'}$ 港口 p 与 $p' \in P$ 之间的距离。Θ_s 航速为 $s \in S$ 时每单位距离的航行时间。Γ_s^i 船舶 $i \in N^*$以航速 $s \in S$ 行驶时每单位距离的燃油消耗。l_i 船舶 $i \in N$ 的长度。F 燃料成本，单位为美元/吨。H 装卸时间成本，单位为美元/小时。D 延误时间成本，单位为美元/小时。I 等待时间成本，单位为美元/小时。U 超过最晚完成时间的成本惩罚，单位为美元。

决策变量：

$x_i^c \in$ 船舶 $i \in N$ 在港口访问 $c \in C_i$时，在码头岸线上的最左侧位置。$y_i^c \in \mathbb{R}^+$ 船舶 $i \in N$ 在港口访问 $c \in C_i$ 时的靠泊开始时间。$v_{i,s}^c \in \mathbb{B}$ 1 若船舶 $i \in N^*$ 选择速度 $s \in S$，在港口访问 c 和 $c+1$ 之间航行；$c \in C_i \backslash \{c_i\}$。$d_i^c \in \mathbb{R}^+$ 船舶 $i \in N^*$ 在港口访问 $c \in C_i$时超过 EFT_i^c的延误。u^c $_i \in \mathbb{R}^+$ 延误$y^{\text{超过 最晚完工时间}^c}$ $_i$ 对于船舶 $p\, i \in N^*$ 在 $p^{\text{港口访问 } c \in \text{复数}}$ $_i$.

辅助变量：

$\sigma^{c,c'}$ i,j $\in \mathbb{B}$　　1 若船舶 i 在港口访问 c $\in$ 复数^{p} i 和港口访问 c′ $\in$ 复数^{p} j 时，位于港口 p $\in$ P 的码头空间中船舶 j 的左侧，则取 1；否则取 0；i, j $\in \mathrm{N}_{p}$，i $\neq$ j。

$\delta^{c,c'}$ i,j $\in \mathbb{B}$　　1 若船舶 i 在港口访问 c $\in$ 复数^{p} i 和港口访问 c′ $\in$ 复数^{p} j 时，于船舶 j 开始靠泊之前完成靠泊，则取 1；否则取 0；i, j $\in \mathrm{N}_{p}$，i $\neq$ j。$\mathrm{r}^{i,c} \in \mathbb{R}^{+}$ 船舶 i 在港口访问 c $\in$ 复数$_{i}$时理想靠泊位置与实际靠泊位置之间的距离 $\in \mathrm{N}^{*}$。

从属变量：

$a_i^c \in \mathbb{R}^+$ 船舶 $i \in N^*$ 在港口访问$c \in C_i$的到达时间。$h_i^c \in \mathbb{R}^+$ 船舶 $i \in N^*$ 在港口访问$c \in C_i$的装卸时间。

$$\min \sum_{i \in N^*} \Big(\sum_{c \in C_i} I(y_i^c - a_i^c) + H(h_i^c) + D(d_i^c) + U(u_i^c) + \sum_{c \in C_i \setminus \{c_i\}} F(v_i^c \Gamma_s^i \Delta^{\rho_i^c, \rho_i^{c+1}}) \Big) \tag{2}$$

$$x_i^c + l_i \le L^p, \quad \forall i \in N_p, c \in C_i^p, p \in P \tag{3}$$

$$x_i^c + l_i \le x_j^{c'} + L^p \left(1 - \sigma_{i,j}^{c,c'}\right), \quad \forall p \in P, i, j \in N_p, i \ne j, c \in C_i^p, c' \in C_j^p \tag{4}$$

$$y_i^c + h_i^c \le y_j^{c'} + M \left(1 - \delta_{i,j}^{c,c'}\right), \quad \forall p \in P, i, j \in N_p, i \ne j, c \in C_i^p, c' \in C_j^p \tag{5}$$

$$\sigma_{i,j}^{c,c'} + \sigma_{i,j}^{c',c} + \delta_{i,j}^{c,c'} + \delta_{i,j}^{c',c} \ge 1, \quad \forall i, j \in N_p, i < j, c \in C_i^p, c' \in C_j^p, c < c', p \in P \tag{6}$$

$$y_i^c + h_i^c + \sum_{s \in S} v_{i,s}^c \Theta_s \Delta^{\rho_i^c, \rho_i^{c+1}} = a_i^{c+1}, \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{7}$$

$$a_i^c \le y_i^c, \quad \forall i \in N^*, c \in C_i \tag{8}$$

$$ES T_i^c \le y_i^c, \quad \forall i \in N^*, c \in C_i \tag{9}$$

$$y_i^c + h_i^c - EFT_i^c \le d_i^c \quad \forall i \in N^*, c \in C_i \tag{10}$$

$$y_i^c + h_i^c - LFT_i^c \le u_i^c \quad \forall i \in N^*, c \in C_i \tag{11}$$

$$\left(1 + \beta r^{i,c}\right) h_0^{i,c} = h_i^c, \quad \forall i \in N^*, c \in C_i \tag{12}$$

$$x_i^c - x_0^{i,c} \le r^{i,c}, \quad \forall i \in N^*, c \in C_i \tag{13}$$

$$x_0^{i,c} - x_i^c \le r^{i,c}, \quad \forall i \in N^*, c \in C_i \tag{14}$$

$$\sum_{s \in S} v_{i,s}^c = 1, \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{15}$$

$$y_i^c, x_i^c \ge 0 \quad \forall i \in N, c \in C_i \tag{16}$$

$$a_i^c, h_i^c, d_i^c, u_i^c, r^{i,c} \ge 0 \quad \forall i \in N^*, c \in C_i \tag{17}$$

$$v_{i,s}^c \in \{0, 1\} \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{18}$$

$$\sigma_{i,j}^{c,c'}, \delta_{i,j}^{c,c'} \in \{0, 1\} \quad \forall i, j \in N_p, i \ne j, c \in C_i^p, c' \in C_j^p, p \in P \tag{19}$$

$^{-}$外部船舶集合N 视为固定。因此，相应的决策变量集合x_i^c、y_i^c、h_i^c、$r^{i,\,c}$对于船舶$i \in {}^{-}N$均为常量，并作为输入提供给该问题。

目标函数 (2) 最小化承运人和码头运营商的运营成本。该成本以等待时间成本、装卸时间成本、延误成本和燃料消耗成本的加权和来衡量。约束条件 (3) 确保每艘船舶都在可用空间内靠泊。约束条件 (4) 和 (5) 通过启用辅助变量 $\sigma^{c,c'}$ 和 $\delta^{c,c'}$，定义每对船舶在各个维度上的相对位置。M值可限定为这对船舶中最晚的

完成时间。约束条件 (6) 确保靠泊时段在时间和空间上不重叠。约束条件 (7) 根据船舶从前一港口出发时选择的航行速度，计算其到达港口的时间。约束条件 (8) 和 (9) 分别确保靠泊开始时间严格晚于到达港口的时间和时间窗开始时间。约束条件 (10) 在预计完成时间超出时计算延误，约束条件 (11) 则定义是否遵守最晚完成时间。约束条件 (12) 计算每艘船舶每次港口访问的装卸时间，而约束条件 (13) 和 (14) 计算其与首选靠泊位置的偏差。最后，约束条件 (15) 确保船舶在港口之间航行时只选择一种速度，约束条件 (16)–(19) 定义决策变量的取值范围。

4. 求解方法

为求解（2）—（19），我们提出了一种自适应大邻域搜索（ALNS）算法。Ropke 和 Pisinger（2006）提出的 ALNS 算法扩展了 Shaw（1998）提出的大邻域搜索方法。在每次迭代中，该方法会对解进行部分破坏和重构，以生成一个新解。在本文中，为破坏解的一部分，我们会移除部分船舶在部分港口的靠泊时间和位置。我们将船舶在其航线所经港口之一的计划靠泊时间与位置组合称为港口访问，本文后续将频繁使用这一术语。此外，在某些情况下，我们会将计划中的港口访问称为矩形，这是因为在时空图中可以用矩形表示靠泊位置和时间（例如，参见图2）。

4.1. 构造启发式

自适应大邻域搜索需要一个初始解才能启动。针对这一步骤，我们提出了一个构造启发式过程，旨在寻找较好的初始解。请注意，泊位分配问题可以视为一个二维装箱问题。然而，在连续泊位场景下，泊位分配问题的复杂性有所增加，因为矩形（即港口访问）的长度会根据其在码头岸线上的靠泊位置而变化。在多港口连续泊位分配问题中，我们需要求解多个连续泊位分配问题，并额外满足部分靠泊时间取决于航行时间的约束。此外，由于船舶所遵循的航线各不相同，贪心方法更难应用，从而使问题更加复杂。我们的构造方法优先减少船舶在港口的延误。为此，我们采取以下措施：(i) 尝试将港口访问安排在较早的时间，并尽量靠近其理想空间位置，从而减少装卸时间；(ii) 减少“无用”空间，换言之，高效安排港口访问，避免在决策空间中产生无法由剩余港口访问填充的空位。请注意，由于允许解超过最晚完成时间，且时间范围不受限制，任何可能的解在数学上都是可行的。然而，我们旨在构造所有船舶均不超过最晚完工时间 的解，因为港口运营方可能会将超过该时间的解视为不可行，并且此类解还会受到较重的惩罚。该方法充当一种贪心启发式，一次安排一个港口访问。待安排的港口访问选择为约束最强的 港口访问。为找到它，我们计算每艘船舶及其每次港口访问的可行靠泊位置和时间。我们通过将码头岸线划分为给定长度（例如 10 米）的区段，并将规划时域划分为 1 小时的时间间隔，来考虑有限的位置和时间集合。对于每个时间点和区段，我们计算船舶是否可以从该时间开始靠泊，并使其最左侧从该区段开始。为了衡量船舶港口访问的受限程度，我们不计入超过最晚完成时间的靠泊时间。在所有尚未排程的港口访问中，我们将可能位置最少的港口访问定义为约束最强的 港口访问。随后我们

将该港口访问安排在某个可行位置上。实际上，我们并不考虑全部位置，而只考虑可行位置的子集，即港口访问矩形与另一个已排程港口访问或决策空间的边界（即码头岸线或规划时域的边界）直接相邻的位置。从这个位置子集中，我们选择使目标函数变化最小的位置。除了安排港口访问时直接计算的装卸成本和延误成本外，我们还需要计算燃油消耗和等待时间成本。只有在该船舶的前一次港口访问已经排程时，我们才考虑这些成本。

安排好一次港口访问后，我们会重复计算并选择约束最多的未安排港口访问，并将其安排在成本最低的有效位置。该过程如算法1所述。

算法1： 构造启发式

数据： *inst*：问题实例
结果： *sol*：包含所有已安排港口访问的解

```
开始
    unsch ← inst                    // initialize entire set of port visits to schedule
    解 ← ∅
    当未调度 ≠ ∅ 时
        // sort unplanned port visits by increasing number of feasible positions
        未调度 ← 排序(未调度)
        安排 ← 弹出首个(未调度)              // get first port visit from the list
        已计划 ← false
        // position at the earliest start time and closest to the ideal position
        位置 ← 最佳StartingPosition（toSchedule）
        当未计划 执行
            如果 feasible(pos, sol) 则
                已规划 ← true
                解 ← 计划(toSchedule，位置)          // schedule the port visit
            else
                位置 ← updatePosition(位置，解)  // update to next feasible position
                 with lowest cost
```

4.2. 移除和插入算子

移除算子的目标是选择一组已排程港口访问，并将其从当前解中移除。本文介绍的所有算子都会选择K个分配进行移除，其数量按待排程港口访问总数的百分比计算ρ。需要注意的是，移除彼此完全无关的港口访问不会带来任何潜在收益。因此，移除算子应致力于移除彼此相关的分配。

应用移除算子后，部分解中有K个缺失的港口访问需要安排。必须高效地为其分配，同时遵守其他分配，并确保解仍然可行。这就是插入算子的目标。

*4.2.1. Shaw*移除

该算子由 Shaw（1998）首次提出，用于选出相关性最高的分配对。为选出这些分配对，我们在公式 20 中定义分配i 与j 之间的相关性度量$M_{i,\ j}$，与 Iris 等人（2017）提出的度量类似。

$$M_{i,j} = A|x_i - x_j| + B|y_i - y_j| + C|(y_i + h_i) - (y_j + h_j)|, \tag{20}$$

其中，x_i、y_i 和$y_i + h_i$ 分别为分配i的靠泊位置、靠泊开始时间和靠泊结束时间。A、B和C 是用于定义各方面重要性的自定义参数。注意，$M_{i,j}$ 的值越低，相关性就越高。为了总计选择K 个分配，我们采用贪心随机准则进行选择。为使分配的选择具有随机性，我们定义参数 α。我们按$M_{i,j}$ 从小到大的顺序对所有港口访问对进行排序，并将其存入列表Ω。然后，按照公式（21）选择列表中的第i个元素：

$$i = \lceil |\Omega| \cdot p^{\alpha} \rceil, \tag{21}$$

其中，p 是一个随机数 $[0, 1)$。请注意，若 $\alpha = 1$，则选择完全随机；但随着 α 的增大，所得结果的确定性也会增强。选中的元素包含两个待移除的港口访问。选择过程持续进行，直到移除K个港口访问。请注意，此方法与Shaw（1998）的原始方法不同，因为后续的访问对不一定要与最初选中的访问对相关。

4.2.2. 时间与空间相关性移除

此移除操作采用不同的相关性度量。我们首先按成本对所有港口访问排序。船舶B_i^c 在港口访问$c \in C_i$ 的成本$i \in N^*$ 定义见公式 (22)。该成本根据船舶在该港口访问期间的等待时间、装卸时间和延误时间，以及从前一个港口（如有）航行至该港口和从该港口航行至下一个港口（如有）的燃油成本的一半来衡量。

$$B_i^c = Hh_i^c + Dd_i^c + I(y_i^c - a_i^c) + \frac{F_i^c}{2} \tag{22}$$

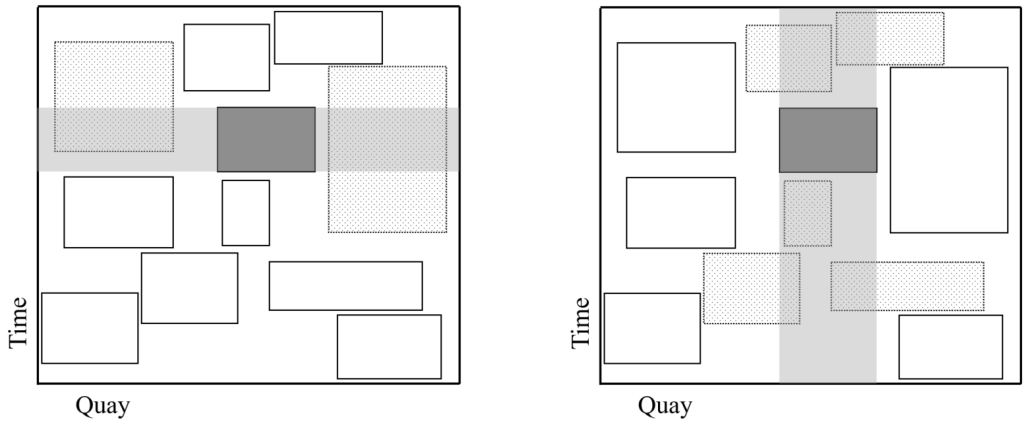
F_i^c 表示与前后港口访问（如有）相关的燃油成本。例如，如果船舶从前一个港口访问c_p 驶往港口访问c，然后继续前往下一个港口访问c_n，则燃油成本按公式 (23) 计算。

$$F_i^c = F(v_i^{c_p,c}\Gamma_s^i\Delta^{\rho(c_p),\rho(c)}) + F(v_i^{c,c_n}\Gamma_s^i\Delta^{\rho(c),\rho(c_n)}) \tag{23}$$

如果港口访问c是该船舶航线上的首次或末次港口访问，则在计算燃油成本时移除相应缺失的航行航段。

随后，我们应用公式 (21) 选取i^{th} 个成本最高的分配，并移除所有邻近分配。我们将与某项分配的距离在一定范围内的所有分配定义为邻近项。我们同时考虑时间和空间上的距离。如果在港口的时空图中将某项分配表示为一个矩形，则邻域表示与该矩形在时间或空间上重叠的区域。所有与邻域部分或完全重叠的其他分配均视为邻近分配，并予以移除。随后，我们选取成本最高的分配，并移除所有邻近分配。重复这一过程，直到移除K 项分配。

图 3 展示了时间和空间上的邻近港口访问示例。根据所考虑的维度，我们定义了两种移除算子，分别为成本-时间移除和成本-空间移除。



(a) 时间相关的港口访问　　(b) 空间相关的港口访问

图 3：深灰色标示给定港口访问在时间和空间上的邻近港口访问。

4.2.3. 随机移除

我们还考虑了一种完全随机的破坏算子。它会随机选择K 个分配并将其移除。该算子的目标并非选择相关的港口访问进行移除，而是帮助扩大搜索范围。

4.2.4. 随机化贪心插入

此方法遵循与构造启发式相同的流程，并在每一步选择要安排的港口访问时加入随机化环节。

根据可用插入位置的数量对所有未计划的港口访问进行排序。可用插入位置是指能够保持解可行的位置。例如，如果前一个或后一个港口访问已安排，则该港口访问需要确保与其通过可行的航行速度相连。我们使用随机化参数 γ 选择港口访问，方式与公式 21 中使用 α 的方式相同。这样会优先考虑可用插入位置较少的港口访问。将选中的港口访问安排在使目标函数增幅最小（即成本最低）的位置。通过重新计算其余港口访问可用插入位置的数量，反复执行这一过程。

4.2.5. κ-后悔插入

这种插入方法基于 Potvin 和 Rousseau（1993）提出的后悔-k 启发式方法。与基本的贪心启发式方法相比，该方法增加了前瞻 组件。对于每个港口访问，我们计算 κ 个最佳船期安排位置，然后将每个位置的后悔 成本度量为最佳位置与第 κ佳位置之间的差值。后悔成本最高的港口访问将成为下一个待规划的港口访问。该过程如算法2所述。

4.2.6. 装箱贪心插入

这种插入方法与第4.2.4节所述的随机化贪心插入类似。主要区别在于规划港口访问的位置。在目标值较低的位置安排港口访问，可能会产生空闲空间

算法2： κ-后悔插入

数据： *sol*、*unsch*、κ：部分破坏的解、待安排的港口访问集合，以及参数 κ

结果： *sol*：已安排所有港口访问的修复后的解。

```
开始
    当unsch ≠ ∅ 时执行
        顺序 ← ∅                                          // 初始化空列表
        对于 港口访问 ∈ unsch执行
            [pos] ← 查找最佳位置(κ)                       // 计算 κ 最佳插入位置
            后悔成本 ← c(pos[κ] − pos[1])                 // 计算后悔成本
            按 ← 排序列表(港口访问,后悔成本)               // 按后悔成本更新列表
        sol ← plan（order[1]）                            // 规划选定的港口访问
        unsch ← pop（order[1]）                           // 更新未规划港口访问集合
```

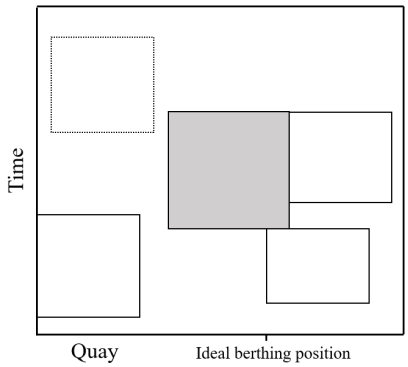


图4：示例位置的图示（实线）与灰色船舶或码头空间严格相邻。虚线表示的位置不属于位置集合，因为它既不与另一艘已规划的船舶相邻，也不与决策空间的边界相邻。

因此导致决策空间利用效率低下。该方法将可选插入位置限制为与其他已排程船舶严格相邻 的位置，或码头岸线边界及规划时域边界处的位置。严格相邻是指待安排的港口访问在至少一个时间区间内（例如一小时）与另一艘船舶紧邻靠泊，或者在与另一艘船舶至少共享一个码头区段的情况下紧靠其前方（或后方）靠泊。此外，我们还考虑以下靠泊位置：一侧位于码头岸线的一端，或靠泊时段分别从最早或最晚的可能靠泊时间开始或结束。图4展示了一些纳入考虑的位置示例。

4.2.7. 到港贪心插入

此方法与第4.2.4节中介绍的方法相同，唯一的区别在于：我们不再按可行插入位置数量递增的顺序对未规划的港口访问进行排序，而是按最早可能到达时间对其排序。此方法的主要目标之一是尽早安排港口访问，即使这可能导致成本增加，

类别	参数
当前最优解	ψ_1
优于当前解	ψ_2
不优于但被接受的解	ψ_3
被拒绝的解	ψ_4

表 1：方法奖励类别

以增加其余未规划港口访问的可能插入位置数量。

4.3. 接受准则

重构出新的解后，我们要么接受它作为新的当前解，要么拒绝它并沿用先前的解。我们采用基于模拟退火（SA）的准则来做出这一决定。这类接受准则已广泛用于自适应大邻域搜索研究（例如参见 Ropke 和 Pisinger（2006）以及 Iris 等人（2017））。如果新解 x′ 优于当前解 x（f (x′) < f (x)），或者新解较差但以概率 $e^{-(f(x')-f(x))}$ T 被接受，我们就接受新解。其中，T 是特定迭代时的当前温度，f (x) 是目标函数。我们分别定义初始温度和终止温度 T_{start} 和 T_{end}，并定义冷却时间 t_{cool}，即从 T_{start} 降至 T_{end}所需的时长。根据这些参数，我们可以通过将其从公式 $T_{end} = T_{start}\tau^{t_{cool}}$中分离出来，定义冷却因子$\tau$ （$0 < \tau < 1$）。利用该冷却因子，我们可以计算任意时刻的温度。给定第 i 次迭代时的温度 T，通过计算 $T' = T\,\tau^{t_{it}}$，即可得到第 i + 1次迭代所用的温度 T′，其中 t_{it}是第 i 次迭代的持续时间。按照 Iris 等人（2017）采用的策略，我们根据第 4.1 节所述初始解的成本 f (x_0) 计算 T_{start} 和 T_{end}，其中 ξ 和 ϕ 定义了用于计算 $T_{start} = \xi f(x_0)$ 和 $T_{end} = \phi f(x_0)$ 的成本百分比。

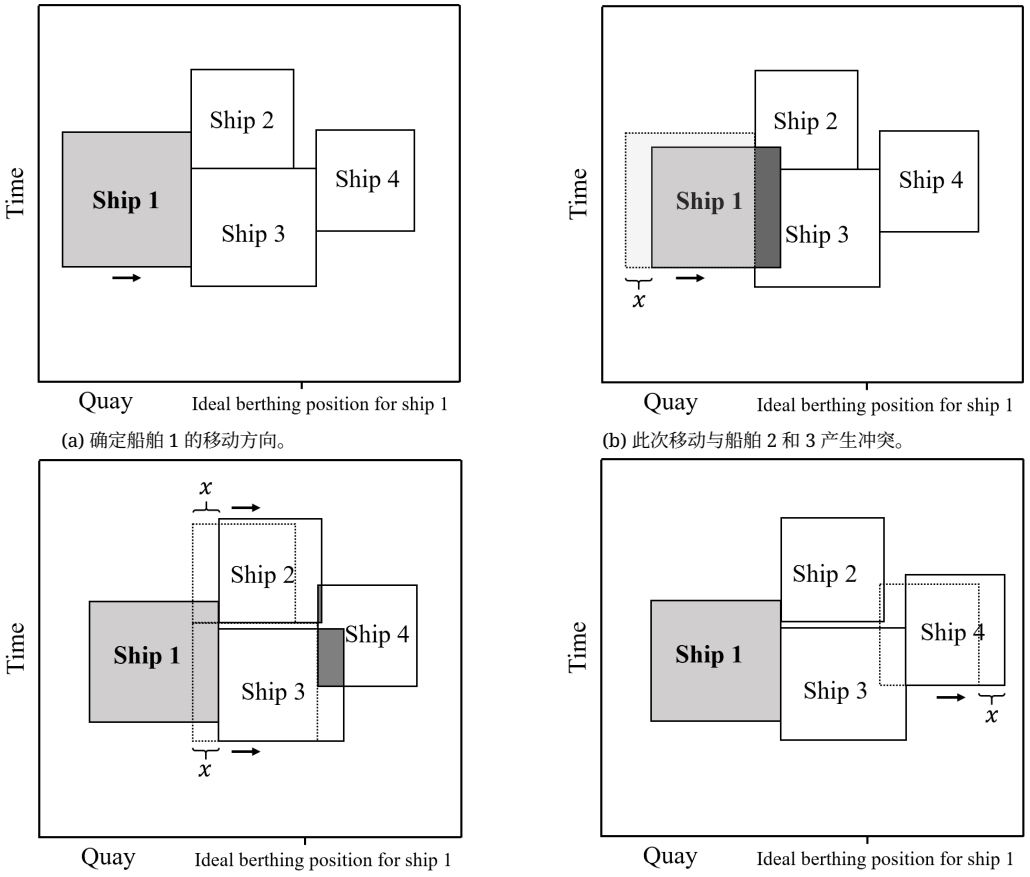
4.4. 自适应权重调整

自适应大邻域搜索方法与标准大邻域搜索（LNS）之间的主要区别之一在于前者具有自适应组件。每次迭代都会评估所采用的移除和插入算子的性能，然后根据这些评估结果更新权重，从而调整选择相应方法的概率。衡量方法性能最常见的方式是根据解的质量为其赋予不同的得分。如表 1 所示，我们定义了四种奖励类别。

设 R 和 D 分别表示插入和移除算子的集合。每次迭代中，每种移除和插入方法被选中的概率分别为 π_i^R, π_i^D。在算法运行过程中，会根据这些方法的性能更新其被选中的概率。在本研究中，我们每隔 Δ_{update} 时间更新一次概率。在这些迭代过程中，我们累计每种方法的 ψ_i^R, ψ_i^D奖励总和，并按照方程 24 更新每种方法的权重ω_i^R, ω_i^D。

$$\omega_i^R = (1-\lambda)\omega_i^R + \lambda\psi_i^R, \quad \omega_i^D = (1-\lambda)\omega_i^D + \lambda\psi_i^D \tag{24}$$

其中，λ 是一个介于 0 和 1 之间的参数，用于表示方法的适应度。如果$\lambda = 0$，权重将保持不变。这意味着，在整个算法运行过程中，每种方法的概率都相同，其行为类似于具有多个



(a) 确定船舶 1 的移动方向。

(b) 此次移动与船舶 2 和 3 产生冲突。

(c) 将船舶 2 和 3 朝同一方向移动，会与船舶 4 产生新的冲突。

(d) 将船舶 4 朝同一方向移动，通过找到新的可行解来结束驱逐链。

图5：局部搜索步骤的表示示例。移动链始于将船舶 1 朝其理想泊位位置移动一步x。灰色的港口访问表示首先移动的船舶，深灰色部分表示重叠区域。虚线矩形表示移动前船舶的原始位置。

邻域的大邻域搜索。如果 $\lambda = 1$，新算子的概率仅取决于上一次 Δ_{update} 获得的得分，而不取决于之前的得分。通常将λ 设为严格介于 0 和 1 之间的中间值。权重更新后，每种修复方法π_i^R和破坏方法 π_i^D的概率可按公式（25）计算。

$$\pi_i^R = \frac{\omega_i}{\sum_{i\in R}\omega_i}, \quad \pi_i^D = \frac{\omega_i}{\sum_{i\in D}\omega_i} \tag{25}$$

4.5. 局部搜索

我们实现了该方法的扩展版本：在重构出一个新解后，执行局部搜索过程。此步骤旨在通过测试对港口访问进行小幅调整，逐步改进解。该过程基于弹射链 策略，该策略已用于许多路径规划和基于网络的问题中（参见 Glover（1992）；Rego（1998）；Br¨aysy

（2003））。在我们的情形中，其思路是通过重新规划一次港口访问，将其调整到更理想的位置（即运营成本更低的位置），从而对解进行扰动；随后，迭代地重新规划所有与此调整冲突的港口访问。扰动链中需要重新规划的港口访问次数最多为K_{chain}；达到此上限或获得无冲突解时，扰动链即终止。图5展示了这一移动的示例。请注意，随着船舶的位置相对于其理想位置发生变化，其装卸时间（即港口访问的垂直维度）也会相应缩短或延长。算法3描述了该过程的伪代码。函数

算法3： 局部搜索过程

数据： sol，K_{chain}：当前解，以及驱逐链的长度（即港口访问移动的最大次数）

结果： sol：所得解

```
开始
    完成 ← 假                                   // 初始化终止准则
    当未完成 时
        下一个解 ← 解                           // 初始化当前最优解
        Δ ← 0                                   // 初始化成本变化量
        对 p ∈P执行
            对于n ∈ N_p do
                待移动项 = [(p,n)]              // 跟踪港口访问以重新规划
                解′ ←解                         // 初始化当前解的副本
                当 k ≤ K_chain 且待移动项 ≠ ∅ 执行
                    (p,n) ← 弹出(待移动)        // 获取要重新规划的港口访问
                    sol′ ← 移动港口访问(p,n, sol′)   // 移动港口访问
                    待移动 ← 计算冲 突(sol′)    // 检查冲突
                    k ← k + |待移动|            // 更新驱逐链长度
                if 待移动 = ∅then
                    δ ← 计算成本变化(sol,sol′)  // 计算成本变化
                    if δ < Δthen
                        Δ ← δ                   // 更新最佳成本增量
                        下一个解 ← 解′          // 更新当前最优解
        如果 Δ < 0则
            解 ← 下一个解                       // 更新要返回的解
        else
            完成 ←true                          // 没有更优的邻域解
```

movePortVisit(p,n) 对给定的港口访问执行扰动（即船舶 n 在港口 p 的港口访问）。需要注意的是，方向由第一次执行的扰动决定。为确定第一次扰动的方向，我们计算沿三个方向移动该港口访问时的成本变化：(i) 沿空间轴向理想位置移动一个线段长度；(ii) 沿时间轴提前一个时间点；以及 (iii) 沿时间轴延后一个时间点。如果该港口访问尚未安排在其理想位置，则检查空间轴方向。沿选定方向执行扰动后，处于冲突中的后续港口访问将以相同的

方向进行扰动。

K_{chain} 的值较高，会增加找到更优解的概率，也会增加计算所需的运算量。参数 K_{chain} 应兼顾解的质量和较低的计算复杂度。因此，我们将 K_{chain} 的值定义为取决于实例中的船舶数量，并令其等于 $K_{chain} = 2 \cdot |N|$。之所以采用 $K_{chain} > |N|$，是因为对于某些移动，冲突的港口访问可能需要经过多次扰动才能找到可行的新位置，而选择较低的 K_{chain} 值可能会限制过多。

4.6. 算法概述

求解方法的概述见算法4。由于局部搜索过程需要额外的计算开销，我们不会在每次迭代中都执行该过程。相反，只有当重构解优于当前解时，我们才执行局部搜索。这样可以减少局部搜索的执行次数，使算法能够进行更多次迭代，同时将局部搜索限定在已有较有希望的解时执行。

算法4： 自适应大邻域搜索过程

数据： *inst*,*param*：算法的问题实例和参数设置
结果： *bestSol*：找到的最优解

```
开始
    ψ,π ← 初始化(inst)                    // 初始化算子选择参数
    解 ←构造启发式(实例)                   // 构造初始解
    最佳解 ← 解
    当未达到时间限制时 执行
        当前解 ← 解
        移除,插入 ← 选择算子(π)            // 选择算子
        sol ← 插入(移除(currSol))          // 获取新解
        如果 c(sol) < c(currSol) 则
            sol ← 局部(搜索)
        如果 isAccepted(sol)则
            若 c(解) < c(最佳解) 则
                最佳解 ← 解
            当前解 ← 解
        π ←更新算子参数(ψ)
```

5. 计算结果

在本节中，我们首先描述基准实例集的生成过程，然后开展计算研究，涵盖方法的性能以及该问题的实践见解。

5.1. 实例生成

据我们所知，Martin-Iradi 等人（2022a）是唯一一项关于多港口连续泊位分配问题的研究。该研究所呈现的实例规模较小且数量有限。因此，我们构建了更为全面的基准实例集。在缺乏现实数据的情况下，可以将连续泊位分配问题的当前基准实例扩展至多个港口。相反，我们决定使用公开获取的港口数据（Marine Traffic，2023），并通过工业研究合作伙伴提供的额外数据对其进行验证，从而为具有连续岸线的多港口泊位分配问题创建实例生成器。

我们考虑三种不同的船舶类型：(i) 长度不超过200米的支线船或小型船舶；(ii) 长度在200至300米之间的中型船舶；以及(iii) 长度超过300米的大型船舶。每种船舶类型的速度—燃油消耗关系各不相同。此外，我们考虑北海三个主要港口的三座码头：(i) 鹿特丹 *APMT*，其码头岸线长1600米（APM Terminals，2022b）；(ii) 不来梅哈芬 NTB，码头岸线长1800米（APM Terminals，2022a）；以及汉堡 EGH，码头岸线长2100米（Eurogate，2022）。这三个港口彼此相距较近，大型、中型和小型船舶会按照不同顺序挂靠这些港口，将其作为各自航线的一部分。

船舶时间窗的时长依据计划港口停靠时长确定。船舶的计划挂港时长通常会在预计到达时间前几天更新。因此，我们为每艘船设定一个固定时间点，即实际到达时间前两周，并将预计到达时间（ETA）与预计离港时间（ETD）之间的时间差作为计划港口停靠时长。我们对三个月期间（2021年1月至3月）各港口、各船舶类型的靠泊船舶的计划港口停靠时长取平均值，以计算这一时长。该值也用于定义最短装卸时间 h_0（见表2）。小型、中型和大型船舶的港口服务时间若分别超过48、72和96小时，则被归类为异常值并从数据集中剔除。原因在于，如此长的服务时间通常涉及维护或加油作业，而这些作业通常不会定期进行。因此，它们不属于本问题的范畴。

表2：每种船舶类型和码头的最短装卸时间（小时）。这些取值定义了$h^i_{,\ 0}{}^c$。

船舶类型\码头	**DEHAM**	**DEBRV**	**NLRTM**
支线船	10.1	12.1	10.4
中型	18.0	21.8	18.4
大型	41.0	33.7	26.7

我们定义了六种不同的船舶模式，每种模式都具有指定的航线、船舶类型和长度。所有船舶均按不同顺序访问两个或三个港口。给定实例中的*N* 艘船舶从这六种模式中抽取。

对于每艘船舶，我们随机设定 (i) 其在所访问的每个港口的期望靠泊位置，以及 (ii) 最早开始时间 ES T^c_i；设定时遵循相关参数，以确保港口之间存在可行航行时间。预计完成时间 EFT^c_i通过将表2中的相应数值加到 ES T ci (EFT ci = ES $T^c_i + h^i_{,0}{}^c$)上计算得出。此外，我们将船舶在该港口可能采用的最短 $h^i_{,0}{}^c$与最长 $h^i_{,max}{}^c$装卸时间之差的一半加到EFT^c_i上，以计算最晚完成时间 LFT ci (LFT ci = EFT ci $+ h^i_{,max}{}^c - h^i_{,2\ 0}{}^c$)。

作为实例生成器的输入，我们定义每个港口中被视为固定的外部船舶数量 *Nout*。对于每艘外部船舶，我们随机确定：(i) 靠泊位置

和时间，(ii) 长度（介于 180 米和 330 米之间），以及 (iii) 装卸时间，该时间根据船舶长度按比例设定。

我们假设，除非有外部船舶占用，否则整条码头岸线均可用于靠泊。我们还考虑了 10 个不同的速度等级，速度均匀分布在 17–21.5 节之间。本文提出的自适应大邻域搜索启发式能够处理任意连续速度值，但本文提出的混合整数规划公式则不能。为确保对各种方法进行公平比较，我们在公式和求解方法中均采用离散的速度等级集合。此外，港口之间的距离根据航线的实际海上距离计算。

5.1.1. 装卸时间

尤其对于离散型泊位分配问题，通常会根据靠泊位置定义不同的装卸时间。对于本文实现的连续型问题，我们采用 Meisel and Bierwirth（2009）提出的装卸时间定义：通过偏差系数 $\beta \geq 0$（即每单位距离〔米〕对应的装卸时间相对增幅）对偏离首选靠泊位置的情况进行惩罚。给定首选靠泊位置处的最短装卸时间 $h^{i,}{}_{0}{}^{c}$ 以及所选位置相对于首选位置的实际偏差 Δb（以米为单位），装卸时间按如下方式计算：

$$h_i^c = (1 + \beta \Delta b) h_0^{i,c} \tag{26}$$

作为参考，中型船舶的装卸时间为 20 至 60 小时，大型船舶则为 30 至 110 小时。这分别对应于在最佳和最差位置靠泊的情况。

5.1.2. 时间窗

在多港口连续泊位分配问题中，时间窗分为两种类型：

- **每艘船在每个挂靠港口的时间窗。**该时间窗由港口停靠时长决定。必须遵守时间窗的起始时间，但可以超过结束时间。因此，靠泊时段可以超出时间窗的结束时间，超出的部分计为延误。

  我们还考虑用于确定最晚完成时间（LFT）的时间窗结束时间。理想情况下，必须遵守该时间窗。不过，我们允许违反该时间窗，但须加计非常高的惩罚成本。

- **泊位位置的时间窗。**该时间窗可视为指定泊位位置或码头岸线某一段的作业时段。我们假设所有泊位位置始终可用。需要注意的是，潜在的维护时间窗或码头岸线部分关闭，也可以用外部船舶占用指定位置和时段的方式进行建模。

5.1.3. 基准实例

使用本节所述的实例生成器，我们创建了一组基准实例。该集合共包含240个实例。每个实例均由表3所列参数值的组合构成：(i) 一个随机种子，(ii) 待优化的船舶数量，(iii) 每个港口的外部固定港口访问次数，以及 (iv) 用于定义可行靠泊位置的码头区段长度。

表 3：基准实例集的参数设置。

参数	Seed	编号数量	每个港口的外部船舶数量	位置间距离
取值	1-10	30, 50, 70	5, 10	10, 20, 40, 80

5.2. 参数调优

ALNS算法共有18个算法参数，包括用于校准算子、选择准则和接受准则，以及新解得分权重的参数（见表4）。为选出算法参数的最佳取值，我们进行了参数调优。我们选取了12个能够代表整个实例集的实例子集。调优时，我们运行自动算法配置器 *Pydgga* (Ans´otegui et al., 2021)，进行30代迭代，并将每次ALNS运行的时间限制设为5分钟。该配置器支持并行化处理，整个参数调优耗时8小时。表4列出了调优所用的各算法参数取值范围及找到的参数设置。模型和求解方法使用 *Julia* 编写，并在配备16 GB RAM的2.90 GHz Intel Xeon Gold 6226R处理器上以单线程运行。

5.3. 方法性能

为了衡量该方法的质量，我们将所提出的方法与其变体以及基线商业求解器进行比较；*CPLEX v12.10*。

表 5 比较了 CPLEX 与自适应大邻域搜索方法的结果。我们计算每次实例运行的目标差距为 $z^{obj} - z_{best} z_{best}$，其中 z^{obj} 是该次运行最优解的目标值，而 z_{best} 是所有实验中的已知最优解。我们按照船舶数量、外部船舶数量以及靠泊位置之间的距离对所有实例进行分组。每组包含 10 个实例，并根据其共同特征命名为*X-Y-Z*。*X* 表示船舶数量，*Y* 表示每个港口的外部船舶数量，*Z* 表示相邻位置之间的距离（以米计，即线段长度）。因此，表中的每一行对应于随机种子值不同的实例的平均值。通过对每个实例运行 10 次，并计算平均值、最佳值和最差值，对自适应大邻域搜索进行测试。

我们发现，CPLEX 的扩展性较差，尤其是在船舶数量超过 30 艘的实例中。对于规模最小的实例，间隙较小，但很快就会增大。平均而言，自适应大邻域搜索方法在大多数实例中取得了更小的间隙，优于商业求解器。间隙可用于比较两种方法的性能，但无法提供最优性保证。CPLEX 得到的下界表明，最优性间隙较大。这可能是由于解的质量较低或下界较弱。Martin-Iradi 等人（2022b）指出，针对离散码头岸线的多港口泊位分配问题混合整数规划公式的松弛可能较弱，并表明网络流重新公式化可以显著加强松弛。正如 Martin-Iradi 等人（2022a）所指出的，多港口连续泊位分配问题的网络流公式可能存在可扩展性问题，但其松弛更强。我们将 Martin-Iradi 等人（2022a）提出的分支定价法结果与自适应大邻域搜索方法进行比较。Martin-Iradi 等人（2022a）提出的公式与本文研究的问题略有不同。该公式为每艘在港口靠泊的船舶定义了一个必须满足的最晚完成时间。我们将 Martin-Iradi 等人（2022a）的方法调整为适用于本研究的公式。分支定价法基于图表示，因此我们需要确定最晚可能的靠泊时间。该时间设定为比

表4：参数调优后算法参数的研究范围和选定设置。

符号	描述	最小值	最大值	调优设置
ϵ	用于计算冷却比率的值	0.005	0.2	**0.157**
φ	用于定义起始温度的初始解目标值百分比（重新加热时）	0.01	0.05	**0.0246**
ξ	用于定义结束温度的初始解目标值百分比（重新加热至该温度）	0.00005	0.001	**0.000269**
ρ	破坏程度，占港口总数的百分比由移除方法移除的访问	0.3	0.6	**0.326**
A	Shaw移除中位置偏差的权重	0.5	2	**0.55**
B	靠泊开始时间偏差的权重在Shaw移除中	0.5	2	**1.36**
C	其中靠泊结束时间偏差的权重Shaw移除	0.5	2	**0.89**
α	Shaw移除的随机性参数方法	1	3	**2.66**
γ	随机贪心的随机性参数修复方法	1	3	**2.85**
μ	到港贪心的随机性参数修复方法	1	3	**2.6**
κ	k-后悔值参数	2	4	**2**
Δ	更新以下内容之间的迭代次数各方法的权重	0.01	0.05	**0.017**
η	用于调整以下内容重要性的参数近期得分相对于先前权重	0.3	0.7	**0.456**
ψ_1	找到当前最优解时的得分	10	20	**11**
ψ_2	找到优于当前解时的得分	4	8	**4**
ψ_3	接受该解时的得分	1	3	**2**
ψ_4	拒绝该解时的得分	-	-	**0**
β	定义位置的参数船舶边界（乘以长度）	2	5	**4.02**

表 5：CPLEX 求解混合整数规划公式以及自适应大邻域搜索方法在 5 分钟和 1 小时时间限制下的间隙。我们还报告了自适应大邻域搜索多次运行中最佳与最差结果之间的平均差距。每行对应一个实例组（即 10 个相同规模实例的平均结果）。

	5 分钟				**1 小时**			
实例组	**MIP 间隙（%）**	**ALNS 间隙（%）**	**最佳自适应大邻域搜索 间隙（%）**	**最差自适应大邻域搜索 间隙（%）**	**MIP 间隙（%）**	**ALNS 间隙（%）**	**最佳自适应大邻域搜索 间隙（%）**	**最差自适应大邻域搜索 间隙（%）**
30_5_10	**10.1**	11.9	7.3	17.4	**1.5**	4.8	1.8	8.4
30_5_20	**7.4**	9.4	4.6	14.9	**1.7**	3.3	0.9	6.0
30_5_40	10.2	**6.9**	3.6	10.2	**1.2**	3.4	1.2	5.5
30_5_80	15.3	**7.5**	4.3	10.5	**2.1**	3.4	0.9	5.5
30_10_10	15.1	**10.0**	6.1	14.5	**2.7**	3.6	0.8	6.8
30_10_20	16.5	**8.1**	4.7	11.9	5.8	**2.9**	0.5	5.2
30_10_40	13.3	**5.9**	3.3	8.4	3.5	**2.3**	0.2	4.3
30_10_80	16.9	**5.2**	3.3	7.6	5.7	**2.3**	0.5	4.5
50_5_10	22.5	**14.4**	8.6	19.3	**2.2**	4.3	0.7	9.6
50_5_20	45.7	**11.4**	6.1	18.2	6.4	**3.6**	0.8	6.9
50_5_40	59.1	**8.8**	4.3	13.1	11.1	**2.9**	0.0	6.2
50_5_80	82.7	**7.2**	4.3	10.6	14.3	**2.9**	0.5	5.3
50_10_10	66.5	**10.3**	5.2	14.6	8.1	**2.9**	0.6	5.9
50_10_20	61.4	**8.3**	4.4	13.0	9.2	**2.2**	0.0	4.6
50_10_40	74.6	**7.0**	3.9	11.3	13.4	**2.7**	0.0	5.5
50_10_80	92.3	**6.2**	3.5	9.3	16.7	**2.2**	0.0	4.3
70_5_10	139.5	**13.8**	7.2	18.9	8.7	**3.9**	0.8	8.2
70_5_20	103.4	**10.9**	5.6	16.3	9.3	**2.4**	0.0	5.9
70_5_40	141.3	**7.8**	4.2	13.4	14.7	**2.4**	0.0	4.8
70_5_80	136.5	**5.8**	2.9	9.0	17.9	**2.3**	0.0	4.5
70_10_10	95.2	**11.7**	7.6	16.0	15.5	**2.7**	0.0	5.5
70_10_20	90.8	**8.8**	4.2	13.0	16.8	**2.3**	0.0	5.1
70_10_40	122.0	**7.3**	3.9	11.1	29.5	**2.3**	0.0	4.7
70_10_80	141.0	**5.5**	3.3	8.4	28.2	**2.1**	0.0	3.9
平均值	65.8	**8.7**	4.8	13.0	10.3	**2.9**	0.4	5.7

表 6：基于 Martin-Iradi 等人（2022a）所用设置的实例集参数设置。

参数	Seed	数量 数量	外部船舶数量 每个港口的	位置间 距离
取值	1-5	4-15	3-5	10, 20, 40, 80

表7：对720个实例进行的性能比较，比较了由 CPLEX 求解的混合整数规划公式、适应 d Martin-Iradi 等人（2022a）提出的分支定价法，以及自适应大邻域搜索方法，时间限制为5分钟和1小时。每行显示具 1 有相同船舶数量的所有实例的平均差距。

数量 船舶	CPLEX 最优性间隙（%）		分支定价法 最优性间隙（%）		CPLEX 间隙（%）		分支定价法 间隙（%）		自适应大邻域搜索+局部搜索 间隙（%）	
	5 分钟	1 小时	5 分钟	1 小时	5 分钟	1 小时	5 分钟	1 小时	5 分钟	1 小时
4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.1	**0.0**
5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
6	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.2	0.1
7	**0.0**	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	0.2
8	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	0.2
9	**0.0**	**0.0**	0.3	0.1	**0.0**	**0.0**	0.2	**0.0**	0.3	0.2
10	**0.0**	**0.0**	0.1	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.5	0.3
11	1.0	0.2	**0.3**	**0.1**	**0.1**	**0.0**	**0.1**	**0.0**	0.3	0.2
12	1.1	0.6	**0.8**	**0.2**	**0.1**	**0.0**	0.4	**0.0**	0.2	0.1
13	4.0	2.7	**1.1**	**0.6**	**0.1**	**0.1**	0.4	0.2	0.4	0.2
14	3.5	2.0	**1.2**	**0.4**	**0.1**	**0.1**	0.6	**0.1**	0.7	0.4
15	7.3	4.8	**1.7**	**1.0**	**0.3**	**0.1**	0.7	0.2	0.5	0.3
平均值	1.41	0.86	**0.49**	**0.20**	**0.06**	**0.02**	0.20	0.05	0.30	0.19

最晚完成时间。这使该方法能够超出最晚完成时间，同时将图保持在合理规模。这个界限较为宽松，我们的实证研究表明，该界限不会影响最优解。为此，我们还使用第 5.1 节中定义的实例生成器，生成了一组规模与 Martin-Iradi 等人（2022a）所介绍实例相近的新实例。该实例集的输入参数见表 6。整个实例集包含 720个实例，每种输入参数组合对应一个实例。结果见表7，其中我们根据船舶数量将实例分为每批 60 个。我们将分支定价法与本研究提出的自适应大邻域搜索方法和混合整数规划模型进行比较。对于分支定价法和 CPLEX，我们计算了各自的最优性间隙；结果发现，在大多数情况下，由于下界更紧，分支定价法取得了更小的间隙。我们还计算了这三种方法相对于已知最优解的间隙。在这种情况下，CPLEX 的性能最佳，这表明尽管其松弛效果较差，但找到的上界接近最优。分支定价法仍表现稳健，但当计算时间较短且实例规模较大时，自适应大邻域搜索方法能够提供更好的解。

自适应大邻域搜索方法有两个主要组成部分，使其区别于其他启发式算法：(i) 引导算子选择的自适应过程，以及 (ii) 在找到有前景的解时执行的局部搜索过程。为量化这两个过程的影响，我们将所提出的方法与分别包含或不包含这两个组成部分的变体进行比较。其中一个变体是不含自适应组成部分的方法（即大邻域搜索（LNS）），这意味着在整个算法运行过程中，每个移除算子和插入算子被选中的概率都相同。另一个变体是不含局部搜索（LS）的自适应大邻域搜索方法。

表8：所提出的自适应大邻域搜索方法各变体的性能比较。

	5分钟				1小时			
实例组	自适应大邻域搜索 + 局部搜索 间隙（%）	自适应大邻域搜索（无局部搜索） 间隙（%）	大邻域搜索 + 局部搜索 间隙（%）	大邻域搜索（无局部搜索） 间隙（%）	自适应大邻域搜索 +局部搜索 间隙（%）	自适应大邻域搜索（无局部搜索） 间隙（%）	大邻域搜索 +局部搜索 间隙（%）	大邻域搜索（无局部搜索） 间隙（%）
30_5_10	11.9	**11.8**	12.0	13.6	**4.8**	4.9	5.5	5.9
30_5_20	9.4	9.6	**8.8**	10.6	**3.3**	3.6	3.4	4.0
30_5_40	**6.9**	7.3	7.3	7.8	3.4	3.3	**2.1**	2.5
30_5_80	7.5	7.5	**6.4**	7.0	3.4	3.7	**2.1**	2.3
30_10_10	**10.0**	10.9	10.6	11.8	**3.6**	3.7	4.2	4.7
30_10_20	**8.1**	8.2	8.6	9.6	2.9	**2.8**	3.1	3.3
30_10_40	5.9	6.3	**5.6**	6.3	2.3	2.4	**1.4**	1.7
30_10_80	5.2	5.4	**4.5**	5.3	2.3	2.3	**0.8**	1.0
50_5_10	**14.4**	15.5	15.1	18.2	**4.3**	5.1	7.1	9.4
50_5_20	11.4	**12.2**	12.8	16.3	**3.6**	4.4	5.1	7.7
50_5_40	**8.8**	10.2	11.0	13.2	**2.9**	3.9	3.8	6.2
50_5_80	**7.2**	8.3	8.2	10.6	**2.9**	3.0	3.4	4.6
50_10_10	**10.3**	11.4	12.0	14.4	**2.9**	3.9	5.1	7.1
50_10_20	**8.3**	9.2	9.9	12.5	**2.2**	3.3	3.6	5.7
50_10_40	**7.0**	8.1	8.1	10.9	**2.7**	3.5	3.1	4.9
50_10_80	**6.2**	6.6	6.6	8.7	**2.2**	2.6	2.6	3.1
70_5_10	**13.8**	14.7	14.6	17.4	**3.9**	5.4	6.7	10.9
70_5_20	**10.9**	11.4	11.4	14.4	**2.4**	4.6	5.1	8.5
70_5_40	**7.8**	10.1	9.4	13.1	**2.4**	3.9	3.9	7.1
70_5_80	**5.8**	7.8	7.6	11.0	**2.3**	3.5	3.2	5.5
70_10_10	**11.7**	12.6	11.8	14.8	**2.7**	4.3	5.2	8.3
70_10_20	**8.8**	10.3	9.6	13.3	**2.3**	4.2	4.6	7.6
70_10_40	**7.3**	9.2	8.4	11.9	**2.3**	3.8	3.9	6.2
70_10_80	**5.5**	6.8	6.6	9.4	**2.1**	3.0	3.0	4.5
平均值	**8.7**	9.7	9.4	11.8	**2.9**	3.7	3.8	5.5

表8比较了各方法的目标差距，时间限制分别为5分钟和1小时。在较短的时间限制下，我们发现，在大多数实例中，不含局部搜索的自适应大邻域搜索表现最佳。随着时间限制延长，局部搜索的价值更加明显，并且在大多数实例中都能取得最佳结果。

为了衡量局部搜索过程的影响，我们比较了局部搜索过程执行频率不同的策略。我们测试了该算法的另外三种变体，分别每隔1、2和4次迭代调用一次局部搜索。结果汇总于表9，从中可以看出，局部搜索会增加每次迭代的计算复杂度。非常频繁地执行局部搜索过程，可能用较少的迭代次数得到较好的解，但也会导致更长的计算时间。所提出的策略在重构解优于当前最优解时执行局部搜索，我们证明了该策略的表现最佳。该方法使我们能够在较少的迭代次数中执行局部搜索，同时也有望得到令人满意且更优的解。

表10和表11总结了自适应大邻域搜索方法中使用的不同移除算子和插入算子的性能。对于每种移除算子，我们计算三个指标：(i) 算子被选中的迭代次数占比% *its*，(ii) 使用该算子找到当前最优解的比例% *new best*，以及(iii) 使用该算子得到的解优于当前解的次数占比% *better than current*。对于插入方法，我们还列出第四列% *of time*，表示每种算子在修复解所花费的总时间中占用的比例。移除方法所花费的时间显著少于插入算子，因此我们不计算移除算子的这一指标。我们观察到，当船舶数量为30时，随机移除是性能更好的移除方法。然而，对于更大的实例，

表9：时间限制为一小时的局部搜索过程在不同频率下的算法比较。

	局部搜索 每次迭代		局部搜索 每2次迭代		局部搜索 每 4 次迭代		每次迭代都进行局部搜索，条件是 解优于当前最优解		
实例组	**Gap (%)**	**迭代次数 x1000**	**Gap (%)**	**迭代次数 x1000**	**Gap (%)**	**迭代次数 x1000**	**Gap (%)**	**迭代次数 ×1000**	**迭代次数百分比 使用局部搜索**
30_5_10	5.3	5.2	5.3	7.3	5.3	7.8	4.8	12.7	1.3
30_5_20	4.2	8.0	4.1	11.1	4.2	12.8	3.3	18.6	1.5
30_5_40	3.8	11.0	3.4	16.5	3.6	21.1	3.4	27.4	2.5
30_5_80	4.3	15.7	3.9	25.7	3.8	36.0	3.4	60.4	2.7
30_10_10	3.9	5.6	4.0	7.1	4.0	7.8	3.6	10.9	1.6
30_10_20	3.1	8.8	3.1	12.3	3.5	13.3	2.9	20.0	1.7
30_10_40	2.5	12.4	2.3	18.2	2.2	23.9	2.3	30.4	2.9
30_10_80	2.7	17.6	2.5	28.9	2.5	39.4	2.3	63.4	3.0
50_5_10	3.9	2.0	4.4	3.2	4.9	3.8	4.3	8.5	0.4
50_5_20	4.2	2.6	4.5	4.3	4.6	6.2	3.6	16.2	0.4
50_5_40	3.4	3.4	2.9	5.3	3.4	7.8	2.9	13.8	0.8
50_5_80	4.1	4.5	3.9	7.6	3.7	11.6	2.9	24.8	1.1
50_10_10	2.9	2.3	2.9	3.5	3.7	4.6	2.9	8.8	0.7
50_10_20	2.6	3.1	2.8	5.1	2.9	6.3	2.2	15.4	0.6
50_10_40	3.1	4.2	3.1	6.4	3.1	9.1	2.7	14.3	1.2
50_10_80	2.9	5.5	2.8	9.2	3.0	13.4	2.2	26.8	1.6
70_5_10	3.4	1.2	3.7	1.9	4.6	2.5	3.9	5.0	0.5
70_5_20	2.9	1.5	3.4	2.5	4.0	3.3	2.4	10.0	0.5
70_5_40	3.4	1.7	3.5	2.9	3.6	4.6	2.4	11.4	0.5
70_5_80	3.7	2.0	3.5	3.7	3.5	6.0	2.3	17.7	0.7
70_10_10	2.8	1.3	3.1	2.1	3.6	2.6	2.7	5.0	0.8
70_10_20	2.5	1.8	2.7	2.8	3.1	4.0	2.3	9.2	0.7
70_10_40	3.3	2.1	3.2	3.5	3.5	5.2	2.3	10.0	0.8
70_10_80	3.5	2.3	3.5	4.1	3.1	6.9	2.1	18.9	0.9
平均值	**3.4**	**5.2**	**3.4**	**8.1**	**3.6**	**10.8**	**2.9**	**19.1**	**1.2**

表10：四种移除算子的性能汇总。实例按船舶数量分组。

数量船舶的	**成本-泊位移除**			**成本-时间移除**			**Shaw移除**			**随机移除**		
	% 个	% 新 best	% 更好比当前	% 其	% 新 best	% 更好比当前	% 它的	% 新的 best	% 更好比当前	% 它的	% 新 best	% 更优优于当前方案
30	6	4	2	17	21	13	28	20	24	50	55	62
50	6	4	1	47	50	38	12	8	8	35	38	52
70	6	4	2	69	81	84	6	2	1	19	12	13

表11：四种插入算子的性能总结。实例按船舶数量分组。

数量的船舶	**高效装箱插入**				**随机贪心插入**				**到港贪心插入**				**k-后悔值插入**			
	% 个	% of time	% 个新 best	% 更优比当前	% 其	% of time	% 新 best	% 更优比当前	% 它的	% of time	% 新的 best	% 更好优于当前	% 它的	% of time	% 新的 best	更优百分比相比当前
30	5	2	4	1	17	10	24	7	10	5	13	3	68	83	60	89
50	5	3	3	1	37	33	34	22	8	6	6	2	49	58	57	75
70	6	3	3	1	47	39	51	46	9	6	6	3	38	52	39	50

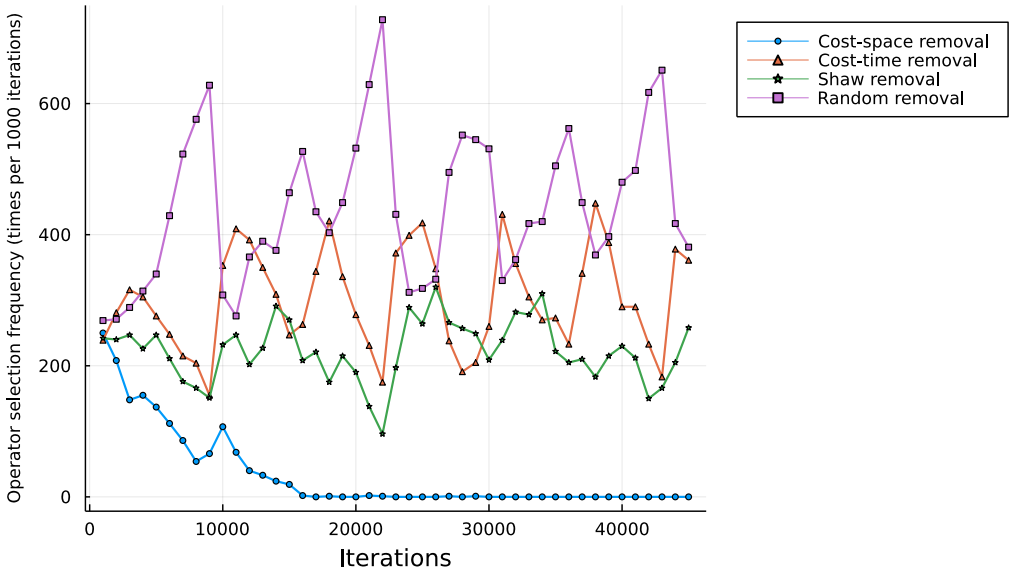


图6：30 10 10组中的一个实例运行算法1小时期间，各移除算子的使用情况。

成本-时间移除的表现更好。这表明，随着港口访问次数增加，移除彼此毫无关联的港口访问的概率也会增大，从而降低纯随机方法的效率。此外，移除时间上重叠的港口访问，比移除泊位空间上重叠的港口访问更有效。船舶可以在码头岸线的多个位置靠泊，而不会造成严重延误或干扰船期；但提前或延后靠泊可能会对其余航段的航行产生负面影响。因此，这种灵活性需要付出更高的成本。我们还注意到，Shaw移除在较大规模的实例中作用会减弱。该算子每次移除一对港口访问，而这对访问之间可能毫无关联，从而加剧整个算子的负面效果。与随机移除类似，港口访问次数越多，这种相互关联性问题就越突出。

就修复方法而言，κ-后悔插入法表现最佳且更为稳健。尽管在某些情况下，它并非使用频率最高的算子，但其计算开销显然很大，超过一半的时间都花在计算与该方法相关的插入上。随机化贪心插入也很重要，并且在规模较大的实例中效果更佳。其余两个插入算子——装箱和到港贪心——表现平平。不过，在算法运行过程中，它们也有助于找到一些最佳解。

为了更好地了解算法的行为以及各算子的使用时机，我们跟踪了算法运行期间每个算子的使用情况。图6和图7展示了一个实例的一小时运行示例，该实例包含30艘船舶、每个港口10艘外部船舶，且码头区段长度为10米。我们观察到，成本-泊位移除只在运行初期使用，此时各算子被选中的概率较为均衡。装箱贪心启发式表现出类似的行为，这与表10和表11所示这两种算子的低使用率相符。尽管如此，其余算子在大多数算法运行中都会使用。一些算子表现出振荡行为，例如成本-时间移除算子。这种行为与

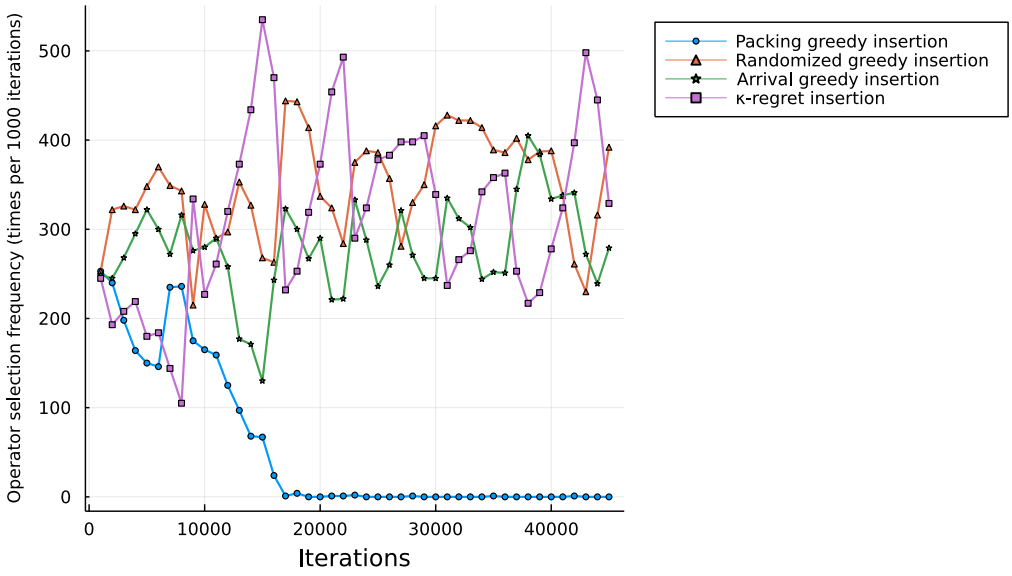


图 7：30 10 10组的一个实例在算法运行 1 小时期间各插入算子的使用情况。

表12：不同码头区段长度下各实例的平均运营成本和成本变化。所有实例均运行一小时。成本单位为千美元。

线段长度	等待	装卸	延误	Fuel	总计	惩罚	迭代次数 ×1000
10	390	537	3210	1274	5411	0.04	8
20	368	537	3246	1275	5426	0.05	15
40	242	541	3449	1276	5509	0.09	18
80	241	542	3910	1275	5968	0.14	35

接受准则的温度会定期升高，这意味着温度越高，该算子被选中的概率也越大。

此外，对于移除和插入算子，我们测试了算法逐个移除表现最差算子的效果，但在所有情况下，方法的性能都变差了。这表明所有算子在一定程度上都有用，并且彼此配合良好。

5.4. 实际影响

该问题涉及两类主要利益相关方，即码头运营商和船舶承运人。目标函数涵盖了双方的运营成本。本节通过开展各种敏感性分析，对不同的运营成本进行拆分和分析。

在表12中，我们按码头区段长度对实例进行分组。这里可以看到一种自然的权衡关系。较短的区段长度能够提供更精细的靠泊位置划分，因此有可能获得质量更好的解。然而，这会增加问题的复杂性；对于我们的方法而言，结果是每小时迭代次数减少。尽管迭代次数不到区段长度为80米的实例的四分之一，该方法仍为区段较短的实例找到了更好的解。目标值的改善主要体现为延误减少和等待时间增加。

表13：按每个港口外部船舶数量分组的实例运营成本。

外部船舶 每个港口	等待	装卸	延误	Fuel	总计	惩罚
0	451	538	2365	1278	4632	0.06
5	287	538	3022	1275	5122	0.04
10	301	538	3227	1275	5341	0.05
20	461	538	2718	1278	8300	0.11

表14：不同燃料价格下每艘船舶的平均燃油消耗量和航行速度

编号 船舶数量	燃料价格（美元/公吨）					
	200		500		1100	
	平均燃油消耗量 （每艘船的公吨数）	平均值 速度（节）	平均燃油消耗量 （每艘船的公吨数）	平均值 速度（节）	平均燃油消耗量 （每艘船的公吨数）	平均值 速度（节）
30	51.04	17.12	50.53	17.04	50.37	17.01
50	51.50	17.10	51.11	17.04	50.99	17.02
70	52.14	17.19	51.35	17.08	51.07	17.03
平均值	51.56	17.14	51.00	17.05	50.81	17.02

装卸成本和燃油成本基本相近。船舶时间窗或港口挂靠已提前规划，并考虑了较低的航行速度。再加上燃油成本占总成本的很大一部分，因此在大多数解中，船舶的航行速度已经是最低速度（见表14）。

我们考察的另一个运营方面是外部船舶对规划过程的影响。我们针对每个港口外部船舶数量不同的实例求解问题，数量从零艘到每个港口二十艘不等。结果汇总于表13。结果支持了这样一种观点：每个港口的外部船舶数量增加，会导致泊位分配更加拥堵，进而提高运营成本。在这种情况下，港口拥堵体现在惩罚 列中，该列表示每个实例中超过最晚完成时间的港口访问平均次数。由于此类延误会受到严厉惩罚，因此在这一方面的改进会显著影响目标值。这些结果还表明，参与协作问题的船舶数量增加时，此类问题的影响程度可能显著增大。参与协作的船舶越多，潜在的联合节省就越大，码头在规划方面也越灵活。

如前所述，燃油消耗是承运人的主要成本驱动因素。过去两年，受全球社会经济和政治局势影响，燃料价格大幅波动。我们假设船舶使用超低硫燃料油（VLSFO），其估计价格为每公吨500美元。然而，过去两年这种燃料的价格在每公吨200至1100美元之间波动。因此，我们还使用每公吨200美元和1100美元的燃料价格测试了该方法（Ship & Bunker，2022）

表14展示了按船舶数量相同的实例分组后，每艘船舶的平均燃油消耗量与航行速度。我们观察到，当燃料价格从每公吨500美元降至每公吨200美元时，平均燃油消耗量可能增加超过半公吨。这种差异在船舶数量较多的实例中更为明显，因为更多船舶会略微加快航行速度，以便更早抵达下一个港口并获得更好的服务。然而，当燃料价格升至每公吨500美元以上时，燃油消耗的降幅相对较小。主要原因在于总体航行速度较低。燃油成本已占运营成本的很大一部分，而解表明，在大多数情况下，船舶的航速接近最低的17节。我们观察到，燃料价格较低时，平均航行速度略有上升；

但实例规模对速度没有影响。Venturini等人（2017）进行的类似敏感性分析也显示了相似的行为。

6. 结论

本文研究了海事协同物流中的一个新兴问题，该问题整合了航运承运人和码头运营商双方的运营。我们提出了一个针对包含航速优化的多港口连续泊位分配问题的新混合整数规划公式，并设计了一个用于求解该问题的ALNS算法。该ALNS算法利用一组多样化且针对问题定制的插入和移除方法，并通过优先采用性能较好的方法来引导搜索。该算法的模块化特性可用于开发面向码头运营商的决策支持工具，运营商的经验也有助于设计新的定制算子。此外，在计算性能方面，该启发式方法能够为规模大于文献所研究实例的问题找到高质量的解，其表现优于CPLEX等商业求解器。我们还从航运承运人和码头运营商的运营成本角度研究了该问题的实际影响，并分析了由此产生的泊位计划和航行速度的质量。我们认为，开展此类协同有助于降低利益相关方的总体成本，同时也能因潜在的燃油消耗降低而带来环境效益。

本研究的某些方面仍有待后续研究或探索。就求解方法而言，插入算子是计算复杂度方面的主要瓶颈。未来可以探索更简单的插入算子或其他启发式变体。进一步研究该方法的可扩展性也可能具有重要意义。毫无疑问，该启发式方法比 CPLEX 的可扩展性更好，而小规模实例的结果表明，自适应大邻域搜索能够获得近似最优解。对于更大规模的实例，CPLEX 的最优性间隙显著增大，且下界变得难以实际应用。最后，将转运或中断管理等实际因素纳入考量，是一个颇具吸引力的研究方向。我们设想采用随机规划等框架来解决这类问题。总而言之，这类研究凸显了协同物流的潜在影响，以及运输业中一体化的价值。

致谢：作者感谢丹麦海事基金对本研究的支持。

参考文献

Ans´otegui, C., Pon, J., Sellmann, M., 2021. 提升进化算法配置。《数学与人工智能年鉴》。 URL https://doi.org/10.1007/s10472-020-09726-y APM Terminals, 2022 https://www.apmterminals.com/en/bremerhaven/ a. practical-information/practical-information 2022 11 4 APM Terminals 不来梅哈芬 NTB。, 访问日期：--。APM Terminals, 2022 https://www.apmterminals.com/en/maasvlakte/ b. about/our-terminal 2022 09 15 APM Terminals 马斯弗拉克特二期。, 访问日期：--。Bierwirth, C., Meisel, F., 2015. 集装箱码头泊位分配与岸桥调度问题的后续综述。《欧洲运筹学杂志》244 (3), 12689, 675–689。Br¨aysy, O., 2003. 求解带时间窗车辆路径问题的反应式变邻域搜索。《INFORMS计算杂志》15 (4), 347–368。Carlo, H. J., Vis, I. F., Roodbergen, K. J., 2014. 集装箱码头的运输作业：文献综述、趋势、研究方向与分类体系。《欧洲运筹学杂志》236 (1), 1–13。

Cheimanoff, N., Fontane, F., Kitri, M. N., Tchernev, N., 2022. 综合泊位分配与特定时不变岸桥分配问题的精确方法和启发式方法。《计算机与运筹学》141, 105695。Cordeau, J. F., Laporte, G., Legato, L., Moccia, L., 2005. 泊位分配问题的模型与禁忌搜索启发式算法。《交通科学》39 (4), 526–538。Du, Y., Chen, Q., Quan, X., Long, L., Fung, R. Y., 2011. 考虑燃油消耗与船舶排放的泊位分配。《交通研究E辑：物流与交通评论》47 (6), 1021–1037。Dulebenets, M. A., 2018. 班轮运输船舶调度问题的综合多目标优化模型。《国际生产经济学杂志》196, 293–318。Dulebenets, M. A., 2019. 通过应用高效的协作协议，最大限度地降低班轮航线服务成本。《IEEE智能交通系统汇刊》20 (1), 8315131。Dulebenets, M. A., Golias, M. M., Mishra, S., 2018. 超额需求下泊位分配的协作协议。《人工智能工程应用》69, 76–92。Dulebenets, M. A., Pasha, J., Abioye, O. F., Kavoosi, M., 2019. 班轮运输中的船舶调度：批判性文献综述与未来研究需求。《柔性服务与制造期刊》33 (1), 43–106。Eurogate, 2022. Eurogate 汉堡。http://www1.eurogate.de/en/EUROGATE/Terminals/Hamburg，访问日期：2022-11-4。Fagerholt, K., 2001. 采用软时间窗的船舶调度：一种基于优化的方法。《欧洲运筹学杂志》131 (3), 559–571。Fagerholt, K., Laporte, G., Norstad, I., 2010. 通过优化航运航线上的航速减少燃料排放。《运筹学会杂志》61 (3), 523–529。Glover, F., 1992. 旅行商问题的新型驱逐链与交替路径法。《计算机科学与运筹学》。《其交叉领域的新进展》，491–508。Guan, Y., Cheung, R. K., 2005. 泊位分配问题：模型与求解方法。《集装箱码头与自动化运输系统：物流控制问题与定量决策支持》，141–158。Guo, L., Zheng, J., Du, H., Du, J., Zhu, Z., 2022. 考虑合作班轮承运人的泊位指派与分配问题。《交通研究E辑：物流与交通评论》164, 102793。Hellsten, E. O., Sacramento Lechado, D., Pisinger, D., 2020. 用于多码头港口支线船舶路径规划与调度的自适应大邻域搜索启发式算法。《欧洲运筹学杂志》287 (2), 682–698。Imai, A., Sun, X., Nishimura, E., Papadimitriou, S., 2005. 集装箱港口的泊位分配：采用连续位置空间方法。《交通研究B辑：方法论》39 (3), 199–221。国际海事组织, 2018年4月。《国际海事组织初步战略：减少船舶温室气体排放》。技术报告 MEPC.304(72), 国际海事组织，（访问日期：2020年5月4日）。URL http://www.imo.org/en/OurWork/Environment/PollutionPrevention/AirPollution/Pages/GHG-Emissions.aspx Iris, C., Lam, J. S. L., 2018. 连续泊位分配与岸桥分配模型：计算比较。《IEEE工业工程与工程管理国际会议》2017-, 374–378。Iris, C., Pacino, D., Røpke, S., 2017. 综合泊位分配与岸桥指派问题的改进模型及自适应大邻域搜索启发式算法。《交通研究E辑：物流与交通评论》105, 123–147。Kim, K. H., Moon, K. C., 2003. 采用模拟退火进行泊位调度。《交通研究B辑：方法论》37 (6), 541–560。Kordi´c, S., Davidovi´c, T., Kova˘c, N., Dragovi´c, B., 2016. 精确求解离散与混合泊位分配问题的组合方法。《应用数学建模》40 (21-22), 8952–8973。Lalla-Ruiz, E., Meli´an-Batista, B., Moreno-Vega, J. M., 2016. 用于泊位调度的协作搜索。《知识工程评论》31 (05), 498–507。Lim, A., 1998. 泊位规划问题。《运筹学快报》22 (2-3), 105–110。Lyu, X., Negenborn, R. R., Shi, X., Schulte, F., 2022. 用于扰动恢复的协作式泊位规划方法。《IEEE智能交通系统开放期刊》3, 153–164。Marine Traffic, 2023. 港口挂靠数据。https://www.marinetraffic.com/en/data/，访问日期：2023年1月14日。Martin-Iradi, B., Pacino, D., Ropke, S., 2022a. 带航速优化的多港口连续泊位分配问题。载于：de Armas, J., Ramalhinho, H., Voß, S.（编），《计算物流学》。施普林格国际出版公司，Cham，第31–43页。Martin-Iradi, B., Pacino, D., Ropke, S., 2022b. 带航速优化的多港口泊位分配问题：精确方法与合作博弈分析。《交通科学》56 (4), 972–999。Meisel, F., Bierwirth, C., 2009. 将起重机作业效率纳入泊位分配问题的启发式算法。《交通研究E辑：物流与交通评论》45 (1), 196–209。Potvin, J., Rousseau, J., 1993. 一种用于车辆路径与调度问题的并行路径构建算法，29

时间窗。《欧洲运筹学杂志》66 (3), 331–340。Rego, C., 1998. 一种用于车辆路径问题的子路径弹出方法。《管理科学》44 (10), 1447–1459。Ropke, S., Pisinger, D., 2006. 一种用于带时间窗的取送货问题的自适应大规模邻域搜索启发式。《交通科学》40 (4), 455–472。Shaw, P., 1998. 使用约束规划和局部搜索方法求解车辆路径问题。《计算机科学讲义》1520, 417–431。Ship & Bunker, 2022 https://shipandbunker.com/prices/av/global/.av-g20-global-20-ports-average 2022 12 18 全球平均船用燃料价格 - 超低硫燃料油。, 访问日期：- -。Steenken, D., Voß, S., Stahlbock, R., 2004. 集装箱码头运营与运筹学——分类与文献综述。Or Spectrum 26 (1), 3–49。Sun, B., Niu, B., Xu, H., Ying, W., 2018. 考虑不可预测扰动的港口与航运公司协同优化。ICNC-FSKD 2018——第14届国际自然计算、模糊系统与知识发现会议，8686901, 113–118。联合国贸易和发展会议, 2020. 2020年海上运输回顾。技术报告，联合国贸易和发展会议。Venturini, G., Iris, C., Kontovas, C. A., Larsen, A., 2017. 考虑航速优化和排放因素的多港口泊位分配问题。《交通研究D辑：交通与环境》54, 142–159。Wang, S., Liu, Z., Qu, X., 2015. 泊位分配的协作机制。《先进工程信息学》29 (3), 572, 332–338。Yu, J., Tang, G., Song, X., 2022. 考虑船舶服务差异化的船舶航速优化与泊位分配及岸桥指派的协同。《交通研究E辑：物流与交通评论》160, 102651.