多目标约束处理方法求解泊位岸桥联合分配问题

黄 涵, 季 彬†

(中南大学 交通运输工程学院, 湖南 长沙 410075)

摘要: 为了制定合理高效的泊位岸桥联合分配方案, 加快船舶周转, 本文针对船舶动态到港的连续泊位建立了以船舶总在港时间最短为目标的泊位岸桥联合分配混合整数非线性模型. 通过多目标约束处理策略将复杂约束的违反程度转化为另一个目标, 从而将原单目标优化模型转化为双目标优化模型, 并用基于快速非支配排序的多目标遗传算法(NSGA-II)对其进行求解. 同时, 针对问题特点, 分别设计了基于调整、惩罚函数、可行解优先和综合约束处理策略的单目标遗传算法对原模型进行求解. 通过多组不同规模的标准算例对本文的方法进行测试, 验证了基于多目标约束处理策略的方法求解效果相较于单目标约束处理策略的方法更加高效和稳定.

关键词: 泊位岸桥联合分配; 多目标优化; 约束处理; 遗传算法

引用格式: 黄涵, 季彬. 多目标约束处理方法求解泊位岸桥联合分配问题. 控制理论与应用, 2023, 40(4): 761 – 771

DOI: 10.7641/CTA.2022.10891

Multi-objective constraint handling method for solving berth allocation and quay crane assignment problem

HUANG Han, JI Bin†

(School of Traffic and Transportation Engineering, Central South University, Changsha Hunan 410075, China)

Abstract: A reasonable and efficient integrated berth allocation and quay crane assignment scheme can improve the efficiency of port operation and speed up vessels turnover. To achieve this, a mixed integer nonlinear model is established for continuous berth with dynamic vessel arrival, aiming at minimizing the total stay time of vessels in port. In this study, a multi-objective constraint handling strategy is presented to convert the complex constraints into an objective, so as to transform the original single-objective optimization model into a dual-objective optimization model, which is further solved by the non-dominated sorting based genetic algorithm (NSGA-II). In addition, other constraint handling strategies, such as the repair method, penalty function strategy, superiority of feasible solutions strategy and comprehensive strategy, are developed according to the characteristics of the problem, and incorporate with genetic algorithm to solve the original model. The test results on numerous instances show that the multi-objective constraint handling-based method is more efficient and stable than the single-objective constraint handling-based methods.

Key words: berth allocation and quay crane assignment; multi-objective optimization; constraint handling; genetic algorithms

Citation: HUANG Han, JI Bin. Multi-objective constraint handling method for solving berth allocation and quay crane assignment problem. *Control Theory & Applications*, 2023, 40(4): 761 – 771

1 引言

随着全球化程度的加深, 世界各国之间的贸易往来日益密切. 港口作为现代物流的运输枢纽, 在经济发展中发挥着不可替代的作用, 其作业能力对于货物能否快速周转至关重要. 在港口作业系统中, 泊位和岸桥是港口装卸作业最为核心稀缺的资源, 其作业效率直接决定了港口的运作效率. 面对日益增长的运输需求, 如何合理地分配泊位岸桥、提高资源利用率和装卸作业的效率, 成为港口不得不考虑的问题.

泊位分配问题 (berth allocation problem, BAP) 按照泊位空间布局可分为离散型泊位分配问题和连续型泊位分配问题. Imai等[1]以船舶总在港时间最短和与靠泊次序相关的船主不满度最低为目标, 建立了针对离散型BAP的混合整数非线性双目标模型. 随后,

收稿日期: 2021−09−22; 录用日期: 2022−04−28.

†通信作者. E-mail: cumtjibin@126.com; Tel.: +86 18908468853.

本文责任编委: 武玉强.

国家自然科学基金项目(72001216, 71672193), 湖南省自然科学基金项目(2020JJ5780)资助.

Supported by the National Natural Science Foundation of China (72001216, 71672193) and the National Natural Science Foundation of Hunan Province (2020JJ5780).

Imai等[2]和Nishimura等[3]考虑船舶靠泊时间约束把模型扩展成船舶动态到港的动态离散型泊位分配, 又把船舶的靠泊优先级引入模型[4]. 离散型BAP复杂度相对较低, 易于求解, 但泊位利用率低, 随着运输需求的增长, 离散型BAP渐渐不能满足生产要求, 学界开始关注连续型BAP. Imai等[5]基于装卸时间取决于靠泊位置的假设提出了针对连续型泊位分配的混合整数非线性模型, 并用求解离散型BAP的启发式算法给出了连续型BAP的解上界和下界. Lim[6]通过将连续泊位分配问题转化成二维装箱问题, 证明了连续泊位分配问题是NP难(nondeterministic polynominal-hard)问题, 精确算法难以有效求解大规模算例, 不能满足应用需求. 因此, 连续泊位分配问题一般用启发式算法求解, 如禁忌搜索算法[7]、遗传算法[8]、粒子群算法[9]等广泛应用于求解该问题.

以上研究均假设装卸时间是固定值或由靠泊位置决定, 但在实际工作中, 装卸时间往往与分配给船舶的岸桥数量相关, 由此产生了泊位岸桥联合分配问题(berth allocation and quay crane assignment problem, BACAP). 由于BACAP有着各种各样的物理、技术约束和优化需求, 该问题的模型较为多样化. Bierwirth和Meisel[10–11]对BACAP的已有成果进行了文献综述, 对现有的大部分模型进行了归类. Park和Kim[12]首次研究了BACAP, 提出一种混合整数非线性规划模型(mixed integer nonlinear programming, MINLP) 并用一种结合了次梯度法和动态规划的两阶段启发式算法求解. Meisel和Bierwirth[13]考虑岸桥间的干扰和距离最佳靠泊位置的偏离程度对装卸时间的影响建立了MINLP模型, 提出了基于构造启发式算法和局部优化算法结合的禁忌搜索算法对模型进行求解, 并提供了标准测试算例. Wang等[14]认为为船舶分配较多岸桥可以减少船舶离港时间延误, 但由于存在岸桥间干扰会产生额外成本, 为此, 该团队考虑船舶离港延误和岸桥运营成本最小化两个目标, 建立了BACAP双目标整数规划模型, 并提出一种启发式方法寻找非支配解集. 宋云婷等[15]以船舶准时离港率最大化和码头装卸作业总成本最小化为目标建立双目标模型, 并提出改进的非支配排序遗传算法进行求解. 吴迪等[16]建立了BACAP的MINLP模型, 并针对模型特点提出了一种进化求解算法. Correcher等[17]建立了连续泊位下BACAP的混合整数线性规划模型, 并进一步考虑岸桥具体分配对该模型进行扩展, 提出融合切平面的迭代算法对其进行求解. Xiang和Liu[18]针对船舶到港时间延误和装箱量变化的不确定性环境, 建立了BACAP鲁棒模型并提出一种基于分解的求解方法.

由于BACAP的复杂性, 现有研究中绝大多数采用启发式算法求解, 约束处理策略对启发式算法求得的可行解的优劣有很大的影响[19]. 已有研究中多采用调整策略[13]、惩罚函数策略[15]和可行解优先策略[16]等. 调整策略是指调整不可行解使其成为可行解, 但其对问题依赖性高, 且可能在调整过程中引入系统偏差. 惩罚函数策略因其通用性好、方便实现得到广泛应用, 但平衡目标函数和惩罚项一直是较为困难的优化问题, 对参数设置要求很高. 可行解优先策略规则简单, 可以提高算法效率, 但在可行域不连通的问题中, 可能会陷入局部最优[20]. 为此, 本文建立了BACAP的MINLP模型, 并提出采用多目标约束处理策略[21](multi-objective constraint handling, MOCH)来处理复杂的时空耦合约束, 通过把船舶不可重叠约束和岸桥总数约束的违反度转化为优化目标, 将带复杂约束的单目标模型转化成约束易处理的双目标模型, 继而采用多目标算法对其进行求解. 该基于MOCH的优化方法保留了大量不可行解参与进化, 从而可以充分挖掘不可行解中的有效信息, 对可行解进行优化, 克服了传统约束处理策略参数敏感、约束调整过程复杂等不足. 为了验证该策略的高效性, 本文针对BACAP设计了调整策略、惩罚函数策略、可行解优先策略和综合策略, 并采用遗传算法分别结合这4种策略对原模型进行求解. 最后, 在不同规模的算例下对MOCH与另外4种策略的求解结果进行对比、分析, 证明了基于MOCH的优化方法可以更高效、稳定地实现BACAP求解.

2 泊位岸桥联合分配模型

2.1 问题描述

船舶在抵港前将船型、预计到达时间和装箱量等信息预报给挂靠港口, 港口工作人员为其制定泊位分配计划和岸桥分配计划, 该计划可以映射到一个二维时空图中, 如图1所示. 图中横轴代表码头岸线, 纵轴表示时间, 岸桥沿岸线布置, 在与岸线平行的轨道上移动. 坐标系中矩形表示计划到港船舶, 矩形长度即船舶长度, 高度即装卸时间. 船舶到港后, 调度员为其指定靠泊时间、位置和具体岸桥, 要满足船舶之间在时空上不可冲突的约束, 即矩形之间不可重叠. 岸桥数量有限且固定, 工作过程中不能跨越其他岸桥. 在整个规划期, 任意时刻调用岸桥总数不能超出. 本文针对船舶动态到港的连续型泊位, 综合考虑岸桥不可交叉约束、岸桥间干扰和距离最佳靠泊位置偏离程度对装卸时间的影响以及岸桥的具体分配等要素, 以船舶总在港时间最短为目标构建BACAP的MINLP模型.

模型基本假设如下: 1) 无水深、船舶吃水、机械故障等制约; 2) 每艘船舶都有最佳靠泊位置, 装卸时间与其靠泊位置和最佳靠泊位置的距离成正比, 与其分配岸桥数量成反比; 3) 每艘船舶都有可分配岸桥数量的范围, 同时服务于一艘船舶的岸桥间存在干扰; 4) 岸桥在轨道上移动, 不能交叉跨越; 5) 分配给每艘

船舶的岸桥数量和具体岸桥在装卸途中不可变更.

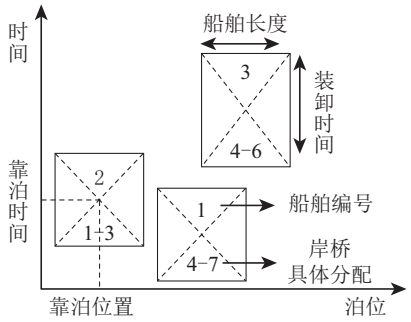


图 1 泊位岸桥联合分配示意图

Fig. 1 Diagram of the BACAP

2.2 模型构建

1) 参数.

V: 规划期内到港船舶集合, $V=\{1,2,\cdots,N\}$; Q: 码头前沿岸桥总数; L: 泊位长度; l_i: 船舶i的长度(已考虑安全距离); c_i: 完成船舶i上集装箱的装卸工作所需工时; A_i: 船舶i的预计到达时间; B_i: 船舶i的最佳靠泊位置; $\overline{q_i}$: 可以分配给船舶i的最大岸桥数量; $\underline{q_i}$: 至少要分配给船舶i的最小岸桥数量; α: 最佳靠泊位置偏离系数; β: 岸桥间干扰系数; M: 一个足够大的正数.

2) 决策变量.

S_i: 整型变量, 船舶i的靠泊时间; P_i: 整型变量, 船舶i的靠泊位置; q_i: 整型变量, 分配给船舶i的岸桥数量; $q_i^{\rm l}$: 整型变量, 分配给船舶i的最左侧岸桥编号.

3) 中间变量.

W_i: 整型变量, 船舶i的等待时间; H_i: 整型变量, 船舶i的装卸时间; D_i: 整型变量, 船舶i的离泊时间; $q_i^{\rm r}$: 整型变量, 分配给船舶i的最右侧岸桥编号; σ_{ij}: 0–1变量, 船舶i, j的靠泊区间没有重叠时为1, 否则为0; δ_{ij}: 0–1变量, 船舶i, j的靠泊时间窗没有重叠时为1, 否则为0; γ_{ij}: 0–1变量, 船舶i的靠泊位置在船舶j的左侧时为1, 否则为0.

$$\min f=\sum_{i\in v}(D_i-A_i), \tag{1}$$

$$P_i-\frac{l_i}{2}\geqslant 0,\ \forall i\in V, \tag{2}$$

$$P_i+\frac{l_i}{2}\leqslant L,\ \forall i\in V, \tag{3}$$

$$S_i\geqslant A_i,\ \forall i\in V, \tag{4}$$

$$\underline{q_i}\leqslant q_i,\ \forall i\in V, \tag{5}$$

$$q_i\leqslant \overline{q_i},\ \forall i\in V, \tag{6}$$

$$H_i=\frac{(1+\alpha|P_i-B_i|)c_i}{q_i^{\beta}},\ \ \forall i\in V, \tag{7}$$

$$D_i=S_i+H_i,\ \forall i\in V, \tag{8}$$

$$|P_i-P_j|\sigma_{ij}\geqslant\frac{(l_i+l_j)}{2}\sigma_{ij},\ \forall i,j\in V,\ j\neq i, \tag{9}$$

$$|\frac{S_i+D_i}{2}-\frac{S_j+D_j}{2}|\delta_{ij}\geqslant\frac{H_i+H_j}{2}\delta_{ij},\ \forall i,j\in V,\ j\neq i, \tag{10}$$

$$\sigma_{ij}+\delta_{ij}\geqslant 1,\ \forall i,j\in V,\ j\neq i, \tag{11}$$

$$q_i=q_i^{\rm r}-q_i^{\rm l}+1,\ \forall i\in V, \tag{12}$$

$$q_i^{\rm r}\leqslant Q,\ \forall i\in V, \tag{13}$$

$$1\leqslant q_i^{\rm l},\ \forall i\in V, \tag{14}$$

$$P_i-P_j\leqslant M(1-\gamma_{ij}),\ \forall i,j\in V,\ j\neq i, \tag{15}$$

$$q_i^{\rm r}<q_j^{\rm l}+M(1-\gamma_{ij}+\delta_{ij}),\ \forall i,j\in V,\ j\neq i, \tag{16}$$

$$\sigma_{ij},\gamma_{ij},\delta_{ij}\in\{0,1\},\ \forall i,\ j\in V,\ j\neq i. \tag{17}$$

目标函数(1)表示最小化船舶在港时间; 约束(2)–(3)限制了船舶必须靠泊在泊位范围内; 约束(4)表示船舶只能在到达之后开始靠泊; 约束(5)–(6)表示分配给每艘船舶的岸桥数量应在允许范围内; 约束(7)–(8)分别定义了船舶装卸时间和离港时间; 约束(9)–(11)保证任意两艘船舶在时空上不重叠; 约束(12)定义了分配给每艘船舶的岸桥数量与具体岸桥编号的关系; 约束(13)保证使用的岸桥不超过岸桥总数; 约束(14)表示至少要从第1个岸桥开始分配; 约束(15)定义了船舶间相对靠泊位置; 约束(16)表示岸桥不可交叉; 约束(17)定义了0–1变量的取值范围.

3 BACAP求解

3.1 基于多目标约束处理策略的求解方法

基于MOCH的求解方法首先需要对第2.2节中的BACAP模型进行转化, 将其中难以处理的复杂约束转化为目标, 从而将原BACAP单目标优化模型转化为双目标模型, 进而采用多目标优化算法对其进行求解, 求得的Pareto前沿中约束违反度为0的解即为所求近似最优解. 鉴于非支配排序遗传算法(non-dominated sorting genetic algorithm-II, NSGA-II)高效的求解性能和广泛成功的应用, 本文采用NSGA-II求解双目标BACAP模型, 其关键技术如下.

3.1.1 约束表达和模型转化

MOCH应用于带M个约束的单目标优化问题一般有两种方案: 一种是将所有约束违反程度一起作为一个目标, 从而将单目标优化模型转化成无约束双目标优化模型; 另一种是将每部分约束违反都转化为一个目标, 从而将原模型转化为$M+1$个目标的无约束优化模型. 考虑到第2种超多目标优化问题会因为解之间的非支配关系弱等问题大大增加问题求解难度[22], 本文采用第1种方案.

考虑岸桥具体分配的BACAP求解可以分两个阶段: 第1阶段决策船舶靠泊时间、靠泊位置和分配岸桥数量; 第2阶段进行岸桥具体分配, 其本质是一个无目标的可满足性问题, 即在第1阶段结果基础上满足约束(14)–(16), 可采用动态规划算法得到具体岸桥分配

最优方案[23]. 分析模型不难发现约束(2)–(6)是变量边界条件, 可通过合理的变量初始化满足, 且本文采用的交叉变异算子在搜索过程中不会破坏其边界. 约束(7)–(8)是中间变量与决策变量的等式关系, 可以转化为决策变量的函数融入约束(10). 同样, 约束(12)给出了分配给船舶的最右侧岸桥编号与决策变量(岸桥数量和最左侧岸桥编号)之间的等式关系, 可以融入约束(13). 然而, 仍然存在两部分复杂约束难以解决: 第一, 船舶时空不可重叠约束(9)–(11)是非线性的; 第二, 岸桥总数约束(13)的违反程度取决于岸桥具体分配结果, 而岸桥具体分配又和船舶相对时空位置耦合, 所以难以处理. 针对这两部分复杂约束, 本文采用MOCH对其进行处理: 将其合理转化为另一个目标, 从而将原带复杂约束的单目标BACAP模型转化为只有简单约束的双目标模型.

约束(9)–(11)的违反程度可以用任意两艘船舶在时空位置上的重叠面积之和度量, 即

$$V_1 = \sum_{i,j\in V, j\neq i} (\max(\frac{l_i+l_j}{2} - |P_i - P_j|, 0) \times \max(\frac{H_i+H_j}{2} - |\frac{S_i+D_i}{2} - \frac{S_j+D_j}{2}|, 0))^{0.5} = \sum_{i,j\in V, j\neq i} f_{V_1}(P_i, P_j, S_i, S_j, q_i, q_j). \tag{18}$$

约束(13)的违反程度可以用岸桥具体分配后超出的岸桥数量之和度量, 即

$$V_2 = \sum_{i\in V} \max(q_i^r - Q, 0) = \sum_{i\in V} f_{V2}(q_i, q_i^l). \tag{19}$$

所以本问题基于多目标约束处理策略的双目标模型是

$$\min f_1 = \sum_{i\in v} (D_i - A_i), \tag{20}$$

$$\min f_2 = V_1 + V_2, \tag{21}$$

$$\text{s.t. 式(2)–(6), (10), (14)–(16).}$$

值得注意的是, 本文定义的约束违反度目标函数中并没有对V_1, V_2两个量纲不一致的量进行归一化处理, 这是因为MOCH只关注其数学意义, 而不关注其物理意义. V_1+V_2大于0和等于0两种情况, 分别对应着解不可行和可行的判定, 而与V_1, V_2各自的绝对值或相对值无关, 所以本文并未割裂二者进行选择, 这不会影响MOCH的可行性和有效性.

3.1.2 染色体编码、解码规则

本文根据问题特征设计了图2所示三段式染色体, 分别表示船舶靠泊位置、靠泊时间和分配岸桥数量, 均采用自然数编码.

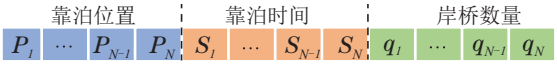


图 2 染色体示意图

Fig. 2 Diagram of the chromosome

解码分为两个阶段. 第1阶段, 根据个体的染色体信息, 可得到所有船舶在二维时空坐标上的分布和分配岸桥数量(如图3(a)所示), 并通过式(20)求得目标函数值. 但具体岸桥与船舶的对应关系不能直接从染色体中得出, 需进一步分配. 第2阶段, 根据船舶相对位置和在泊时间为其分配具体的岸桥. 默认岸桥在泊位上从左到右排列, 序号依次增大, 从泊位最左侧开始依次按照染色体中确定的数量为船舶分配岸桥. 首先, 搜索靠泊位置最靠左的船舶, 从第1个岸桥开始为其分配岸桥. 然后, 在未分配岸桥的船舶中搜索靠泊位置最靠左的, 再在已分配岸桥的船舶中寻找与该船舶在泊时间存在重叠的船舶, 找到其中最右侧岸桥编号最大的, 从下一个岸桥开始为该船舶分配岸桥. 重复上述操作, 直至所有船舶都被分配了具体的岸桥. 解码完成后, 可进一步得到具体岸桥分配方案(如图3(b)所示). 按照解码步骤可以满足岸桥不可交叉约束, 但从图中可以看出, 这是一个不可行解, 违反了船舶时空位置不可重叠约束和岸桥总数约束(假设共有6台岸桥), 其违反程度可分别根据式(18)–(19)计算, 另一个目标函数值可通过式(21) 求得.

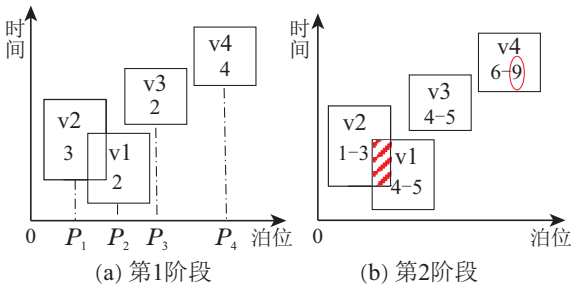


图 3 染色体解码示意图

Fig. 3 Diagram of the chromosome decoding

3.1.3 种群初始化

该问题模型约束较多, 为了缩小求解空间、降低求解难度, 在种群初始化时应在允许范围内生成基因. 靠泊位置子染色体上的基因上下界分别为$L - l_i/2$和$l_i/2$, 靠泊时间子染色体上的基因生成范围是船舶预计到达时间到规划期之间, 岸桥数量子染色体上的基因在最小、最大可分配岸桥数量之间生成, 具体生成方法如下:

$$x_i = \text{round}(x_i^{\min} + \text{rand} \cdot (x_i^{\max} - x_i^{\min})), \tag{22}$$

其中: x_i是基因值, $x_i^{\max}$ 和$x_i^{\min}$分别是基因上、下界, rand是0~1之间的随机数, round$(\cdot)$是四舍五入函数.

3.1.4 选择、交叉、变异算子

本文采用二元锦标赛选择算子[24], 在种群中随机抽取2个个体进行竞争(锦标赛), 将其中最优的个体作为母代进行交叉变异产生下一代种群, 直至子代数量等于种群规模. 该算子可以最大程度保证优胜劣汰, 也给较差的个体一定生存空间, 加快种群收敛速度的

同时保证种群多样性. 交叉算子采用两点交叉和实数交叉相结合的策略, 在染色体上3N个基因位中随机生成两个点(N是船舶数量), 子代继承母代在两个点之前和之后的基因, 两点及两点之间的基因通过母代基因的线性组合生成. 变异操作选择单点变异法, 在染色体中随机选择一个点, 子代在该处基因在允许范围内重新生成基因值, 其他基因位继承母代基因.

3.1.5　可行解的选择

当且仅当第2个目标即约束违反为零时, 解是可行的. 由于NSGA-II基于Pareto支配关系和拥挤距离进行精英保留, 每代种群只会保留极少数可行解, 算法的目的不在于找到很多的可行解, 而在于找到一个足够好的可行解. 非支配排序和精英保留策略一方面保证在迭代中可以保留大量目标值较好的不可行解作为基因库, 与可行解重组可能会带来更高质量的可行解, 这样可以充分利用不可行解中的有用信息, 避免算法陷入局部最优; 另一方面, 一旦出现可行解, 根据Pareto支配关系, 该可行解一定不被其他不可行解支配, 会被一直保留, 除非找到更好的可行解支配当前可行解. 图4是迭代完成后种群中个体的分布示意图, 可以看出种群中仅含2个可行解, 其余均为不可行解. 靠近原点的系列解代表算法求得的Pareto前沿, 其与横坐标相交的个体即为求得的近似最优解.

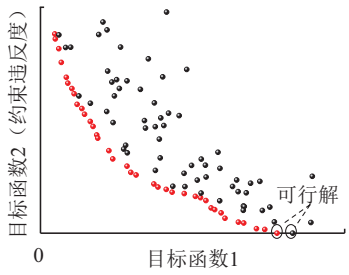


图 4　种群中个体分布示意图

Fig. 4　Diagram of individuals distribution in the population

3.2　单目标求解方法

本文采用遗传算法对第2.2节模型进行求解, 并针对问题特点, 分别设计了调整策略、惩罚函数策略、可行解优先策略和综合策略进行约束处理. 4种约束处理策略在算法中的不同之处体现在锦标赛选择和精英保留机制. 锦标赛选择时, 调整策略选择目标值较好的个体作为母本进行交叉操作, 惩罚函数策略选择适应度高的个体, 而可行解优先和综合策略则按照可行解优先原则选取母本. 精英保留时, 调整策略按照目标值从小到大的顺序选择进入下一次迭代的个体, 惩罚函数和综合策略按照适应度从高到低的顺序, 可行解优先策略则按照约束违反度最小且目标值最好的原则选择. 另外, 为求证多目标约束处理策略的有效性, 本节采用与多目标约束处理策略相同的编码、解码规则、初始种群生成方式和选择、交叉、变异算子.

3.2.1　调整策略(repair method, RM)

本文提出一种调整策略, 对分配方案中的船舶逐个进行调整, 具体实现过程如下:

步骤 1　从未调整的船舶集合V_{NA}中找出最早开始靠泊的船舶;

步骤 2　检查该船舶与已完成调整的船舶集合V_{A}之间是否违反不可重叠约束, 如违反则将其靠泊时间设为与其重叠的船舶的离港时间, 并更新其离港时间, 返回步骤1; 否则, 进入步骤3;

步骤 3　对该船舶和V_{A}里的船舶进行具体岸桥分配, 若存在船舶为其分配的最右侧岸桥编号大于岸桥总数, 则将其靠泊时间推迟至V_{A}中有船舶离开, 并更新其离港时间, 返回步骤1; 否则, 该船调整完毕;

步骤 4　重复步骤1–3, 直至所有船舶都完成调整.

3.2.2　惩罚函数(penalty function, PF)策略

这种约束处理方式以适应度函数为依据比较解的优劣, 适应度函数要正确合理地反映出解的质量和可行性, 一般是在不可行解的目标函数值后加一个违反约束的惩罚项组成, 本文提出以下适应度函数:

$$\mathrm{fit}=\sum_{i\in V}\frac{\mathrm{BH}_i}{Z-\sum\limits_{i\in V}\mathrm{BH}_i}-\frac{C_1V_1}{V_1^{\max}}-\frac{C_2V_2}{Q},\tag{23}$$

$$V_1^{\max}=\sum_{i,j\in V,\,j\neq i}\left(\frac{l_i+l_j}{2}\cdot\frac{H_i+H_j}{2}\right)^{0.5},\tag{24}$$

其中: V_1, V_2定义同第3.1.1节; BH_i是船舶i在最佳靠泊位置靠泊并分配最大允许数量岸桥时的最短装卸时间; Z是目标函数值; C_1, C_2是根据问题规模设定的常量, 问题规模越大, 约束冲突增大, 数值应相应减小.

3.2.3　可行解优先(superiority of feasible solutions, SF)策略

按照可行解优先的原则选择进入下一次迭代的个体, 将约束违反度作为优先级第一的原则, 目标函数值作为优先级第二的原则, 所以当两个解X_i, X_j进行比较时, 在两种情况下认为X_i比X_j优: 1) X_i的约束违反程度比X_j低; 2) X_i和X_j约束违反程度一样, 但X_i比X_j目标函数值小. 本文SF策略的约束违反度定义与MOCH相同.

3.2.4　综合策略(comprehensive strategy, COM)

在算法中综合使用调整、可行解优先和惩罚函数策略: 首先, 生成解后, 对不可行解进行部分调整. 不考虑岸桥不可交叉约束和具体分配进行基因调整, 使得任意时刻调用的岸桥总数不超过现有岸桥总数, 如图3(a)所示情形, 但若考虑不可交叉约束则可能没有

足够的岸桥以实现具体分配, 这样可以保留部分不可行性, 提高岸桥利用率, 缩短船舶装卸时间; 其次, 在二元锦标赛选择时, 采用可行解优先策略; 最后, 针对不可重叠约束和具体分配后的岸桥总数约束, 将约束违反作为惩罚项加入目标函数值, 以适应度函数为依据选择进入下一代的个体.

4 算例分析

本文采用的NSGA-II和遗传算法均采用C语言编写, 在Windows10操作系统, 英特尔Core i5/3.00 Ghz, 8 GB内存的电脑上运行. 为验证MOCH的有效性, 对基于MOCH和另外4种单目标约束处理策略的算法求解结果进行对比和分析.

本文采用两套算例: 第1套是文献[13]的标准算例, 规划期为168 h. 根据船舶数量, 算例分为20, 30和40这3组, 每种规模包含10个算例. 第2套参照文献[17]将规划期延长至210 h, 根据文献[13]的算例生成规则随机生成50, 60 规模算例, 每种规模包含5个算例. 每个算例中有60%小型船舶(80∼210 m)、30%中型船舶(210∼300 m)和10%巨型船舶(300∼400 m), 船舶的到港时间在0∼168 h内, 总体呈均匀分布. 泊位长度为1000 m, 岸桥数量为10. 岸桥间干扰系数为0.9, 最佳靠泊位置偏离系数为0.01. 但文献[13]对泊位和时间进行了离散化, 且对于靠泊位置的定义与本文略微不同, 因此需对算例数据按照数量关系进行相应调整.

对种群规模和交叉变异算子参数分别在一定范围内取值并设计正交实验, 在不同规模的算例中随机抽取2组对不同参数组合进行测试(给定足够大的迭代次数), 选择20次测试结果平均值最佳的参数组合: 种群规模为100; 交叉概率和变异概率分别为0.8和1/n(n为决策变量个数, $n = 3N$); 交叉系数为0.3. 确定参数后, 通过观察算法的收敛情况确定不同规模算例的最大迭代次数, 在保证算法收敛的前提下,尽量减少计算负担: 设置20, 30, 40, 50, 60规模算例的最大迭代次数分别为1000, 1500, 2000, 2500和3000.

4.1 不同规模算例中不同约束处理策略结果分析

用基于5种约束处理策略的算法分别求解40个算例, 为克服随机性影响, 每个算例运行20次. 对运行结果进行统计得到每种方法求解各个算例的最优值(Bt)、平均值(Ag)、标准差(SD)、平均计算时间(Te)和20次运行中求得可行解的次数(Nfs). 每个算例结果中最小的最优值、平均值和标准差用黑体标出.

表1是20艘船的算例的统计结果, 所有策略在20次迭代中都得到了可行解, 故省略Nfs行. 在所有的算例结果中, MOCH都得到了最小的最优值和平均值, 同时得到了最小的标准差, 表明了该策略求解泊位岸桥联合分配问题的有效性和稳定性. RM稍次于MOCH, 在大多数算例中, 最优值和平均值都优于另外3种约束处理策略, 同时具有较强的稳定性, 但运行时间最长. 另外3种约束处理方式在20规模的算例中的统计数据非常相近.

表 1 20规模算例中不同策略的结果对比

Table 1 Comparison results for the 20-size instances of different constraint handling strategies

		20–001	20–002	20–003	20–004	20–005	20–006	20–007	20–008	20–009	20–010
Bt (h)	MOCH	**184.6**	**138.5**	**176.2**	**159.6**	**139.9**	**158**	**164**	**146.7**	**166.1**	**174.8**
	RM	185.4	140.5	180.6	163.9	143.7	160	165.4	153.4	172.9	179.5
	PF	191.5	140.2	194.7	174.4	142.1	159.4	169.7	151	171.4	193.8
	SF	192.7	141.8	201.4	173.5	153.2	160.6	174.6	159.8	174.8	191.9
	COM	196.1	140.6	200.7	178.9	145.2	160.3	171.9	155.4	173.0	192.3
Ag (h)	MOCH	**186.2**	**138.8**	**180.2**	**163.4**	**140.1**	**158.1**	**165.0**	**148.4**	**169.8**	**176.4**
	RM	196.6	144.1	190.7	174.2	150.5	168.3	171.7	157.8	179.2	191.9
	PF	212.9	152.5	232.3	207.2	161.6	177.4	188.0	174.9	192.5	209.6
	SF	234.6	156.1	233.6	201.2	174.6	177.8	199.5	180.2	189.8	217.3
	COM	227.4	154.4	221.8	201.9	162.2	176.1	192.7	170.3	192.5	218.3
SD	MOCH	**1.2**	**0.2**	**3.5**	**4.0**	**0.2**	**0.1**	**1.4**	**0.7**	**2.5**	**1.6**
	RM	7.3	1.6	7.4	7.8	4.6	5.1	4.0	5.3	5.5	7.5
	PF	16.6	11.0	21.0	22.8	13.8	20.2	19	14.2	15.2	14
	SF	34.6	12.7	24.5	23.4	19.5	29.8	17.6	16.9	14.8	17.3
	COM	24.2	13.2	21.2	25.9	10.7	12.2	12.0	9.5	16.5	17.4
Te (s)	MOCH	9.5	9.6	10.0	9.7	9.6	9.7	9.8	9.6	9.8	9.7
	RM	14.6	14.4	14.9	14.5	14.3	14.4	14.5	14.1	14.4	14.4
	PF	6.3	6.3	6.3	6.4	6.3	6.4	6.4	6.4	6.4	6.4
	SF	6.4	6.3	6.3	6.3	6.2	6.3	6.3	6.4	6.2	6.2
	COM	7.4	7.3	7.3	7.4	7.3	7.4	7.3	7.3	7.3	7.3

表2给出了30规模算例的统计结果. MOCH仍然在所有算例中都取得了最好的最优值、平均值和标准差, 并且这种优势相较于20规模的算例结果变得更加明显. RM在计算结果上仍是次优的, 但随着算例规模增大, 其计算时间大幅增加, 超过1分钟, 是其它约束处理策略的计算时间的近3倍. 另外3种约束处理方式的计算结果差异性逐渐显现, 在大多数算例中, COM可以得到最好的最优解, PF求得的解平均值较好, 而SF求得的结果具有最大的方差, 这表明运用该种策略求解质量不好的同时也不具备良好的收敛性. 另外, PF, SF和COM在小部分算例的20次运算中并没有全部得到可行解, 而MOCH和RM全部得到了可行解, 故省略Nfs行(表3–4同理).

表3给出了40艘船的算例的统计结果, MOCH的优越性得到进一步验证: 在几乎所有算例中, 其最优值、平均值和标准差都远优于其它4种约束处理策略, 仅在个别算例中的平均值或标准差略次于RM. COM在两个算例(40–007, 40–010)的20次迭代中有1次没有得到可行解, PF仅在两个算例(40–001, 40–003)的20次迭代中全部得到了可行解, 而SF在所有算例中都没有得到过20次可行解, 甚至在40–005算例中没有得到可行解, 而且在可求得可行解的算例中最优值和平均值表现也是最差的. 在计算时间方面, MOCH, PF, SF和COM都能在1 min左右完成迭代, 而RM需要约3 min.

表4是随机生成的50, 60艘船算例的统计结果, 可以看出所有方法求得的BACAP方案船舶总在港时间相较于40规模的算例都有大幅的增加, 一方面是因为装卸作业增加导致装卸时间增加, 另一方面是港口装卸作业能力已达到极限, 到港船舶的等待时间大大延长. 在这种情况下, 基于MOCH的方法仍然在几乎所有算例都求得了更好的最优值和平均值, 仅在60–004算例较RM稍差, 同时表现出了较强的求解稳定性, 求解时间也是所有方法里最短的. SF随着算例规模增加, 求解性能急剧下降, 在绝大多数算例中表现最差, 在60规模的算例中仅求得一次可行解. PF和COM求解质量一般, 仍然不能保证每次运行都能求得可行解.

表 2　30规模算例中不同策略的结果对比

Table 2　Comparison results for the 30-size instances of different constraint handling strategies

		30–001	30–002	30–003	30–004	30–005	30–006	30–007	30–008	30–009	30–010
Bt/h	MOCH	**268.3**	**209.6**	**247.4**	**233.3**	**282.3**	**235.5**	**243.0**	**243.6**	**296.6**	**267.2**
	RM	294.8	224.2	260.9	239.2	293.7	248.7	247.0	256.4	329.6	282.0
	PF	323.5	230.3	267.5	245.0	325.6	298.5	275.4	294.8	356.4	328.0
	SF	350.2	247.1	262.8	266.2	376.8	297.9	297.1	293.5	374.0	368.6
	COM	300.3	228.3	278.4	238.2	345.0	284.4	312.9	291.1	336.5	360.4
Ag/h	MOCH	**277.2**	**211.5**	**249.0**	**237.5**	**295.3**	**238.0**	**246.5**	**245.8**	**310.6**	**275.4**
	RM	324.4	231.4	269.9	252.0	329.9	271.4	261.4	268.2	351.6	314.6
	PF	392.7	255.8	321.3	310.7	389.4	352.2	344.7	372.5	429.1	395.2
	SF	440.1	277.9	326.7	361.1	437.1	347.7	361.3	411.6	460.6	470.8
	COM	393.6	267.9	330.6	310.8	442.4	331.4	364.2	365.8	437.3	431.6
SD	MOCH	**12.8**	**1.8**	**1.2**	**3.4**	**8.9**	**3.7**	**2.6**	**2.4**	**7.5**	**5.9**
	RM	17.7	4.6	7.4	8.0	17.2	11.4	8.1	9.6	16.3	17.0
	PF	40.8	23.6	33.2	43.2	30.9	42.8	49.9	52.6	54.0	46.7
	SF	57.3	26.0	37.1	81.0	62.7	34.6	52.4	112.0	54.2	129.9
	COM	66.8	26.9	37.4	52.5	51.4	33.6	37.1	47.9	72.2	60.7
Te/s	MOCH	29.2	29.2	29.1	29.2	29.1	28.8	29.6	28.9	28.8	29.0
	RM	64.0	62.7	62.7	62.4	62.7	63.1	63.7	61.1	62.0	61.3
	PF	21.2	21.3	21.0	21.2	21.1	20.9	21.2	21.5	21.2	21.1
	SF	20.5	20.4	20.6	20.4	20.3	20.5	20.2	20.1	20.4	20.5
	COM	24.8	24.9	24.7	25.2	24.9	25.0	25.2	25.3	24.8	24.6
Nfs/次	PF	19	20	20	20	19	20	20	20	19	20
	SF	20	20	17	20	19	20	20	20	20	20
	COM	19	20	20	20	20	20	20	20	20	20

为进一步验证不同约束处理策略结果的差异性, 采用Wilcoxon秩和检验 (wilcoxon signed-rank test, WRS)对采用不同策略得到的20次结果进行统计分析. 在5%的显著水平下, 该方法返回原假设(两组独立数据来自具有同一分布的总体)的双边检验的概率 p 值, 若 $p < 5\%$ 则表示在该水平下拒绝原假设

($h=1$), 反之则接受原假设($h=0$). 图5给出了40个算例中MOCH与RM, PF, SF和COM间WRS检验返回的p值, 分别记作WRS1, WRS2, WRS3和WRS4, 可以看到几乎所有算例中p都远小于0.05, 这表明在较高的置信度水平下MOCH与其他策略的结果存在显著差异.

表 3 40规模算例中不同策略的结果对比

Table 3 Comparison results for the 40-size instances of different constraint handling strategies

		40–001	40–002	40–003	40–004	40–005	40–006	40–007	40–008	40–009	40–010
Bt/h	MOCH	**388.0**	**349.1**	**395.2**	**454.0**	**340.3**	**380.6**	**378.5**	**439.4**	**349.2**	**372.6**
	RM	453.9	361.9	426.2	476.6	354.4	432.9	411.4	494.5	401.7	431.2
	PF	458.8	418.2	526.8	548.8	432.2	562.5	513.1	630.3	496.6	446.2
	SF	528.3	549.9	565.5	594.7	–	634.4	679.4	805.5	548.9	644.2
	COM	470.4	391.4	528.8	495.1	450.3	448.6	480.3	617.2	441.9	474.5
Ag/h	MOCH	**418.7**	**356.9**	**437.9**	**488.2**	393.7	**404.9**	**392.0**	**494.3**	**377.2**	**406.3**
	RM	500.0	399.2	507.6	506.2	**393.2**	485.8	452.1	543.5	449.4	505.5
	PF	661.7	524.6	701.4	644.3	549.2	697.6	636.3	810.2	574.9	666.2
	SF	701.8	658.2	776.8	777.3	–	781.6	751.3	1004.5	745.3	843.1
	COM	635.6	562.2	682.8	663.2	597.4	681.3	618.2	779.9	652.4	701.8
SD	MOCH	**23.1**	**7.7**	**36.9**	26.5	69.8	**16.9**	**13.4**	34.3	**15.9**	**22.4**
	RM	30.2	26.3	56.3	**17.9**	**25.3**	33.7	28.9	**32.6**	26.6	43.5
	PF	100.0	37.1	111.9	73.2	48.1	95.3	64.3	96.6	52.3	98.5
	SF	70.2	165.7	128.5	258.2	–	135.2	68.0	137.9	159.1	172.8
	COM	90.6	101.0	85.9	108.7	108.2	121.4	95.0	113.6	125.0	117.5
Te/s	MOCH	64.2	64.3	64.2	65.1	64.3	64.4	63.6	64.3	63.5	64.2
	RM	180.4	179.8	185.3	188.5	187.0	181.9	180.9	188.6	177.8	189.0
	PF	49.2	48.7	48.6	48.9	47.1	48.5	48.4	48.7	48.4	48.8
	SF	47.3	47.7	48.9	48.9	49.1	47.7	49.4	48.8	48.1	48.6
	COM	60.1	60.7	60.8	58.9	61.4	60.8	59.1	59.9	59.6	59.5
Nfs/次	PF	20	18	20	15	11	16	13	19	19	14
	SF	19	3	18	2	0	10	7	13	16	10
	COM	20	20	20	20	20	20	19	20	20	19

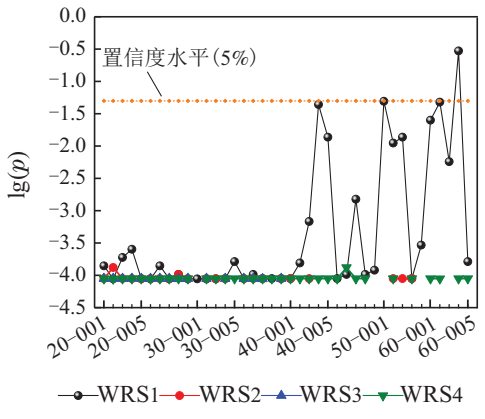


图 5 不同约束处理策略计算结果的WRS检验

Fig. 5 The WRS test results of different methods

综合来看, MOCH求解质量最好, 这是因为非支配排序保留了大量目标值极具优势的不可行解, 而这些不可行解的基因包含着有价值的信息, 进化过程中可以充分利用从而产生更好的可行解. SF的思想与其恰恰相反, 进化过程首先淘汰约束违反度较大的不可行解, 实验结果表明这种策略在小规模算例中能较快求得可行解, 但容易陷入局部最优, 在大规模算例中, 难以求得可行解, 这也验证了MOCH的正确性和有效性. PF和COM的求解质量较SF好, 并且这种优势随着算例规模增大而增大. 这是因为算例规模增大必然导致约束冲突增加, 生成可行解的概率降低, 所以SF很难得到可行解, 而PF和COM以适应度为依据选择下一代的个体, 在这个过程中部分目标函数值较好的不可行解被保留参与进化, 所以求得可行解的质量更好, 也相对稳定. 本文提出的RM求解效果仅次于MOCH, 这是源于其精巧的调整过程, 但是随着问题规模增加, 调整复杂度越来越高, 耗费时间也越来越长. COM也对基因进行了部分调整, 所以不能求得可行解的概率比PF和SF低得多, 但求解时长也随着算例规模增大而相应增加.

4.2 不同约束处理策略收敛性能分析

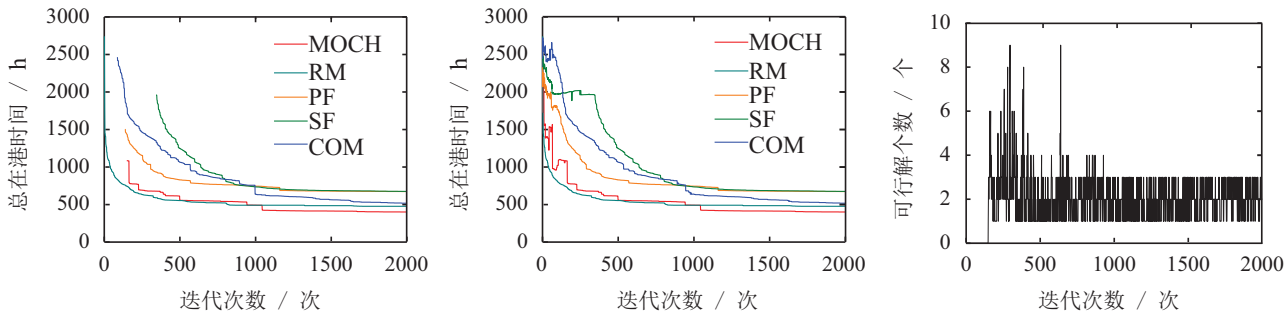
为了探究约束处理策略对进化过程的影响, 本文记录5种策略在40–001算例第1次求解过程中全局最

优的可行解, 作图6(a)收敛曲线, 并记录每次迭代当前种群中最好的解, 作图6(b). RM求得的个体都是可行解, 在迭代过程中最优解的目标函数值不断减小, 且当前种群最好解和全局最优解是一致的, 所以该策略在图6(a)和图6(b)中曲线完全相同. 从图6(a)可以看到MOCH在第149次迭代时出现可行解, SF在第343次迭代出现可行解, 在这两种策略中, 一旦出现了可行解, 就会一直保留这个解, 直到出现更好的可行解, 因此这两种策略在图6(a)和图6(b)中找到第1个可行解后的曲线是完全一致的. PF和COM的第1个可行解分别出现在第138次迭代和第87次迭代. 从图6(b)可以看出, MOCH的目标函数值震荡幅度最大, 找到第1个可行解之后仍有多次目标值的大幅改进, 说明该策略寻优能力强, 跳出了局部极值. 与之相对的SF目标值震荡幅度最小, 曲线相对光滑. PF和COM在出现可行解之前震荡幅度较大, 出现可行解后仍然有小幅的震荡, 这意味着出现了适应度更高的不可行解, 但振幅逐渐减小, 曲线趋向光滑.

表 4　50, 60规模算例中不同策略的结果对比

Table 4　Comparison results for the 50-size and 60-size instances of different constraint handling strategies

		50–001	50–002	50–003	50–004	50–005	60–001	60–002	60–003	60–004	60–005
Bt/h	MOCH	**422.8**	**582.2**	**835.7**	**515.6**	**563.2**	**1145.0**	**789.0**	**863.4**	1071.8	**592.5**
	RM	518.9	643.8	858.4	603.2	628.6	1187.0	826.8	906.0	**1003.9**	791.6
	PF	797.8	786.1	1163.3	678.1	794.1	1830.1	1245.2	1382.8	1656.7	1292.5
	SF	739.6	897.1	1290.2	970.0	932.7	–	–	–	–	1595.6
	COM	603.4	785.7	1069.7	792.0	749.6	1530.3	1068.7	1226.7	1417.1	1042.0
Ag/h	MOCH	**514.7**	**668.9**	**959.8**	**567.7**	**638.2**	**1335.4**	**926.4**	**981.7**	1192.8	**757.3**
	RM	556.6	716.5	1052.1	695.0	695.4	1434.8	943.2	1063.5	**1162.7**	948.5
	PF	921.1	1062.0	1380.9	998.8	901.8	1926.6	1375.9	1654.0	1955.3	1574.6
	SF	1209.5	1301.4	1780.8	1292.9	932.7	–	–	–	–	1595.6
	COM	805.4	911.4	1343.4	972.4	883.2	1881.1	1428.8	1560.7	1750.9	1461.6
SD	MOCH	57.8	49.4	**98.2**	**31.6**	**32.7**	**130.9**	91.3	**75.3**	96.0	**99.5**
	RM	**40.5**	**35.1**	98.6	50.7	39.8	141.4	**59.4**	93.0	**86.3**	102.6
	PF	113.0	134.5	125.0	133.2	63.7	130.6	138.0	177.9	422.2	185.7
	SF	319.7	270.2	465.7	332.3	–	–	–	–	–	–
	COM	133.3	100.0	125.3	126.4	103.8	220.0	222.8	196.6	188.2	234.0
Te/s	MOCH	92.0	94.7	96.0	93.7	91.4	182.1	179.2	180.2	181.1	180.4
	RM	421.8	452.9	443.6	431.6	438.3	1037.2	995.8	953.9	967.5	937.5
	PF	99.3	102.2	102.1	101.3	99.3	196.3	196.4	196.3	195.9	195.9
	SF	98.6	100.6	100.3	100.4	97.7	187.7	187.4	188.8	187.9	187.3
	COM	123.7	125.2	119.0	118.0	119.0	248.2	246.4	251.2	248.1	246.8
Nfs/次	PF	19	20	20	20	12	4	3	12	2	13
	SF	14	16	18	7	1	0	0	0	0	1
	COM	18	20	19	20	19	20	20	19	20	20



(a) 全局最优可行解收敛曲线　(b) 种群中最好解变化曲线　(c) 种群中可行解个数

图 6　不同约束处理策略求解BACAP的算法收敛过程

Fig. 6　The convergence process of different methods

为进一步阐明MOCH的特殊性, 本文记录每次迭代其种群中可行解个数, 作图6(c), 可以看出每一代种群中可行解都是较少的. 找到可行解后的算法初期, 可行解个数相对较多, 说明算法搜索到的解在解空间上分布较分散, Pareto分层较多, 质量较差的可行解也被保留进入下一次迭代. 自1000次迭代之后, 种群内可行解的个数在{1, 2, 3}间波动, 表明算法进入收敛阶段, 种群中个体集中分布在Pareto前沿附近, Pareto分层少, 所以只有极少数可行解被保留.

4.3 BACAP方案分析–以算例40–004为例

图7为基于MOCH的算法求得40–004算例的近似最优解示意图. 分析算例数据可知, 船舶到港时间具有较强的不平衡性: 在0∼30 h, 60∼90 h内船舶到港稀疏, 在30∼60 h, 90∼160 h内船舶到港集中. 这种不平衡性在分配方案上也得到了体现: 港口较为空闲时, 船舶到港无需等待即可为其安排靠泊, 此时泊位、岸桥资源较为空闲; 港口繁忙时段, 船舶调度计划紧凑, 岸桥最大利用率100%, 最小利用率80%, 泊位资源非常紧张. 在规划期内的40艘船舶中, 有33艘船舶等待时间在5 h以内, 其中16艘船不需等待. 除了第28艘船等待时间超过10 h, 其余6艘船等待时间都在10 h以内, 所有船舶均在规划期内完成了装卸任务.

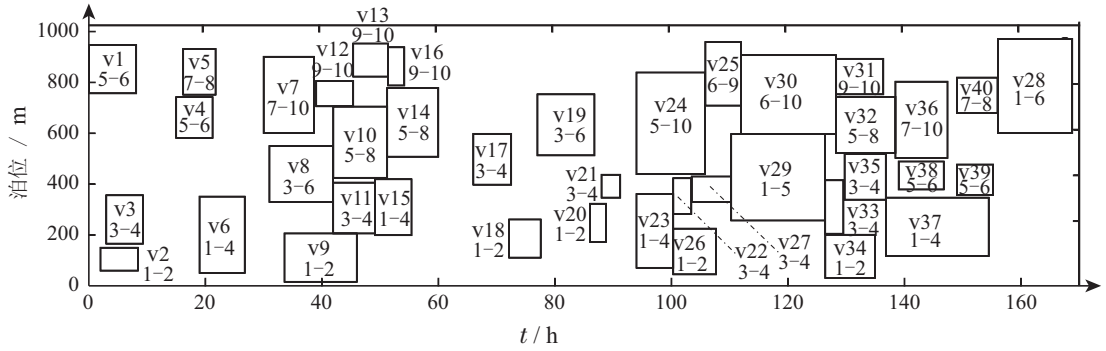


图 7 算例40–004的近似最优分配方案

Fig. 7 The near-optimal scheme of instance 40–004

5 结论

本文考虑岸桥不可交叉及其具体分配建立了BACAP的单目标MINLP模型, 并针对问题特点, 设计了调整、惩罚函数、可行解优先和综合约束处理策略并结合遗传算法对其进行求解. 同时, 基于MOCH将原带复杂约束的单目标优化模型转化只有简单约束的双目标优化模型, 并用NSGA-II对其进行求解. 针对5组不同规模算例的实验结果表明, MOCH相较于其它约束处理策略可以在较短时间内求得质量更好的解. 算例规模越大, MOCH的优势越明显, 尤其在大规模算例中, 其求解质量和稳定性远高于其他策略, 求解时间也更短.

需要指出的是, 本文将MOCH应用于BACAP的两部分不同约束时,并没有考虑二者间相对数量关系, 这虽然不会影响该策略的可行性,但不同的总约束违反度刻画方式会在算法迭代过程中对个体选择造成不同影响. 因此, 针对MOCH应用于复杂约束时, 如何平衡不同约束的违反度取值范围及其对算法迭代的影响机理需要进行更深入的研究. 另外, 针对港口作业需求的不平衡性, 今后的研究中可以对船舶航速进行优化, 避免船舶集中到港, 可以大大降低分配难度, 同时减少船舶等待时间.

参考文献:

[1] IMAI A, NAGAIWA K, TAT C. W. Efficient planning of berth allocation for container terminals in asia. *Journal of Advanced Transportation*, 1997, 31(1): 75 – 94.

[2] IMAI A, NISHIMURA E, PAPADIMITRIOU S. The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, 2001, 35(4): 401 – 417.

[3] NISHIMURA E, IMAI A, PAPADIMITRIOU S. Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research*, 2001, 131(2): 282 – 292.

[4] IMAI A, NISHIMURA E, PAPADIMITRIOU S. Berth allocation with service priority. *Transportation Research Part B: Methodological*, 2003, 37(5): 437 – 457.

[5] IMAI A, SUN X, NISHIMURA E, et al. Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B*, 2005, 39(3): 199 – 221.

[6] LIM A. The berth planning problem. *Operations Research Letters*, 1998, 22(2/ 3): 105 – 110.

[7] TONG J, NACHTMANN H. A tabu search approach to the cargo prioritisation and terminal allocation problem. *International Journal of Shipping and Transport Logistics*, 2020, 12(3): 147 – 173.

[8] BACALHAU E T, CASACIO L, DE AZEVEDO A T. New hybrid genetic algorithms to solve dynamic berth allocation problem. *Expert Systems with Applications*, 2021, 167: 114198.

[9] PARK H J, CHO S W, LEE C. Particle swarm optimization algorithm with time buffer insertion for robust berth scheduling. *Computers & Industrial Engineering*, 2021, 160: 107585.

[10] BIERWIRTH C, MEISEL F. A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 2010, 202(3): 615 – 627.

[11] BIERWIRTH C, MEISEL F. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 2015, 244(3): 675 – 689.

[12] PARK Y M, KIM K H. A scheduling method for berth and quay cranes. *OR Spectrum*, 2003, 25(1): 1 – 23.

[13] MEISEL F, BIERWIRTH C. Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research. Part E: Logistics and Transportation Review*, 2009, 45(1): 196 – 209.

[14] WANG T, DU Y, FANG D, et al. Berth allocation and quay crane assignment for the trade-off between service efficiency and operating cost considering carbon emission taxation. *Transportation Science*, 2020, 54(5): 1307 – 1331.

[15] SONG Yunting, WANG Nuo, WU Nuan. Optimization of berth and quay crane collaborative scheduling considering both the cost and time guarantee rate. *Operations Research and Management Science*, 2020, 29(4): 130 – 137.
(宋云婷, 王诺, 吴暖. 兼顾成本与时间保证率的泊位及岸桥协同调度优化. 运筹与管理, 2020, 29(4): 130 – 137.)

[16] WU Di, WANG Nuo, LIN Wanni, et al. Alternate evolution algorithm based on plant growth simulation for berth-quay crane joint allocation model. *Journal of Traffic and Transportation Engineering*, 2018, 18(3): 199 – 209.
(吴迪, 王诺, 林婉妮, 等. 泊位–岸桥联合分配模型的模拟植物生长交替进化算法. 交通运输工程学报, 2018, 18(3): 199 – 209.)

[17] CORRECHER J F, ALVAREZ-VALDES R, TAMARIT J M. New exact methods for the time-invariant berth allocation and quay crane assignment problem. *European Journal of Operational Research*, 2019, 275(1): 80 – 92.

[18] XIANG X, LIU C. An almost robust optimization model for integrated berth allocation and quay crane assignment problem. *Omega*, 2021, 104: 102455.

[19] JI B, YUAN X, YUAN Y. Modified NSGA-II for solving continuous berth allocation problem: Using multi-objective constraint-handling strategy. *IEEE Transactions on Cybernetics*, 2017, 47(9): 2885 – 2895.

[20] HAMZA N, SARKER R, ESSAM D. Evolutionary search from the interior of feasible space. *Symposium Series on Computational Intelligence (SSCI)*. Canberra, Australia: IEEE, 2020: 353 – 359.

[21] GARZAFABRE M, RODRIGUEZTELLO E, PULIDOO G T. Constraint-handling through multi-objective optimization: The hydrophobic-polar model for protein structure prediction. *Computers & Operations Research*, 2015, 53: 128 – 153.

[22] SEGURA C, COELLO C A C, MIRANDA G, et al. Using multi-objective evolutionary algorithms for single-objective constrained and unconstrained optimization. *Annals of Operations Research*, 2016, 240(1): 217 – 250.

[23] TÜRKOĞULLARI Y B, TAŞKIN Z C, ARAS N, et al. Optimal berth allocation and time-invariant quay crane assignment in container terminals. *European Journal of Operational Research*, 2014, 235(1): 88 – 101.

[24] SHANG L, SHANG Y, HU L, et al. Performance of genetic algorithms with different selection operators for solving short-term optimized reservoir scheduling problem. *Soft Computing*, 2020, 24(9): 6771 – 6785.

作者简介:

黄　涵　硕士研究生, 目前研究方向为优化理论、算法及其在港口资源调度领域的应用, Email: 2521807984@qq.com;

季　彬　教授, 目前研究方向为优化理论、算法及应用、物流配送优化理论与应用、航运网络优化和资源调度等, Email: cumtjibin@126.com.