大连理工大学

硕士学位论文

基于知识与仿真的集装箱堆场作业计划研究

姓名：沈剑峰

申请学位级别：硕士

专业：管理科学与工程

指导教师：金淳

20061201

摘　要

随着经济全球化、贸易自由化的深入发展，我国逐步融入世界经济大潮之中，对外贸易迅速发展。大规模的贸易产生了大规模的物流，港口是国际物流网络的枢纽结点，集装箱运输是现代航运的主要方式，建设现代化的集装箱港口，是我国物流基础设施建设的重要组成部分。

集装箱堆场作业主要有提箱、进箱、装船、卸船四种，堆场作业计划的内容包括空间资源和设备资源的配置。本文按不同的作业类型，对国内外学者就空间资源和设备资源的配置两个方面的研究现状进行了分析，将提箱作业和进箱作业（箱位分配）确定为本文的研究重点。

针对集装箱堆场箱位分配问题中规则因素较强的特点，建立基于知识的箱位分配计划方法。首先，描述了该问题中知识构成、表示及获取方法；然后，提出了基于知识的箱位分配模型，包括分配区域划分、作业模式及规则匹配、最优箱位选择三个部分。

针对集装箱堆场的提箱作业优化问题，提出了一个以作业成本最小化为目标的多阶段决策优化模型，该模型由 2 部分组成：倒箱优化模型和最短路搜索模型。同时设计了一个内外两层的嵌套遗传算法，内层算法负责搜索最短路径，外层算法优化倒箱作业过程。遗传算法中引入了父代参与选择和最优个体保留两种改进策略，并对 GA 的编码解码设计和适应度函数设计作了重点讨论。

在对箱位分配作业和提箱作业研究的基础上，设计并实现了集装箱堆场大门作业计划系统。该系统嵌入了箱位分配作业计划和提箱作业计划算法，建立了堆场数据库，实现了两种作业计划的连续、实时操作。

最后，针对系统测试存在大量测试数据难以获取，系统长期连续运行性能难以评价的困难，提出了一个基于仿真的信息系统性能测试框架，利用该框架对集装箱堆场大门作业计划系统进行性能测试。该方法将箱位分配作业计划算法和提箱作业计划算法融入仿真系统，测试数据由仿真系统产生，解决了测试数据的来源问题，并通过仿真统计分析评价系统的运行性能，测试结果表明该方法是可行且有效的。

关键词：仿真；知识；堆场作业；遗传算法；性能测试

Research on Container Yards Operation Schedule based on Knowledge Engineering and System Simulation

Abstract

With the development of economic globalization and the trend of trade freedom, China is on the way to the spring tide of world economy step by step and its international trade grows quickly. Large-scale logistics comes along large-scale trade; port is the hinge node of the international logistics net, and container transportation is one of the most important mode of modern ship. Therefore, it is an important part of our country's Logistics Infrastructure construction to build modern container terminals.

Operation in container yard includes delivery, receiving, load and unload; yard operation schedule revolves deployment of resources such as yard space and facility. In this paper, analysis of published results of research on yard operation schedule is reported; both delivery and receiving (storage space allocation) are thought to be the main research point of this paper.

For storage space allocation on container yard, there are many kinds of rules in planning a container's location. For this characteristic, a knowledge-based storage space allocation method is proposed. Firstly, composition, representation and acquirement of knowledge is described; then, a knowledge-based model for storage space allocation is established. This model is comprised of three parts: partition of allocation area, operation mode and rule match, the best slot selection.

For the optimization problem on container pick-up operation scheduling, a multi-stage mathematical programming model is established to minimize the total operation cost. This model is comprised of two parts: rehandling operation scheduling and the shortest path search. A nested genetic algorithm with two layers is proposed, where the inner layer algorithm is embedded in the outer one. The outer algorithm is responsible for optimizing rehandling operation scheduling, and the inner is for searching the shortest path of rehandling operation. Two reformative operations are introduced, including the parent population involving selection and optimal individual maintaining. The design of fitness function and the scheme of encoding and decoding of GA are discussed significantly.

Based on the analysis of yard storage space allocation and container pick-up operation, we design and implement a Container Yard Operation Schedule System, in which the algorithms of yard storage space allocation and pick-up operation scheduling are embedded.

A yard database is established to ensure the operation schedule can be carried out continuously and real-timely.

Finally, for the system testing, it is found that it is difficult to collect plenty of testing data and evaluate the system performance during a long period of execution, so a discrete event simulation-based testing method is proposed to overcome these difficulties. In the infrastructure of the proposed method, the information system to be tested is combined with the simulation system, the testing data is generated by simulation and the testing efficiency is greatly improved; the performance of the information system also can be evaluated by statistical analysis. The results show that the proposed method has qualified validity.

Key Words: Simulation; Knowledge; Yard Operation; Genetic Algorithm; Performance Testing

独创性说明

作者郑重声明：本硕士学位论文是我个人在导师指导下进行的研究工作及取得研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写的研究成果，也不包含为获得大连理工大学或者其他单位的学位或证书所使用过的材料。与我一同工作的同志对本研究所做的贡献均已在论文中做了明确的说明并表示了谢意。

作者签名：沈剑峰 日期：2007年1月5日

大连理工大学学位论文版权使用授权书

本学位论文作者及指导教师完全了解“大连理工大学硕士、博士学位论文版权使用规定”，同意大连理工大学保留并向国家有关部门或机构送交学位论文的复印件和电子版，允许论文被查阅和借阅。本人授权大连理工大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，也可采用影印、缩印或扫描等复制手段保存和汇编学位论文。

作者签名：沈剑峰

导师签名：

2007 年 1 月 5 日

1 绪论

1.1 问题的提出

随着经济全球化、贸易自由化的深入发展，我国逐步融入世界经济大潮之中，特别是加入 WTO 之后，进一步加快了对外贸易的发展。据统计，我国进出口总额由 1978 年的 206 亿美元，增长到 2004 年的 1.15 万亿美元，年均增长 16.7%[1]；2005 年，我国进出口总额 7620 亿美元，同比增长 23.2%[2]，2006 年上半年，全国进出口总额 7957 亿美元，同比增长 23.4%[3]。

大规模的贸易活动产生了大规模的物流，港口在国际物流网络中扮演着枢纽结点的重要角色。以 2004 年为例，我国货物进出口总额 1.15 万亿美元，其中进出口货物运输的 90%以上是通过港口和航运来实现的[4]。集装箱运输这种运输方式因为其安全，具有可提高装卸效率、减轻劳动强度、加快车船周转、节省包装和运输费用、有利于标准化和组织多式联运等优点[5]，从而成为现代航运的主要方式，集装箱吞吐量也成为港口规模的重要指标。建设现代化的集装箱港口，是我国物流基础设施建设的重要组成部分。

1.1.1 集装箱堆场作业计划的内容

集装箱港口应该具备的必要设施有：泊位、码头前沿、集装箱堆场、港口大门、货运站、控制室、行政楼、维修车间等，其中主要的是泊位、集装箱堆场和港口大门。在港口这个系统的三个主要要素中，泊位和港口大门处于系统的边界，它们是港口系统和外界交换物质和能量的接口，其中泊位联系着水路一侧，大门联系着内陆腹地；堆场是水陆集装箱的集散中转地，是港口各种资源和活动交汇的枢纽，担负着集装箱的交接、堆存、货运等功能。

集装箱港口的主要业务可分为进口和出口两种，如图 1.1 所示。进口业务的作业流程简单地可以表示为：进口卸船→堆场作业→外卡提箱；出口业务的作业流程则可以表示为：外卡进箱→堆场作业→出口装船[6]。相应地，与堆场有关的作业有四种，即进口卸船、出口装船、外卡提箱、外卡进箱。前两种发生在堆场与泊位之间，后两种发生在堆场与大门之间，这四种作业均属于随机过程，涉及堆场空间、设备等多种因素，从而增加了合理安排作业计划的难度。

集装箱港口是资本密集型的经济实体，要使其作业物流顺利而经济地运行，物流资源的合理有效配置是必要保障；堆场是集装箱港口的重要组成部分，堆场作业资源的配置关系到港口的运作效率及其竞争力。堆场作业计划就是合理配置港口资源，安排堆场的各种作业，达到节约作业成本、提高作业效率的目的。

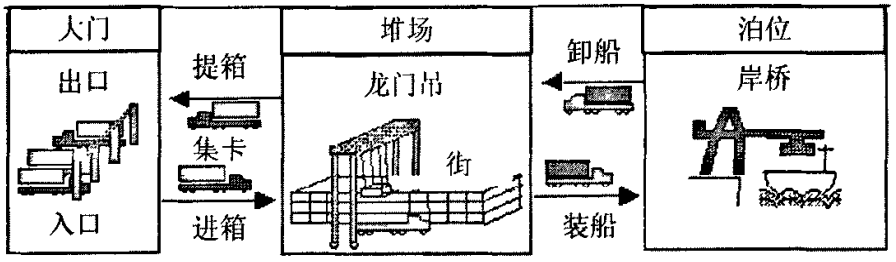


图 1.1 集装箱作业

Fig. 1.1 Operation of Container terminal

从内容上讲，堆场作业计划是对堆场作业资源的合理配置，堆场作业资源可分为堆场空间资源和堆场设备资源两种，相应地，堆场作业计划也可分为堆场空间资源计划和堆场设备作业计划[7]。

堆场空间资源主要是指集装箱堆场箱区的空间大小，空间资源计划主要是指作业堆场箱区应分配的集装箱堆存数量和箱位安排问题，合理的堆存数量和箱位安排不仅能够提高堆场的利用率，而且能够提高堆场机械的作业效率、减少设备之间的相互等待时间，缩短船舶在港停留时间。

堆场设备资源是指进行装卸、搬运、堆垛作业的专用机械，主要有叉车、正面吊和龙门吊。堆场设备作业计划主要是指堆场设备的配置数量以及堆场设备在箱区间的移动路径选择，合理的堆场设备的配置数量以及移动路径可以提高设备工作效率和利用率。

因此，以计划内容和港口作业为两个维度，堆场作业计划的矩阵分析如图 1.2 所示：

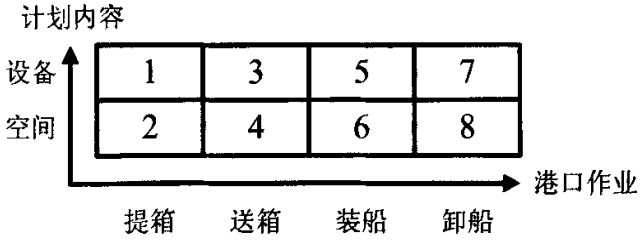


图 1.2 堆场作业计划的矩阵分析

Fig. 1.2 Matrix analysis of yard operation schedule

其中“1”为提箱作业的设备资源配置，“2”为提箱作业的空间资源配置；“3”为进箱作业的设备资源配置，“4”为进箱作业的空间资源配置；“5”

为装船作业的设备资源配置，“6”为装船作业的空间资源配置；“7”为卸船作业的设备资源配置，“8”为卸船作业的空间资源配置。

1.1.2 提箱和箱位分配作业计划

港口大门一侧的外卡提箱和外卡进箱作业计划研究对堆场作业有着重要影响，是堆场作业计划的重要组成部分，其中提箱针对的是进口箱，箱位分配针对的是出口箱。

研究提箱作业计划问题的必要性在于问题本身的复杂性和问题得到解决的重要意义。由于客户需要提取的集装箱往往不能直接提取，因而提箱作业过程中会产生许多倒箱操作，不合理的倒箱会降低堆场作业效率，增加作业成本，甚至于导致集装箱堆放的混乱，堆场内交通阻塞等现象。提箱作业计划就是，提供为提取堆场中某个集装箱而引起 的全部倒箱作业步骤的方案，它包括两个方面：

(1) 倒箱的先后顺序；

(2) 每个待倒箱放置的目标位置。

提箱作业计划是一个复杂的问题，涉及很多因素，如堆场的堆存状态、设备的作业能力、作业成本等。因此，得到一种最优的提箱作业计划（倒箱方案）需要权衡多个方面，达到一个综合的目标。

箱位分配作业计划是港口大门作业的另一方面，是指依照堆场管理的各种原则，为进场集装箱分配一个最优放置箱位，使得随机到达的集装箱的堆垛符合堆码要求，方便装船作业，降低堆场作业的成本、提高效率，减少船舶在港停泊时间。

该问题的难点在于：从时间上看，进场箱的到达时间是随机的，重量分布也不可预知，所以给集装箱分配箱位的过程中信息是不完备的。从空间上看，堆场面积有限，给到达箱预留缓冲区空间，累积到一定数量后重新优化堆垛设计不可行，况且又增加了作业成本。因此需要在堆场空间利用和后续装船作业的期望时间之间做出权衡。

1.2 国内外研究现状

1.2.1 堆场空间资源配置的研究现状

堆场空间资源的分配主要因轮船到港时的卸船作业和外卡进箱作业而发生。

(1) 卸船作业的堆场空间资源分配

卸船作业是船舶到达事件引起的，此时集装箱批量到达，堆场空间资源分配问题往往和堆场设备资源的调度交织在一起。优化的目标大多集中在高的堆场利用率、少的卸船（船舶在港）时间和高的机械作业效率。解决该问题时，需要考虑以下几个方面：

① 船舶达到规律和需要卸载的集装箱的相关特征；

② 堆场空间的制约；

③ 外卡随机提取进口箱时的期望倒箱数。

在针对卸船作业的堆场空间资源分配方面，De Castinhno 和 Dagano[8]研究了两种策略下的进口集装箱的堆存问题，一种策略是按照集装箱大小尺寸分类堆放，另一种策略是按照到达时间先后依次堆放。Kim[9]研究了定期班轮和不定期船次的进口箱堆存空间分配问题，按照进口船舶连续均匀到达、周期性到达、随机到达三种情况，分别建立数学模型，模型的目标是外卡随机提取进口箱时的期望倒箱数最小，模型优化的变量是堆场的集装箱堆放高度。但他假设只允许在同贝内倒箱，仅仅考虑只有场桥一种设备作业的情形。Peter[10]讨论了集装箱堆场的最优堆存方案，建立了一个混合整数线性规划模型，目标函数是使船舶停港时间最小化，并用遗传算法求解。Ebru[11]研究了多跨运车约束的作业计划问题，将进口堆场空间配置问题同岸桥调度问题一起考虑，他把这一问题分成三个子问题：① 为每个卸船的集装箱分配堆场空间；② 为运输集装箱而分配拖运设备；③ 调度岸桥的装卸作业；优化的目标是对船舶（船队）的服务时间最短。

(2) 进箱作业的堆场空间资源分配

外卡进箱作业的发生是一系列离散随机事件，时间跨度是预定班次班轮到港前的载入期，该作业的最终目的是为集装箱出口装船作准备，优化的目标包括高的堆场利用率、少的预期后续装船时间、低的后续装船作业时的翻倒率，需要考虑的是堆场作业原则。

在针对进箱作业的堆场空间资源分配方面，Sculli 和 Hui[12]通过仿真的方法研究了堆场堆高、堆场利用率和翻倒率之间的关系。Tabel-Ibrahimi[13]等也采用仿真的方法讨论了出口箱区不同安排原则下堆高、堆场利用率和翻箱率之间的关系。

Kim 和 Bae[14]针对集装箱出口箱装船前为加快装船作业速度，减少装船时间，往往需要重新调整堆垛的问题，提出一种变当前贝堆垛为期望贝堆垛的方法，实现倒箱数量最少、设备移动距离最短的目标。该方法方法将问题分为 3 个子问题：贝匹配、移动计划、任务排序，对每个子问题建立一个数学模型。Kim[15]等考虑集装箱的重量属性，堆场贝的堆存状态和已有集装箱的重量分布，提出动态规划模型，采用了加权的方法计算期望翻倒次数，最小化装船作业时的期望倒箱作业。在此基础上，提出从最优解集中生成决策树的方法，支持实时决策。Kim 和 Park[16]讨论了出口集装箱的动态分配方法，建立了混合整数规划模型。基于集装箱在港停留时间和梯度下降优化，分别提出两个启发式算法，为到达堆场大门的出口集装箱分配一个箱位，使堆场具有高的空间利用率和高的装船作业效率。Kim 和 Lee[17]采用约束条件技术，在满足堆场作业设备、集卡的作业效率约束的前提下，解决出口箱的箱位分配问题，达到提高后续装船作业的效率的目标。

(3) 堆场空间资源分配的其他相关研究

由于卸船作业和进箱作业都会产生对堆场空间的占用，有的学者对以上两个方面作了综合研究，E.mcdowill[18]等讨论了集装箱港口的作业流程，在考虑各作业流程所发生成本的基础上，建立了一个成本模型来解决堆场堆存问题。这种成本模型虽然可以降低作业总成本，但是没有考虑港口作业设备效率的问题。Kim[19]讨论了由场桥执行提箱作业时，合理确定堆垛高度和宽度的优化算法，分别达到提取一个箱的期望倒箱数量最小，提取贝中所有箱的期望倒箱数量最小的目标。Ping cheng[20]等讨论了堆场集装箱空间的分配问题，提出用多种算法解决该问题，包括禁忌搜索方法和“squeaky wheel”优化方法，提出了一种混合方法来求解。

Chunqian Zhang[21]讨论了4种箱（进口卸船、堆场待提、大门入场、出口装船）的堆场空间箱位分配问题，分析了这四种作业对堆场空间发生的变化以及相互之间的影响。为达到减少船舶在港停留时间、增加堆场设备利用率的目标，作者采用了滚动计划法，在每个计划期，问题分解为两个层次，分别建立了数学模型。第一层规划计划期内每个时间段中各个街的堆放集装箱总的数量；第二层估计每个计划期内每艘船将要装卸集装箱的数量，减少装卸时运输集装箱的距离。

近年来，国内也有部分学者对堆存问题进行了研究。徐剑华[22]提出用择箱指数方法寻求集装箱最佳堆存高度和最佳布置，以使堆场的面积利用率和堆场的取箱效率获得综合优化的结果。郝聚民、纪卓尚[23]等在图搜索技术和模式识别理论的基础上建立了随机条件下的混合作业堆场贝优化模型。杨淑琴、张运杰[24]等根据出口集装箱先到港、尽量轻箱在下重箱在上的原则，采用启发式方法合理安排出口集装箱堆场箱位安排，以达到装船作业时堆场机械的翻箱率最低。Kim[25][26][27]研究了多种数学算法和费用模型以寻找在不同标准的堆放策略。

1.2.2 堆场设备资源配置的研究现状

(1) 装船/卸船作业的堆场设备资源分配

堆场设备资源计划在提箱、进箱、装船、卸船四种作业中均涉及到，主要在于装船和卸船作业，因为这两种作业发生时，时间紧、搬运工作量大，涉及的因素多。研究该问题，主流的方法还是整数规划、动态规划、分枝定界法等运筹学方法，优化目标集中在设备路径、作业成本、作业时间的最小化。

Kim[28]研究了装船作业时，跨车的路径优化问题，以跨车在堆场中总的运行路径最小、作业时间最少为目标，将问题归结为整数规划问题，提出相应算法，确定跨车在各个贝准备提取的集装箱数量和跨车提取集装箱的贝的访问顺序。K.H.Kim 和 J.W.Bae[29]采用混合整数规划模型研究将集装箱分配给自动牵引车(AGV)的调度问题。Kozan 和

Preston[30]为加快对集装箱处理作业，减少集装箱转移时间和轮船在港时间，建立多阶段动态规划模型，处理进口箱的堆放、出口箱的装船，设备的调度，并用遗传算法求解该模型。Michael[31]等分析了船舶稳定性和集装箱的重量关系，针对同类箱的摆放问题提出了相应解决方法。Ng 和 Mak[32]研究了在某一给定的装船/卸船作业中，堆场起重机的作业调度问题，采用分枝定界法，最小化工作等待时间。

(2)龙门吊设备资源分配

龙门吊（也称场桥）是堆场内承担装卸作业的主要设备，龙门吊作业调度和路径优化的相关研究已成体系，主流的解决方法是建立解析模型[33]，提出相应算法。

Chung[34]等提出了龙门吊的路径选择问题，目的是减少龙门吊的不必要移动，提高龙门吊的工作效率。Lai 和 Lam[35]运用模拟方法比较了不同配置策略下龙门吊的使用效率、作业量及等待时间。Chuqian Zhang[36]认为龙门吊的工作量分布随着时间的推移而变化，因此需要对龙门吊在各街区之间进行动态配置。他将堆场工作的延迟最小作为优化目标；采用混合整数规划，拉格朗日松弛变量法获得接近最佳解决方案。后来[37]，他通过建立一个整数规划模型研究了龙门吊的工作任务分配问题，但是该模型是在给定龙门吊台数基础上给出的配置方案，不能优化龙门吊配置台数。

(3) 堆场设备资源分配的其他相关研究

有的学者并不针对某项具体作业或者某种具体设备研究堆场设备资源的配置，而是立足于设备资源的规划层次，采用系统的观点，分析多种因素间的相互影响。

Gambardella[38]运用网络流方法对整个港口的装卸资源进行了仿真研究，运用混合整数规划模型对桥吊、龙门吊等港口资源进行配置，最后得到一张计划时域内的资源配置计划表。Lu Chen[39]等以改善各种设备间的合作，提高港口作业效率为目标，研究了集装箱处理问题，作者将此问题归结为一个带有优先约束、调整次数和模块化的混合流水线计划问题，用禁忌搜索方法解决此问题。

在国内，真虹[40]通过仿真证明，岸桥集中作业时，平均单船作业时间最短，但泊位利用率低。因此，当船舶到港密度较小或码头服务能力较强时，岸桥集中作业相对于均匀分配方式，更有助于减少船舶在港逗留时间。张新艳[41]用基于生物进化策略对堆场集装箱运送顺序进行优化，实现了基于生物进化策略的港口集装箱物流子系统的优化，并实现了基于面向对象的 petri 网的仿真模型。

1.2.3 知识工程和系统仿真在堆场作业计划中的应用研究现状

(1) 知识工程相关方法在堆场作业计划中的应用研究现状

知识工程的概念由 E. Feigenbaum 于 1977 年提出，它的研究内容包括知识的表示、知识运用、知识获取的理论、方法及实现，融合了人工智能、数据库技术、数理逻辑，是研究知识信息处理的学科[42]。

堆场作业计划、资源配置中存在许多优化问题，需要知识工程中的搜索技术，其中启发式方法、禁忌搜索、遗传算法等智能搜索方法的应用较多，规则、知识的应用有待进一步发展。Ebru K.Bish[11]针对某集装箱港口的多个起重机受限的调度问题进行了研究，开发了一种启发式的算法，解决了调度的 NP 难题，使每艘船的服务时间达到最小。Ebru K. Bish[43]等建立了基于车辆的集装箱港口集卡分派模型，目标为船舶在港时间最小，并给出了启发式算法。Kim 和 Lee[44]研究了集装箱提箱/进箱集卡的等待问题，对静态排队问题提出动态规划模型，对集卡连续、随机、动态到达情况下，提出基于学习的产生决策规则的方法，列出了一些启发式规则，实现集卡等待时间最小化的目标.

Kozan 和 Preston [45]研究了多式联运港口中装船/卸船作业时，集装箱的堆放和转运问题，目标是确定最优存储策略和最优集装箱处理计划。在班轮周期性到达情况下,提出了集成集装箱转运模型和集装箱箱位分配模型的迭代搜索模型，用遗传算法、禁忌搜索算法、遗传/禁忌搜索混合算法求得最优解。Mattfeld 和 Orth[46]对于多式联运港口被指派执行堆场作业的拖运车调度问题，提出了含有面向周期的生产能力利用策略的进化算法，该策略控制指派车辆在指定时间到指定地点的启发式构建。达到了提高作业效率，平衡计划期间车辆的移动分布的目标。

(2)系统仿真方法在堆场作业计划中的应用研究现状

系统仿真是以数学理论、相似原理、信息技术、系统理论及其应用领域有关的专业技术为基础，以计算机和各种物理效应设备为工具，利用系统模型对实际的或设想的系统进行模拟研究的一门综合技术[47]。目前，系统仿真的发展方向大致包括建模方法学，一体化，仿真数据库，面向对象仿真，分布交互仿真，开发并发、开放、多媒体仿真环境，智能仿真，嵌入式仿真，基于虚拟现实的仿真技术等[48]。总体呈现出以下几个方面的变化：仿真规模由小到大，从局部到整体，由以实物及外场为主向以数学模型及实验室内仿真为主。仿真体系结构从集中式、封闭式向分布式、交互式转变[49]。

系统仿真的方法在集装箱港口、堆场管理方面的研究和应用较多，取得了较好的应用效果，但仍然有待进一步发展，难点在于港口系统仿真模型的建立。Yun, Won Young 和 Choi, Yong Seok 用面向对象的方法，利用 SIMPLE++仿真软件，建立了港口系统分析仿真模型，模型中将港口系统分为大门、堆场、泊位三部分，作业设备有正面吊、龙门吊、拖车、堆场牵引车。仿真试验对各种设备的利用率、堆场空间占用率、船舶的等待时间等指标进行了分析[6]。Ballis 和 Abacoumkin[50]建立了含有由于设备不匹配而引起

的交通堵塞和延迟的仿真模型，使用该模型来评价码头的堆场设计、设备数量、集卡到达和运行规则。Kozan[51]研究了利用火车疏运集装箱的码头吞吐量，并用启发式理论将火车分配给轨道，将起重机、叉车和跨运车分配给集装箱，通过仿真分析不同运作模式对码头集装箱作业的影响。

Gambardella[52]等将装船／卸船作业时，资源的配置和计划问题形式化，分层次解决。该方法的可行性通过一个离散事件仿真模型来验证，仿真结果表明：资源配置和装卸作业顺序计划相配合，减少了堆场拖运车的数量，降低了集卡的平均队长。金淳等[53]针对港口堆场作业的调度问题，提出一种智能仿真方法，包括系统状态评价、作业规则、堆垛高度的调整和作业计划。为了实现最优作业调度，建立了基于模糊神经网络的控制结构，调度过程分为两个阶段：一是到港集装箱数量的预测阶段，二是确定作业规则和堆垛高度的推理阶段。作业计划是一个带有作业约束的模糊多目标规划问题，用遗传算法结合仿真的方法解决该问题。

1.3 研究内容和技术路线

本文对集装箱堆场作业计划进行了研究，就港口大门一侧的提箱作业计划和箱位分配作业计划进行深入分析、建立模型、提出解决方案。研究工作的主要内容有：

(1) 分析箱位分配作业问题，结合知识工程的相关方法，提出基于知识的箱位分配模型，为随机到达港口的一个出口集装箱分配最佳箱位。

(2) 分析提箱作业问题，建立提箱作业优化模型，结合遗传算法，求得既满足堆场作业规则，又使倒箱作业成本最小的提箱作业方案。

(3) 设计并实现集装箱堆场大门作业计划系统，系统中实现上述两种作业计划算法，建立堆场数据库，使得箱位分配作业计划和提箱作业计划可以实时地、连续地进行。

(4) 按照离散事件系统仿真的基本原理，采用相关仿真算法实现仿真控制，开发离散事件系统仿真工具。

(5) 对箱位分配作业和提箱作业计划模块进行性能测试。

本文涉及的技术方法主要有：

(1) 用基于知识的方法解决箱位分配作业计划问题；

(2) 用内外两层嵌套的遗传算法解决提箱作业优化问题；

(3) 用 VC++的 ATL COM 技术开发箱位分配作业计划算法和提箱作业优化算法；在此基础上，基于 VC++的 MFC 开发了集装箱堆场大门作业计划系统，调用算法的 COM 组件，建立了堆场数据库；

(4) 采用面向对象的方法设计、开发离散事件系统仿真工具，用动态连接库(Dynamic Link Library, dll)技术实现，提供 API 接口；

(5) 用基于仿真的方法，将箱位分配作业计划算法和提箱作业优化算法的 COM 组件嵌入仿真系统中，对上述两个算法模块进行性能测试。

图 1.3 说明了研究内容和技术方法之间的关系，图 1.4 说明了本研究的技术路线。

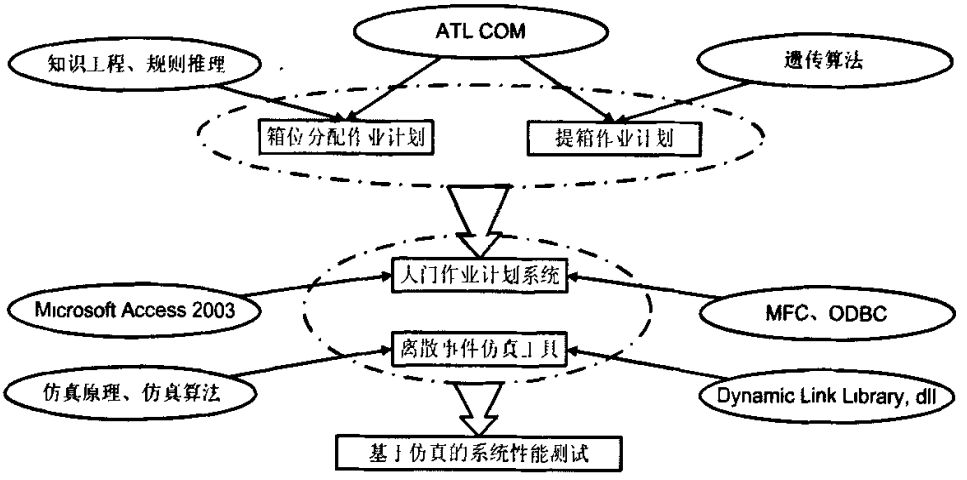


图 1.3 研究内容和技术方法之间的关系

Fig. 1.3 Relationship between research content and technology method

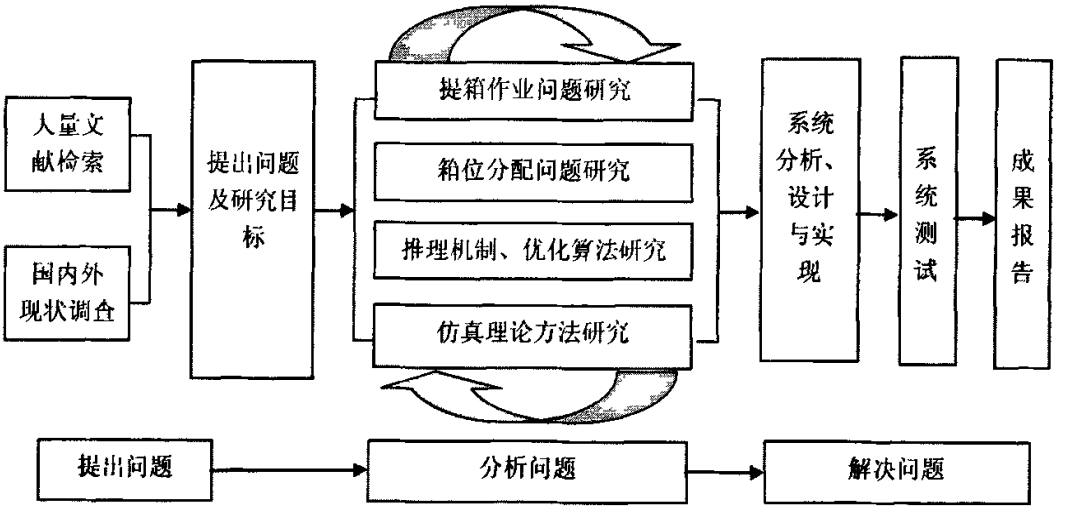


图 1.4 技术路线

Fig. 1.4 Technology route

2 基于知识的集装箱堆场箱位分配计划研究

箱位分配是一种以堆场管理原则为中心的计划活动，其中存在较多的规则和制约因素，为此，本文提出并实现一种基于知识的，方便后续作业的集装箱箱位分配计划方法。

2.1 箱位分配作业知识构成及其表示

2.1.1 箱位分配作业知识构成

在出口集装箱的箱位分配问题中，根据堆场管理原则和作业设备的特点，可得到多种关于作业的知识，按其作用可分为：分配区域划分规则、堆垛作业模式和设备作业规则。

(1) 分配区域划分规则

堆场集装箱的堆放一般遵循 PSCW 原则，即对同一目的港(Port)、同一尺寸(Size)、同一种类(Category)的集装箱，按重量级别(Weight)堆放在堆场的同一贝上，满足 PSCW 原则的集装箱集合为同类箱组。按照这一原则，可以得到相应的分配区域划分规则，将同类箱组放在同一区域。

(2) 堆垛作业模式

堆垛作业模式（以下简称模式）指的是，对于街中的任意一个贝的各行，集装箱堆垛作业的方向及先后次序。常见的堆垛作业模式有：从小行到大行、从大行到小行、从中间到两边（如图 2.1）。

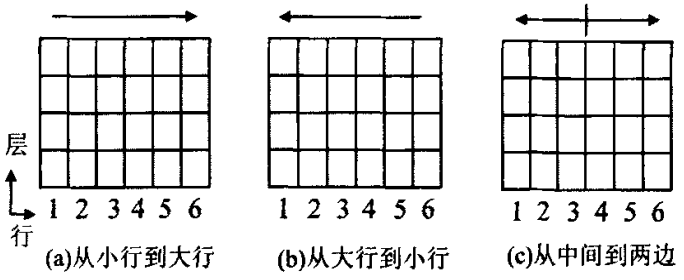


图 2.1 堆垛作业模式

Fig. 2.1 Stow operation mode

(3) 设备作业规则

设备作业规则（以下简称规则）和设备的物理及空间的作业特性有关，它影响到设备在物理及空间上的作业能力和对同一个贝中不同箱位的作业顺序。通过作业能力可以

判断某种堆场状态下设备能否将集装箱放入某位置，通过作业顺序则可判断设备在当前贝中应当优先选择的位置。堆场的作业设备有堆场桥式起重机（场桥）、正面吊、叉车等多种，不同类型设备的作业能力和作业顺序各不相同，均需要对作业规则作合理定义。

2.1.2 知识表示方法

知识可以认为是在实践中应用并证实一个有组织的概念或框架时产生的结果，由一个包含语义信息的特征集以及与之相关的约束和规则集组成。从工程的角度，知识被理解为有助于解决问题的可复用的模式化的信息，一般表示形式为：概念、规则、规律、模式、约束和可视化。知识表示则是描述这些结果所做的一组约定并易于被计算机接受和处理的一种表现形式，是知识的符号化、编码化过程。知识表示主要是选择合适的形式表示知识，是对知识进行输入、分类、标准化等一系列的加工和处理，使其便于公开、共享和交流,并能够通过信息手段进行传递。

知识表示领域的核心是解决如何进行信息的编码并对推理计算模型加以利用，其表示方式常常取决于人类知识的结构及其机制[54]。传统的知识表达模式有谓词逻辑方法、框架、产生式规则、状态空间、语义网络、脚本、直接表示法、面向对象的知识表示方法等。近几年，由于将本体引入知识工程领域，知识表示领域又出现了一些新的方法。

(1) 产生式规则

产生式是最早采用的知识表示方法。Shortloffe 在著名的专家系统 MYCIN 中,首先引入了产生式的概念。

产生式通常表示具有因果关系的知识，其基本形式是：

P→Q， 或者 IF P THEN Q

其中 P 是产生式的前提，用于指出该产生式是否可用的条件；Q 是一组结论或操作，用于指出当前提 P 被满足时，应该得出的结论或应该执行的操作。整个产生式的含义是：如果前提 P 被满足，那么得出结论 Q 或执行 Q 所规定的操作。例如：

r1： IF 动物会飞 AND 会下蛋 THEN 该动物是鸟

就是一个产生式，其中 r1 是产生式标识号，“动物会飞 AND 会下蛋” 是前提，“该动物是鸟”是结论。

产生式的形式化描述，可用巴科斯范式 BNF(Backus Nromal Form)表示，描述如下：

<产生式>::=<前提>→<结论>

<前　提>::=<简单条件>|<复合条件>

<结　论>::=<事实>|<操作>

<复合条件>::=<简单条件> AND <简单条件>[(AND<简单条件>)...]

|<简单条件>OR<简单条件>[(OR<简单条件>)...]

<操　作>::=<操作名>[(<变元>,...)]

产生式可以表示精确的知识，也可以表示不精确的知识。用产生式表示知识的系统中，决定一条知识是否可用的方法是检查当前是否有已知事实可与前提中所规定的条件匹配，这种匹配可以是精确的，也可以是模糊的[55]。

(2) 面向对象表示法

面向对象的的知识表示方法可以将产生式规则、框架等按照面向对象程序设计方法组成一种混合知识表达形式。面向对象方法的主要特点是封装、继承和多态。封装是指以对象为中心，将对象的属性、行为和特征、相关领域的知识和数据处理方法等有关知识封装在类中。

面向对象方法的继承性体现了概念分离抽象，在对象继承结构上，下层对象继承上层对象的特征(属性和行为)，因而便于知识的演化和增量式扩充。同时面向对象方法具有信息的隐藏性，对象将其实现细节隐在内部，而不会对外界产生影响，这就保证了有关对象知识的可构造性和易维护性。面向对象方法以抽象数据类型为基础，方便地描述复杂对象的静态特征、动态行为和相互关系，既集中了 单一知识表示方法的优点，又符合专家对领域对象的认识模式。

2.1.3 箱位分配作业知识表示及获取

(1) 知识表示

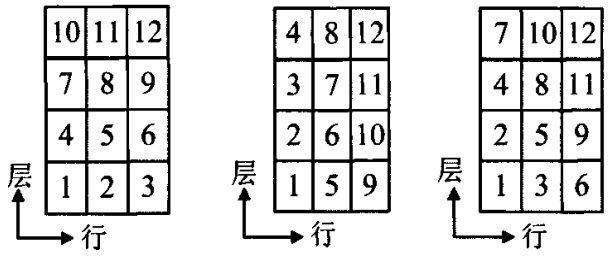
在箱位分配问题的三类知识中，分配区域划分规则和堆垛作业模式采用适合于表达判断性知识的产生式规则表示，其形式为：

IF (前提 1)&(前提 2)&.....&(前提 p) THEN(结论 1)&(结论 2)&.....&(结论 q)，例如：

IF (目的港=天津)&(尺寸=20) &(箱型=普通干货箱) THEN (分配区域=***F01***)。

IF (街代码=A01)& (贝号=1)THEN(作业模式=从小行到大行)。

设备作业规则可以由设备作业模板来表示。若设备作业能力最大跨度为 m 行，最高层为 n 层，则设备作业规则可由一个 $m\times n$ 的矩阵 O_{mn} 来反映，这个矩阵就是作业模板。其中 m 和 n 定义了设备的作业能力，矩阵中存放 1,2,...$m\times n$ 的自然数，它定义了设备对贝中各个箱位的作业顺序。图 2.2 列举了 m=3，n=4 时，场桥的作业模板 O_{34} 示意图，包括垂直作业、水平作业和混合作业。



(a) 水平作业　(b) 垂直作业　(c) 混合作业

图 2.2 设备作业模板示意图（m=3，n=4）

Fig. 2.2 Sketch map of operation template of facility (m=3, n=4)

设备作业模板在形式上为一个矩阵，但本质上表达的是设备作业所遵循的规则。若用产生式规则[50]来表示模板所表达的知识，相当于如下语句：

IF 位置行号=i & 位置层号=j;

THEN 位置(i,j)的作业顺序号=k.

其中 $i\in[1,m]$，$j\in[1,n]$，$k\in[1,m\times n]$

用模板表示相当于集成了多条产生式规则，简化了知识的表示，在形式上更加直观。

(2) 知识获取

从纯技术的角度来看，知识获取是指知识从外部知识源到计算机内部的转换过程。就是将一些问题求解的知识从专家的头脑中和其它知识源中提取出来，并按照一种合适的知识表示方法将它们转移到计算机中。

箱位分配问题的中的知识采用基于数据库的获取方法得到，每一类知识分多张表存储（参见附录 A），每张表的若干字段集合表示规则前提，其余字段表示规则结论。这样，通过数据库实现对规则的管理。

例如，通过表 T（field1, field2, ...field i, field i+1,... fieldn），可表示规则 IF(field1) & (field2) &.....& (field i) THEN (field i+1) &.....& (fieldn)，表中每条记录可表示具有该种结构的规则，通过对表 T 的增、删、改实现对规则的管理。

2.2 基于知识的箱位分配模型

2.2.1 模型原理

对于任意一个集装箱，其箱位分配的过程可看作按上述知识和规则运作的许多判断活动的集合[51]，每经过一个（或一组）判断，可选择箱位的范围就相应减少，将最终选出的集装箱位置作为最优位置分配给集装箱。设问题的状态空间为 $U<S,F,P,C>$，其中 S 为堆场状态集合，F 为作业设备集合，P 为计划分配的箱位集合，C 为待分配的集装箱。

设最初状态空间为 $U_0<S_0,F_0,P_0,C>$，其中 S_0 代表整个堆场的堆放状态，F_0 代表不确定的作业设备，P_0 代表整个堆场中的箱位。箱位分配模型表示如下图：

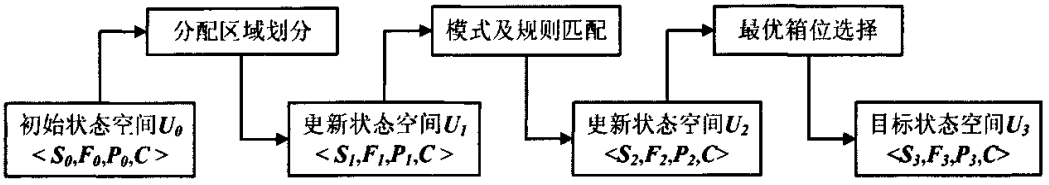


图 2.3 基于知识的箱位分配模型

Fig. 2.3 Knowledge-based model of storage space allocation

① 分配区域划分。通过分配区域划分规则判断，将待分配集装箱的箱位选择范围从整个堆场缩小至规则定义的某一区域。状态空间更新为 $U_1<S_1,F_1,P_1,C>$，S_1 代表分配区域堆场状态，F_1 代表分配区域的作业设备，P_1 代表分配区域中的箱位集合。

② 模式及规则匹配。将堆垛作业模式、设备作业规则和堆场状态相匹配，排除不满足堆垛作业模式和设备不可作业的位置，状态空间更新为 $U_2<S_1,F_1,P_2,C>$，P_2 代表分配区域中符合分配条件的箱位集合。

③ 最优箱位选择。从 P_2 中选择最优箱位，状态空间更新为 $U_3<S_1,F_1,P_3,C>$，P_3 代表最终选定的最优箱位。

2.2.2 分配区域划分

分配区域的划分过程如图 2.4 所示。

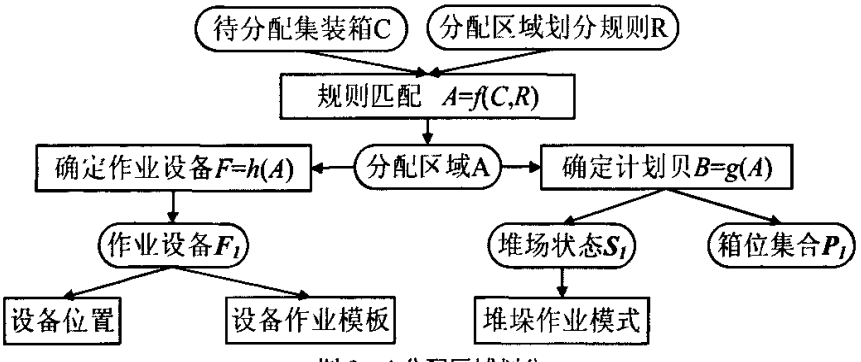


图 2. 4 分配区域划分

Fig. 2.4 Partition of allocation area

首先，进行出口箱区域划分。分配区域的划分通过匹配待分配集装箱的 PSCW 属性和分配区域划分规则实现，规则匹配可由公式(2.1)表示。

$$A = f(C,R) \tag{2.1}$$

其中，f 是规则匹配函数，A 是分配区域，R 是规则集，C 是集装箱属性集。

其次，每个分配区域由若干个计划贝组成，通过 $B=g(A)$ 确定，其中，g 是计划贝的定义函数，B 为计划贝集合。计划贝的堆垛作业模式、贝中的箱位是否放有集装箱，可由堆场状态 S_1 表示，区域中的所有箱位由箱位集合 P_1 表示。

然后，港口的堆场管理可根据现场情况，为区域调度作业设备，设调度函数为 $F=h(A)$，其中，F 是调度的作业设备，设备的作业模板和位置由 F_1 确定。

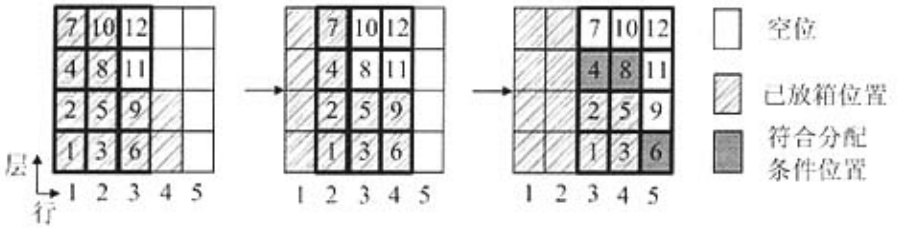
经过以上步骤之后，问题状态空间表示为 $U_1<S_1,F_1,P_1,C>$，模式及规则匹配的初始条件被确定下来。

2.2.3 模式及规则匹配

模式及规则匹配过程在方向上遵循堆垛作业模式，在作业顺序上按照设备作业模板。本文以图 2.1(a)的“从小行到大行”的作业模式和图 2.2(c)的“混合作业”模板为例，说明匹配方法。

步骤 1：模板最小序号所在行和作业起始行对齐（图 2.5(a)），若该行已放满箱，模板右移一行（图 2.5(b)），直到模板最小序号所在行没有放满箱，匹配成功（图 2.5(c)），模板中的所有位置被赋予相应的作业顺序号，如图 2.5(c)所示，模板区域内的所有箱位均带有 1~12 的作业顺序号。

步骤 2：根据堆场状态和设备作业能力，选出可以分配的箱位，如图 2.5(c)中顺序号为 4、6、8 的箱位。



(a) 模板匹配起始状态　(b) 模板右移一行　(c) 匹配结果

图 2.5 模板匹配过程（从小行到大行）

Fig. 2.5 Template match process (ascendant mode)

堆垛作业模式为“从大行到小行”下的模板匹配，则将模板对称翻转，从贝的最大行开始，按类似于“从小行到大行”的匹配方法进行。“从中间到两边”的模板匹配时，其原理相同，中间的作业起始行往小行方向一侧按“从大行到小行”模式匹配，另一侧按“从小行到大行”模式匹配。

2.2.4 最优箱位选择

模式及规则匹配后，得到分配区域中符合分配条件的箱位集合 P2，还需要从 P2 中挑选一个最优箱位作为最终为集装箱分配的箱位。最优箱位选择的依据有两个：

① 集装箱的堆放尽量按设备作业模板顺序进行；

② 集装箱的摆放满足重量等级要求，即符合重箱压轻箱，或者轻箱压重箱的原则。

由此，最优箱位的选择可分为以下 3 个步骤。

步骤 1：按作业模板序号排序。由模式和规则匹配可知：匹配得到的箱位，同一贝中每行至多 1 个（图 2.5(c)中序号为 4、6、8 的箱位分布在不同的行中）。对于一个 m 行的模板，可以将匹配得到的箱位分成 m+1 个等级，其中 1~m 个等级在模板范围之内，按序号从小到大排列；第 m+1 个等级为模板范围之外的符合分配条件的箱位。

设分配区域共有 t 个贝，模板为 m 行，则经过模式和规则匹配后，满足分配条件的箱位集合 $\boldsymbol{P_2}$ 如式(2.2)所示。

$$P_2=[\rho_1,\rho_2\cdots\rho_m,\rho_{m+1}]=\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} & p_{1(m+1)} \\ p_{21} & p_{22} & \cdots & p_{2m} & p_{2(m+1)} \\ & & \cdots\cdots & & \\ p_{t1} & p_{t2} & \cdots & p_{tm} & p_{t(m+1)} \end{bmatrix} \tag{2.2}$$

其中 $\rho_i=\begin{bmatrix} p_{1i} \\ p_{2i} \\ \cdots \\ p_{ti} \end{bmatrix}$ (i=1,2…m+1)，且作业顺序号满足 $p_{j1}<p_{j2}<\ldots<p_{j(m+1)}$ (j=1,2…t)。

步骤 2：重量等级判断。出口箱的箱位分配需要按照重量等级要求进行，如重箱压轻箱，或者轻箱压重箱的原则。设待分配箱位的集装箱的重量等级为 w_0，位于 $\boldsymbol{P_2}$ 中的箱位 p_{ij} 下方的集装箱重量等级为 w_{ij}，重量等级规则函数为 $f(w_0, w_{ij})$，若 $f(w_0, w_{ij})$ 值为 true，则 p_{ij} 满足重量等级要求，否则不满足。

在上一步中，$\boldsymbol{P_2}$ 中所有箱位被分成 m+1 个等级，因此重量等级判断的顺序按照等级高低进行。首先对 ρ_1 中各个分量判断，若存在 $p_{1j}\in\rho_1$ 满足重量等级要求，则 $\rho_2,\rho_3,\ldots\rho_{m+1}$ 无须判断；若不存在 $p_{1j}\in\rho_1$ 满足重量等级要求，则对 ρ_2 中各个分量判断，依此类推。

步骤 3：选择最优箱位。设满足重量等级要求的的箱位集合为 $P_2'=[p_{21}', p_{22}', \ldots p_{2k}']$，最优箱位的选择以当前设备移动距离最小为原则。令设备所在贝为 b_{fac}，箱位 p_{ij}'所在贝为 b_{ij}，mov_{ij} 为设备移动到箱位 p_{ij}'的距离，根据公式 (2.3) 得到移动距离最小的箱位，作为最终选择分配的箱位 P_3。

$$\min mov_{ij} = | b_{fac} - b_{ij} | \tag{2.3}$$

2.3 实例分析

集装箱随机到达港口，限于篇幅，本例只对具有相同 PSCW 属性的集装箱进行分析。设到达集装箱的数量为 30，其重量等级服从均匀分布 U(1,3)，具有属性：目的港为天津、尺寸为 20 尺、箱型是普通干货箱。设集装箱箱号为 *Ti*（*i* 表示集装箱的到达顺序），并存在以下分配区域划分规则：

IF (目的港=天津)&(尺寸=20) &(箱型=普通干货箱) THEN (分配区域=*F01*)

该批集装箱分配区域在 F01，假设该区域有 2 个贝：A01005，A02007；最初没有放任何集装箱，堆垛模式及贝定义如表 2.1 所示。

设为区域 *F01* 作业的设备为场桥，其初始位置为 A01001，作业能力为 4 层，作业模板采用图 2.2(c)所示的混合作业模板。

表 2.1 堆垛模式及被定义

Tab. 2.1 Definition of stow mode and bay

计划贝	堆垛模式	行数	层数	轻重原则
A01005	小行到大行	6	4	重压轻
A02007	大行到小行	6	4	重压轻

采用本方法的得到的箱位分配结果如图 2.6 所示，图中集装箱由“箱号/重量等级”表示，箱号也表示集装箱的到达序号。

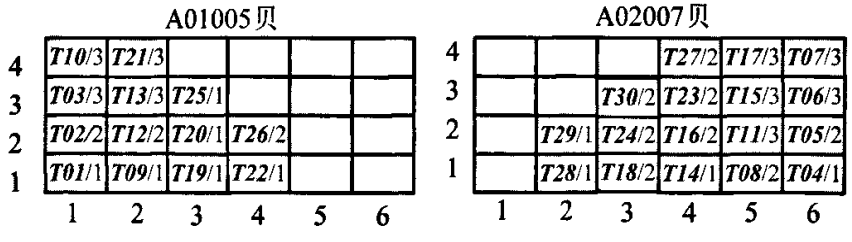


图 2.6箱位分配结果

Fig. 4.6 Result of container allocation

这里以最后一个箱 ***T30*** 为例说明按照箱位分配模型所述步骤的最优箱位选取过程：

(1) 当 ***T30*** 到达后，根据分配区域划分规则，将 ***T30*** 划入 ***F01*** 区域；此时，问题的状态空间为：$U_1<S_1,F_1,P_1,C>$，他们的具体含义如下。

S_1：表 4.4 所列贝定义及堆垛模式，计划贝中各个位置；

F_1：场桥，当前位置为 A02007，作业模板；

P_1：A01005 贝、A02007 贝中的所有箱位；

C：***T30***

(2) 用图 2.2(c)所示的“混合”作业模板匹配 A01005 贝和 A02007 贝，问题的状态空间更新为 $U_2<S_1,F_1,P_2,C>$，其中 S_1、F_1 和 C 含义不变，P_2={(A01005,5,1), (A01005,3,4), (A01005,4,3), A02007,1,1), (A02007,2,3), (A02007,3,3)}。

将 P2 中的所有位置分贝按模板序号排列，得到：

$$P_2=\begin{bmatrix}(A01005,5,1)&(A01005,3,4)&(A01005,4,3)\\(A01007,3,3)&(A01007,1,1)&(A01007,2,3)\end{bmatrix}$$

(3) 各个位置的评价结果如表 2.2 所示。

表 2.2 各位置评价结果

Tab. 2.2 Evaluation result of each position

位置	作业序号	重量等级满足否
(A01005, 5, 1)	6	Y
(A01005, 3, 4)	7	Y
(A01005, 4, 3)	8	Y
(A02007, 3, 3)	4	Y
(A02007, 1, 1)	6	Y
(A02007, 2, 3)	7	Y

由表 2.2 可见，(A01005,5,1)、(A02007,3,3)均满足重量等级要求，然而，由于设备对 ***T29*** 箱作业完毕后停留在 A02007 贝，根据设备移动距离最小的原则，求得(A02007,3,3)为最优位置。

通过该方法将堆场的管理规则作为箱位分配的依据，充分利用箱位分配作业知识，所产生的堆垛符合重量等级要求，方便后续装船作业，同时考虑了设备的移动距离，减少了作业成本。

3 基于嵌套遗传算法的集装箱堆场提箱作业计划研究

本文所研究的问题是：在集装箱堆场同一街的给定若干贝范围内倒箱，不同类型作业设备（叉车、正面吊、场桥，以下称设备）以不同的方式作业，如何求得使提箱作业总成本最小的作业方案。此问题涉及提箱作业过程中倒箱路径的选择、堆场放置状态的变化、设备作业模式等多因素的组合，是一个组合优化问题。本文针对此问题提出了一种嵌套算法对提箱作业进行优化。

3.1 提箱作业描述

集装箱堆场的堆存状态如图 3.1 所示，每个箱位由三元组（贝，行，层）决定。对于任意一个待提的集装箱，倒箱作业在该箱无法直接提取时就需要执行。以提取位于图 3.1 中（5，3，1）位置的集装箱为例，就有 3 个集装箱需要倒走，他们分别位于（5，3，2），（5，3，3）和（5，3，4）。提箱作业就着眼于怎样将待倒集装箱移到各自的目标位置，使得整个提箱作业过程的成本最小，并减少作业时间。

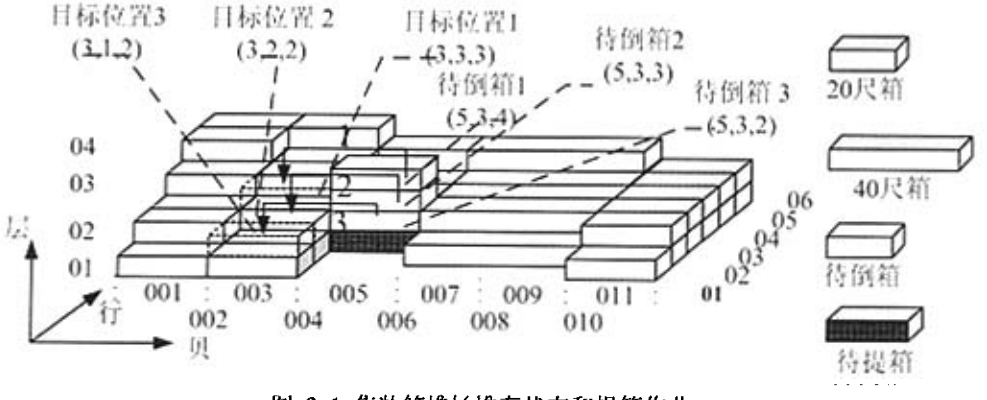


图 3.1 集装箱堆场堆存状态和提箱作业

Fig. 3.1 Configuration of the container stacking and pick-up operation

本研究假设：

(1) 提箱作业考虑在一条街的范围内进行；

(2) 作业计划时只指定一种设备进行作业。

并非所有堆场中的空位可以放置待倒箱，因为其目标位置的选择须遵循一定的作业规则，这些规则也是构成模型的约束条件。主要的作业规则如下：

规则 1: 目标位置必须是空的，而且在指定的作业范围之内。

规则 2: 即使在相同的堆场堆存状态下，不同的作业设备具有不同的作业模式，因此目标位置的选择应当符合指定作业设备的特点。

规则 3: 待到箱和目标位置所在贝的尺寸应当相匹配，即 20 尺箱不允许放在 40 尺的贝中，反之亦然。

规则 4: 堆场中某些特殊的箱型是不可压的，位于这种集装箱上方的位置不能作为目标位置。

规则 5: 所有事先已经被预订提取的集装箱称为预约箱，位于预约箱上方的位置也不能作为目标位置。

3.2 提箱作业优化模型的建立

集装箱提箱作业优化模型由 2 部分构成：倒箱优化模型和最短路搜索模型，其中最短路搜索模型计算一次倒箱的最小路径成本，它嵌入在倒箱优化模型之中。

3.2.1 模型中符号的定义

模型中符号定义如下：

b: 街中包含的贝数目；

r: 街中包含的行数目；

t: 街中包含的层数目；

n: 待倒箱个数；

s_i: 第 i 个待倒箱在街中的原位置坐标，$s_i=(sb_i,sr_i,st_i)$；

$\boldsymbol{S}$: 按倒箱先后顺序排列的全部待倒箱原始位置的集合，$\boldsymbol{S}=\{s_1,\ldots,s_n\}$；

d_i: 第 i 个待到箱的目标位置在街中的坐标，$d_i=(db_i,dr_i,dt_i)$；

$\boldsymbol{D_i}$: 第 i 个待到箱的目标位置集合，$\boldsymbol{D_i}=\{d_{i1},d_{i2}\ldots d_{ij}\ldots d_{im}\}$，$m$ 为满足作业规则的目标位置个数。

$\boldsymbol{H_i}$: 将第 i 个待倒箱从 s_i 倒至 d_i 的作业最小成本；

$\boldsymbol{R_i}$: 设备空车从 d_i 移动到 s_{i+1} 返回最小成本；

$\boldsymbol{TC}$: 全部倒箱作业的总成本，其值为 $\Sigma \boldsymbol{H_i}+\boldsymbol{R_i}$；

$\boldsymbol{X}$: 表示模型的一组可行解，$\boldsymbol{X}=\{d_{1j1}\ldots d_{ijl}\ldots d_{njn}\}$，$d_{ijl}$ 表示待倒箱 i 最终被放置的位置。优化模型的目标就是找到一个使得 $\boldsymbol{TC}$ 最小的 $\boldsymbol{X}$ 。

$\boldsymbol{B_i}$: 当倒第 i 个集装箱时堆场的堆存状态，即街中所有位置的集装箱放置状态集合；

3.2.2 倒箱优化模型

对于某一指定的待提箱，根据堆场堆存状态和设备作业特点可以计算得到所有待倒箱的数目和原位置，即 n 和 S。把位于 s_i 的待倒箱倒走可分为 2 步：

(1)：把待倒箱从 s_i 倒至 d_{ij}，产生倒箱作业成本 H_i，堆场状态改变。

(2)：设备空车返回到下一个待倒箱位置 s_{i+1}，产生空车返回成本 R_i。

随着待倒箱逐个移至各自的目标位置，堆场状态一直不断地发生变化。假设倒第 i 个集装箱时的堆场当前装态为 $Q_i=[B_i, s_i, D_i]^T$，那么堆场的状态转移函数为：

$$B_{i+1}=T_i(D_i,d_{ij}) \tag{3.1}$$

这里，$T_i(D_i, d_{ij})$ 表示将 d_{ij} 从 D_i 中选出来作为位于 s_i 的待倒箱 i 的目标位置，进而堆场状态转化为 B_{i+1}。所以，当且仅当待倒箱 i 的目标位置确定后，待到箱$(i+1)$的目标位置集合 D_{i+1} 才能确定。由此可见，倒箱作业是组合的、动态的复杂问题；倒箱作业优化问题可以归结为多阶段决策问题，其中的每个阶段由以上描述的两步组成。

当待倒箱数目为 n 时，倒箱作业优化模型的状态空间可以表示成如图 3.2 所示的 n 层树形结构。

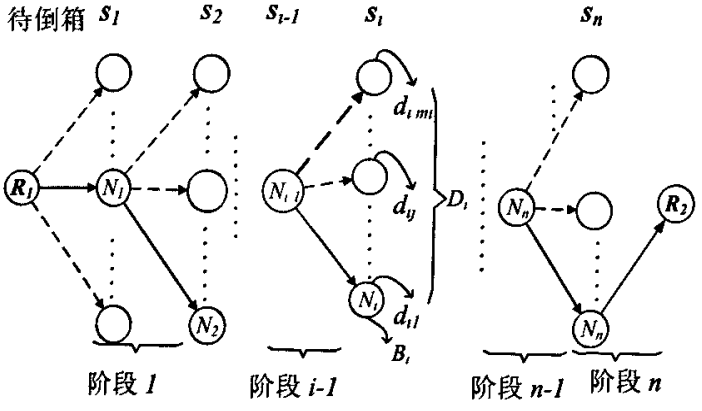


图 3.2 倒箱优化模型的树形结构状态空间

Fig. 3.2 The tree structure of rehandling operation scheduling model

根节点 R 是一个虚拟节点，它也可以被看作是待提箱。每个节点由 3 部分组成：堆场状态 B_i，待倒箱 i 的原位置 s_i 和目标位置 d_{ij}。节点 $Ni\ (B_i, s_i, d_{ij})$ 即意指在堆场状态为 B_i 时将待倒箱从 s_i 移至 d_{ij}，每个分支表示设备返回到下一个待倒箱所在位置 s_{i+1}，这样，每个节点及其分支就是一个倒箱阶段。例如，N_1 及连接 N_1 和 N_2 的分支即为

第 1 阶段，表示位于 s_1 的待倒箱被倒至 d_{1j}，然后设备返回到 s_2，作业成本在这个过程中产生。该过程不断重复直至第 n 个阶段完成，即 s_n 被倒走并且设备返回到待提箱所在位置。第 i 阶段的作业成本（H_i+R_i）逐个累加到总成本 TC 中，而从 D_i 中选择 d_{ij} 得到 X 使得 TC 最小是倒箱优化模型的目标。

倒箱优化的数学模型可由公式 (3.2)表示，其中的 $f_k(d_{ij})$ (k=1,2,3,4,5)是根据作业规则得到的约束函数，公式 $f_k(d_{ij})$=1 表示目标位置 dij 满足第 k 个约束条件。H_i 和 R_i 的最小值由下一个部分将要讨论的最短路搜索模型计算。

$$
\begin{aligned}
&min \quad TC = \sum_{i=1}^{n} (H_i + R_i) \\
&H_i = h(s_i, d_{ij}) = h(sb_i, sr_i, st_i, db_i, dr_i, dt_i) \\
&R_i = r(s_{i+1}, d_{ij}) = r(db_i, dr_i, dt_i, sb_{i+1}, sr_{i+1}, st_{i+1}) \\
&s.t. \qquad f_k(d_{ij}) = 1 \qquad (k = 1,2,3,4,5)
\end{aligned}
\tag{3.2}
$$

3.2.3 最短路搜索模型

设备带箱或不带箱向各个方向移动的单位成本不同，把集装箱从原位置倒至目标位置的路径也有多条。因此，找到一条最短路径，使得 H_i 和 R_i 的值最小有着极大的意义，最短路搜索模型就负责完成这一功能。

在特定的堆场状态下，路径的成本值由路径的起点 S 和终点 T 决定，位于某个位置的集装箱最多有 6 个方向可以移动，包括上、下、左、右、前、后，寻找从原位置 S (b_0, r_0, t_0)到目标位置 $T(b_1, r_1, t_1)$ 的最优路径如图 3.3 示意。

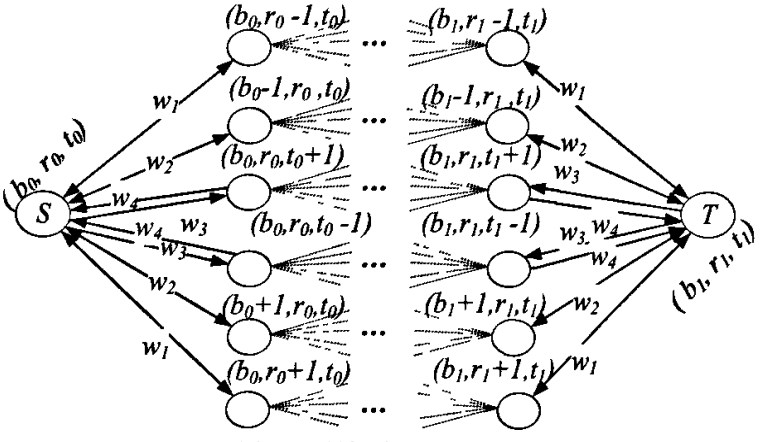


图 3.3 最短路径搜索过程

Fig. 3.3 Process of shortest path search

w_1, w_2, w_3, w_4 为设备分别进行贝移、行移、提升一层、下降一层作业时的单位移动成本；

假设在 S 和 T 之间有 t 个贝，设备带箱可通过的位置集合为 $POS_i\{p_{i1},p_{i2}...p_{i\ ki}\}$ ($1<=i<=t$，k_i 是 POS_i 中元素的个数)。当集装箱在各个贝间移动时，他从 S 出发，通过多个 p_{ij} 后到达 T，所以，路径可由 $U=\{S, p_{1g1}, p_{2g2},...p_{tgt}, T\}$ 表示。路径优化的目标就是找到一条使得从 S 到 T 的成本最小的路径 U，并将最小路径成本（H_i 或 R_i）传给外层模型

最短路径搜索的数学模型可由公式(3.3)和(3.4)表示。$M_{bi}, M_{ri}, M_{ui}, M_{di}$: 对于路径 i，设备作业过程中在 4 个方向上移动的距离。

$$\begin{aligned}
&\min H_i = h(s_i,d_{ij}) = v(s_i,p_{1k}) + \sum_{e=1}^{t} v(p_{eq},p_{e+1k}) + v(p_{tk},d_{ij}) \\
&\min R_i = r(s_{i+1},d_{ij}) = v(s_{i+1},p_{1q}) + \sum_{e=1}^{t} v(p_{eq},p_{e+1k}) + v(p_{tk},d_{ij}) \\
&1 \le q,k \le k_i, 1 \le e < t, p_{ij} \in P_i
\end{aligned} \tag{3.3}$$

$$\begin{aligned}
&v(S,T) = C_1 M_{bi} + C_2 M_{ri} + C_3 M_{ui} + C_4 M_{di} \\
&M_{bi} = |S_{bi} - T_{bi}| \\
&M_{ri} = |S_{ri} - T_{ri}| \\
&M_{ui} = \begin{cases} T_{ti} - S_{ti}, T_{ti} \ge S_{ti} \\ 0 \qquad , T_{ti} < S_{ti} \end{cases} \\
&M_{di} = \begin{cases} S_{ti} - T_{ti}, T_{ti} < S_{ti} \\ 0 \qquad , T_{ti} \ge S_{ti} \end{cases}
\end{aligned} \tag{3.4}$$

3.3 嵌套遗传算法设计

由优化模型的分析可知，外层模型是一个组合优化问题，内层模型是一个最短路径搜索问题。相关研究表明：遗传算法（GA）对这两类问题的求解均有较好的适应性。Maojun Li 针对组合优化问题提出了一种单性生殖遗传算法(PGA)，并分析了其模式定理和全局收敛性[56]。Miguel Rocha 研究了组合优化问题中的基于顺序号编码方法的遗传算法[57]。Gomez-Albarran 研究了基于遗传算法的路径优化策略[58]。Cedric Davies 利用遗传算法在权值是时间变化的函数的动态网络图中搜索最短路径[59]。

3.3.1 遗传算法

遗传算法是一类模拟进化算法，其基本思想源于 20 世纪 60 年代。当时 J.Holland 在研究机器学习过程中，提出了一种借鉴生物进化机制的所谓自适应机器学习方法，1975 年，他发表“Adaptation in natural and artificial systems”的专著，如今发展成为标准形式的遗传算法。

(1) 标准遗传算法

标准遗传算法的流程如图 3.4 所示：

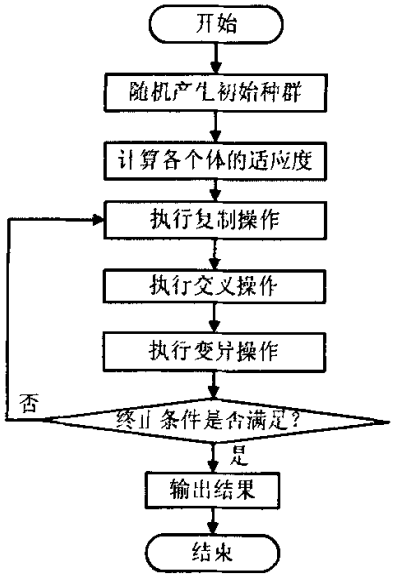


图3.4 标准遗传算法流程图

Fig. 3.4 Flow of standard genetic algorithm

构成标准遗传算法的要素有：

① 染色体编码方法。编码方法可以分为三大类：二进制编码方法、浮点数编码方法、符号编码方法。

② 个体适应度评价。根据不同种类的问题，必须预先确定好由目标函数值到个体适应度之间的转换规则，特别是要预先确定好当目标函数值为负时的处理方法。

③ 遗传算子。标准遗传算法的选择运算使用比例选择算子；交叉运算使用单点交叉算子；变异运算使用标准位变异算子或均匀变异算子。

④ 标准遗传算法的运行参数。标准遗传算法有 4 个运行参数需要提前设定：

M：群体大小。即群体中所含个体的数量，一般取为 20 至 100。

T：遗传运算的终止进化代数，一般取为 100 至 500。

pc：交叉概率，一般取为 0.4 至 0.99。

pm：变异概率，一般取为 0.0001 至 0.1。

(2) 遗传算法求解复杂系统优化问题的通用框架

遗传算法提供了一种求解复杂系统优化问题的通用框架，如图 3.5 所示。对一个需要进行优化计算的实际应用问题，一般可按下述步骤来构造求解该问题的遗传算法：

① 确定决策变量及其各种约束条件，即确定个体的表现型 X 和问题的解空间。

② 建立优化模型，即确定出目标函数的类型及其数学描述形式或量化方法。

③ 确定表示可行解的染色体编码方法，即确定出个体的基因型及算法的搜索空间。

④ 确定解码方法，即确定出个体基因型 X 到个体表现型 X 的对应关系或转换方法。

⑤ 确定个体适应度函数，即确定由目标函数 $f(X)$ 到适应度 $F(X)$ 的转换规则。

⑥ 设计遗传算子，即确定出选择、交叉、变异等遗传算子的操作方法。

⑦ 确定遗传算法的有关运行参数。

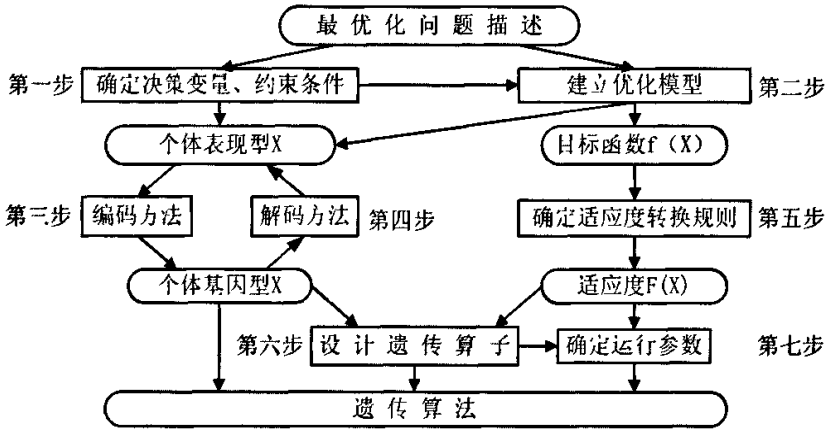


图3.5 遗传算法的主要构造过程示意图

Fig. 3.5 Flow of standard genetic algorithm

(3) 遗传算法优点

遗传算法是一类可用于复杂系统优化计算的鲁棒搜索算法，与其他一些优化算法相比，它主要有以下几个特点：

① 遗传算法以决策变量的编码作为运算对象。传统的优化算法往往直接利用决策变量的实际值来进行优化计算，但遗传算法对决策变量编码处理，使得我们可以方便地应用遗传操作算子，编码处理方式更显示出了独特的优越性。

② 遗传算法直接以目标函数值作为搜索信息。它仅使用由目标函数值变换来的适应度函数值，就可以确定进一步的搜索方向和搜索范围，无需目标函数导数值等其他辅助信息。这个特性对很多目标函数无法或很难求导的，或导数不存在的函数优化问题，以及组合优化问题等，应用遗传算法时就比较方便。再者，直接利用目标函数值或个体适应度，可把搜索范围集中到适应度较高的部分搜索空间中，从而提高了搜索效率。

③ 遗传算法同时使用多个搜索点的搜索信息。遗传算法从由很多个体组成的一个初始群体开始最优解的搜索过程，而不是从一个单一的个体开始搜索。对这个群体所进行的选择、交叉、变异等运算，产生出新的群体，其中又包括很多群体信息，所以实际上相当于搜索了更多的点，这是遗传算法所特有的一种隐含并行性。

④ 遗传算法使用概率搜索技术。与很多传统的优化算法使用确定性的搜索方法相比较，遗传算法属于一种自适应概率搜索技术，增加了搜索过程的灵活性。

3.3.2 提箱作业优化的嵌套遗传算法

本文针对提箱作业优化问题提出嵌套遗传算法，如图 3.6 所示。其中外层 GA 算法用于求解倒箱优化模型，内层 GA 算法求解最短路搜索模型；内层 GA 嵌入外层 GA 之中，计算每次倒箱的最优 H_i 和 R_i 值，并以参数形式传给外层 GA。内外层 GA 调用相同的 GA 流程，但有各自不同的适应度函数。

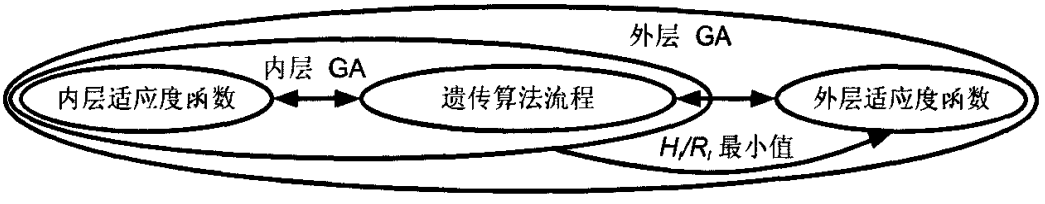


图3.6 嵌套遗传算法的结构

Fig. 3.6 The structure of nested genetic algorithm

遗传算法解决提箱作业问题的重要影响因素在于算法收敛速度和确保收敛到最优解，为此本算法设计采用父代参与选择方法，加快算法收敛速度；采用最优个体保留的方法确保收敛到最优解，其基本流程如下：

(1)外层 GA 初始化：确定外层 GA 杂交概率 P_c，变异概率 P_m，种群规模 *popsize*，随机产生初始种群 X_0，置遗传代数计数器 t=0；

(2)杂交：对种群 X_t，独立实施杂交，产生新种群 Y_t；

(3)变异：对种群 Y_t，独立实施变异，产生新种群 Z_t；

(4)评价：对种群 X_t+Z_t 中所有个体进行适应度评价，选出最优个体 ***BESTt***，调用内层最短路搜索遗传算法，内层 GA 的算法流程和外层相似。适应度评价使用适应度函数，该函数由目标函数得来，根据公式(3.2)、(3.3)可知，内层为外层提供 H_t 和 R_t 的值；

(5)保留全局最优个体：比较 $BEST_t$ 和全局最优个体 *opt*，将适应值高者保留为 *opt*；由于 *opt* 是全局最优个体，因而该方法叫做“最优保留”；

(6) 选择：对种群 X_t+Z_t 进行选择，产生下一代种群 X_{t+1}；由于 X_t 是父代群体，所以称为“父代参与选择”；

(7)检验停止准则：若满足停止准则，则停止；否则置 t=t+1；转(2)；

(8)得到最优倒箱成本。

3.3.3 编码和解码设计

编码采用了自然数编码的方式，每个个体 $G\{g_1, g_2 \ldots g_n\}$ 由 n 个整数组成，而 G 的第 i 个成员 g_i 代表一个序号。对外层 GA 来讲，就表示 D_i 的第 g_i 个成员，而对内层 GA 来讲，它表示 P_i 的第 g_i 个成员 D_i。所以这是一种间接编码方式，它把 d_{ij} 在 D_i 或者 p_{ij} 在 P_i 中的序号保留在基因中，而不是他们的值。

解码过程即把 G 转换为可行解的过程。对外层算法来说，可行解为 X，如图 3.7 所示，g_i 表示 D_i 中第 g_i 个成员 d_{i,g_i} 被选择作为待倒箱 i 的目标位置，同时 d_{i,g_i} 添加到 X 中成为其第 i 个成员。

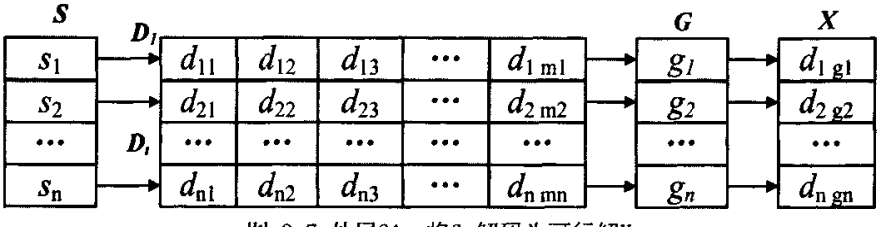


图 3.7 外层GA：将G 解码为可行解X

Fig. 3.7 Outer GA: decoding G into candidate solution X

类似的，对于内层算法来说，可行解为 U，如图 3.8 所示，g_i 表示 POS_i 的成员在 P_i 中的序号，意指 $p_{i\,gi}$ 被选择作为连接原位置和目标位置的路径中的一个节点，并作为第 i 个成员加入到 U 中。

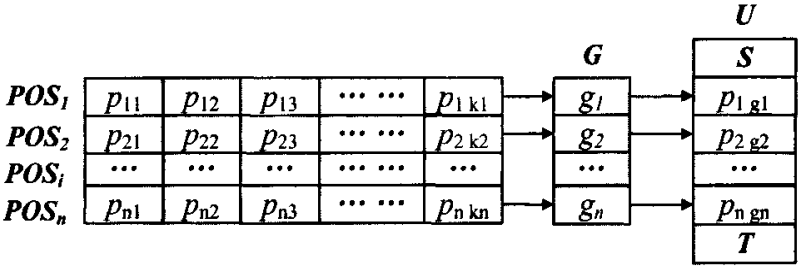


图 3.8 内层GA：将G 解码为可行解U

Fig. 3.8 Inner GA: decoding G into candidate solution U

3.3.4 适应度函数设计

遗传算法的适应度函数与提箱问题模型中的目标函数，即公式 (3.2)、(3.3) 相关，外层模型和内层模型的目标函数都是求最小值问题，适应度函数采用设定最大值 C_{max} 的方法，由目标函数转换得到。因此外层倒箱优化遗传算法的适应度函数如公式 (3.5) 所示。

$$Fit1(C)=c_{max}-C, C<c_{max} \tag{3.5}$$

式中 C_{max} 为倒箱成本值 C 的最大值估计，其值通过运行算法时的参数设定来确定。

内层最短路搜索遗传算法如公式（3.6）所示：

$$Fit2(v)=V_{max}-v, v<V_{max} \tag{3.6}$$

其中 V_{max} 为倒箱移动路径成本 H_i 或空车返回成本 R_i 的最大估计值，通过运行算法时的参数设定来确定。

3.4 实例分析

本实例用于验证算法对本问题的正确性和有效性。3 种作业设备：正面吊（**FMC**）、叉车（**FL**）和场桥（**TC**）的作业成本权值如表 3.1 所列。

堆场初始状态如表 3.2 所示，该街定义为 5 个贝，4 行、4 层。***Txx*** 表示集装箱箱号，表中没有箱号的位置是堆场空位，***T30*** (7,1,2) 是待提集装箱，业务需求期望的算法运行时间是不超过 1.5 秒。

表 3.1 不同设备的作业成本权值

Tab. 3.1 Operation cost weight for different facilities

设备		成本权值			
类型	状态	w1	w2	w3	w4
FMC	带箱	15	13	10	6
	空车	10	8	0	0
FL	带箱	16	14	8	5
	空车	10	8	0	0
TC	带箱	20	11	5	3
	空车	15	6	0	0

表 3.2 堆场初始状态

Tab. 3.2 Initial stacking status of a container block

行 \ 贝 / 层		1	3	5	7	9
1	4				T32	
	3	T03			T31	
	2	T02	T12	T21	T30	T38
	1	T01	T11	T20	T29	T37
2	4					
	3	T06				
	2	T05	T14	T23	T34	
	1	T04	T13	T22	T33	
3	4					
	3					
	2	T08		T25		T40
	1	T07	T15	T24	T35	T39
4	4		T19			
	3		T18	T28		
	2	T10	T17	T27		
	1	T9	T16	T26	T36	T41

表 3.3 列出了 ***NGA*** 在表 3.1 和表 3.2 所列初始状态下优化提取集装箱 ***T30*** (7,1,2) 的作业的优化结果，倒箱范围为 1、3、5、7 和 9 贝中的所有空位。

3.3 优化结果

Tab. 3.3 Optimization results

设备	FMC	FL	TC
n, *m* 值	*n*=3, *m*=5	*n*=6, *m*=7	*n*=2, *m*=14
Pc/Pm	0.3/0.02	0.3/0.02	0.3/0.02
popsize (outer/inner)	20/10	20/10	20/10
待倒箱及其目标位置	*T36*: (9, 4, 2) *T32*: (7, 3, 2) *T31*: (5, 2, 3)	*T36*: (9, 4, 2) *T35*: (9, 4, 3) *T34*: (5, 4, 4) *T33*: (9, 4, 4) *T32*: (1, 4, 3) *T31*: (1, 4, 4)	*T32*: (7, 2, 3) *T31*: (7, 2, 4)
NGA 成本	370	842	244
SGA 成本	378	853	256

从表 3.3 可知：***NGA*** 得到的优化结果要好于 ***SGA*** 得到的优化结果，究其原因在于：***NGA*** 可以搜索得到最优倒箱策略和最短路径，但 ***SGA*** 却不一定收敛到最优解。因此 ***NGA*** 对于求解提箱作业计划问题的适用性得到验证。

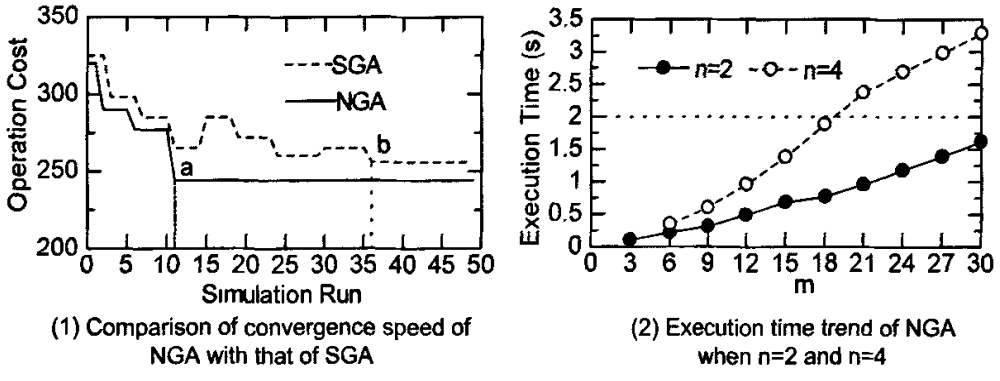


图 3.9 NGA的收敛速度和运行时间

Fig. 3.9 Convergence speed and execution time of NGA

我们对 ***NGA*** 和 ***SGA*** 随着仿真运行的收敛过程进行了比较，如图 3.9(1)，图中曲线表示以 **TC** 为作业设备时，最小作业成本的变化趋势。由图中可以看出：***NGA*** 的收敛速度要快于 ***SGA***，***NGA*** 在遗传代数为 11 时达到成熟收敛(图中 a 点)，但是 ***SGA*** 在遗传代数为 36 时收敛到次优解(图中 b 点)。在 ***SGA*** 的收敛过程中，存在最优解的退化现象(图

中区间 15-20，30-15)，但 ***NGA*** 不会出现这种现象。原因在于 ***NGA*** 中引入了父代参与选择和最优个体保留两种改进方法。

因此，***NGA*** 相对于 ***SGA*** 有两个优点：

(1) 保证收敛到全局最优解；

(2) 收敛速度较快。

为分析算法的执行效率，图 3.9(2)描述了算法的执行时间和参数 m 和 n 之间的关系，两条曲线分别表示在 n=2 和 n=4 的情况下，采用场桥作业，m=3, 6, ... , 30 时的算法执行时间趋势。由图可知：当 n=2 时，对于所有 m，算法的运行时间都少于 2 秒；当 n=4 时，算法在 m<=18 时少于 2 秒。在 m 值相同时，n=4 时的曲线斜率要大于 n=2 时的曲线斜率。

进一步的分析可知：问题的搜索空间是 m^n，是 n 的指数函数，但外层 GA 调用内层 GA 的次数为 $2n \times popsize \times generation$，近似于 n 的线性函数。因此，NGA 的搜索效率是指数函数和线性函数的综合结果，而 *generation* 是由算法的收敛速度决定的，所以，当 GA 的控制参数得到合理设置时，***NGA*** 无需搜索所有节点，具有较高的搜索效率。

4 集装箱堆场大门作业计划系统的设计与实现

前两章分别对集装箱堆场的箱位分配作业计划和提箱作业计划进行了分析，并提出了基于知识的箱位分配计划方法和提箱作业优化的嵌套遗传算法。本章采用 COM 组件技术开发实现了以上两个算法，并建立了堆场数据库，设计实现了堆场作业计划系统，调用算法 COM 组件，完成大门处对堆场作业计划功能。

4.1 系统分析与设计

4.1.1 系统分析

大门子系统为堆场管理系统操作层的一个子功能模块，如图 4.1 所示。大门的通过速度在一定程度上约束了场站的作业效率，若速度较慢不但容易造成车辆拥挤，交通不畅，而且场地作业也容易处于等待状态，降低场站整提作业效率。

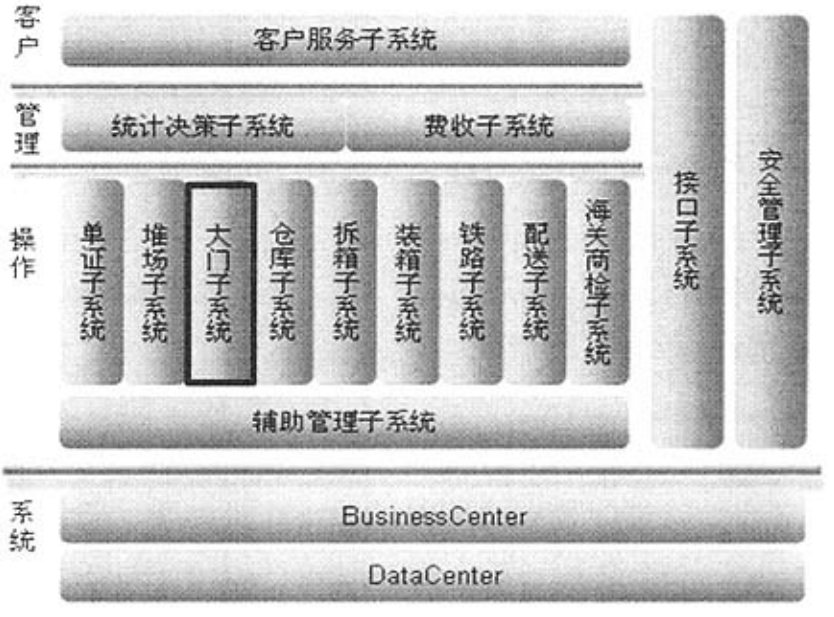


图 4.1 总系统架构

Fig 6.1 General system architecture

箱位分配作业计划的作业流程为：① 集卡到达大门，大门记录集装箱属性信息；② 大门侧的堆场作业计划系统调用箱位分配作业计划算法，得到最优目标箱位；③ 系统打印操作条，交给集卡司机；④ 集卡司机按操作条指示送箱到指定位置；⑤ 堆场设备操作员按操作条指示进行放箱作业，把目标箱放到堆场中；⑥ 集卡离开堆场。

提箱作业计划的作业流程为：① 空载集卡到达大门，向大门提交提箱订单；② 大门根据提箱订单为集卡提供提箱作业计划；③ 系统打印操作条，交给集卡司机；④集卡根据大门提供的提箱作业计划进场提箱。⑤ 堆场设备操作员按操作条指示进行提箱和倒箱作业；⑥ 集卡提取集装箱后，离开堆场。

4.1.2 系统设计

(1) 总体结构

系统的总体结构如图 4.2 所示，包括提箱作业计划、箱位分配作业计划、堆场数据库、用户界面等部分。提箱作业计划执行提箱优化算法，箱位分配作业计划执行箱位分配算法；堆场数据库储存堆场的堆存状态、设备作业能力和成本等信息，为算法初始化提供参数，并接收算法执行后返回的结果，修改堆场状态；用户界面贯穿以上三者之中。

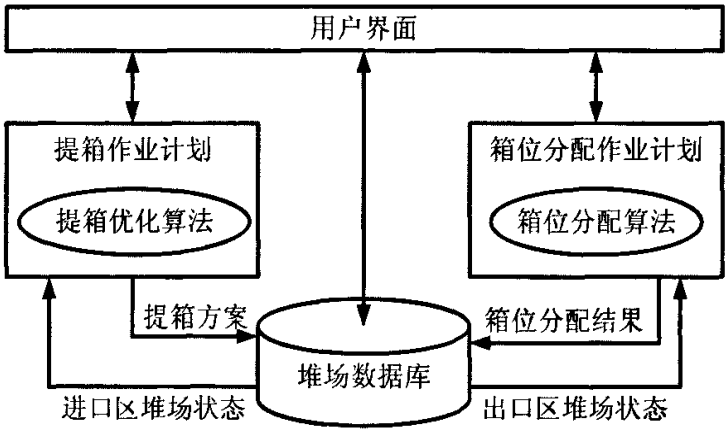


图 4.2 系统总体结构

Fig. 4.2 General architecture of system

(2) 功能模块

系统的功能模块如图 4.3 所示，分为数据库管理、大门作业计划、基于仿真的性能测试和显示堆场。数据库管理分为添加信息、修改信息、删除信息、信息查询等功能，维护堆场数据库中的相关信息；大门作业计划是系统的核心功能模块；基于仿真的系统性能测试分为仿真试验、仿真模型、仿真运行、仿真结果分析等模块，其相关研究将在

下一章阐述；显示堆场包括堆场布局（俯视图）和贝切图两个子模块，方便用户直观地获取堆场的堆存状态信息。

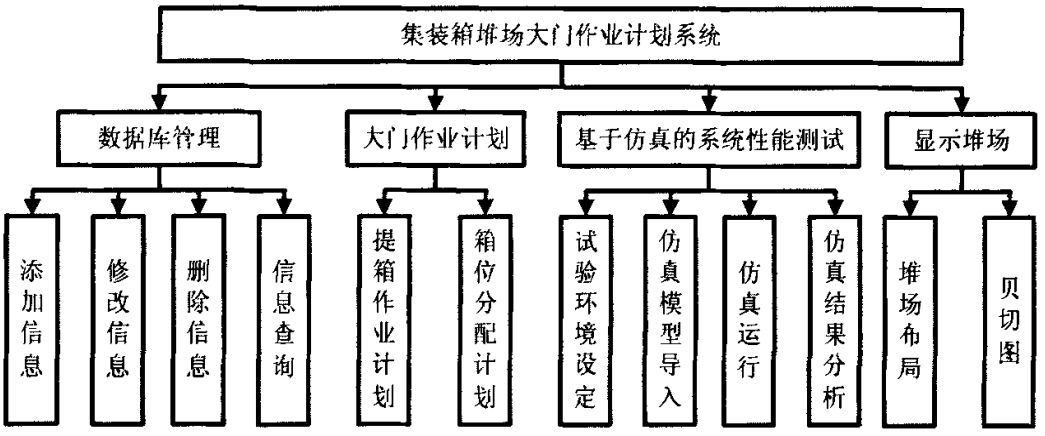


图 4.3 集装箱堆场大门作业计划系统

Fig. 4.3 Function modules of schedule system in Container Yard Gate

大门作业计划分为提箱作业计划和箱位分配计划，这两个模块的作业计划都是针对提取或分配一个集装箱的作业计划。对提箱作业计划而言，系统输入待提箱的箱号和倒箱范围，作业计划模块调用提箱优化算法，返回相应倒箱策略；对箱位分配作业计划而言，系统输入集装箱相关属性，系统调用箱位分配作业计划算法，返回给集装箱分配的位置。图 4.4 和图 4.5 分别为箱位分配作业计划和提箱作业计划模块的用户界面。

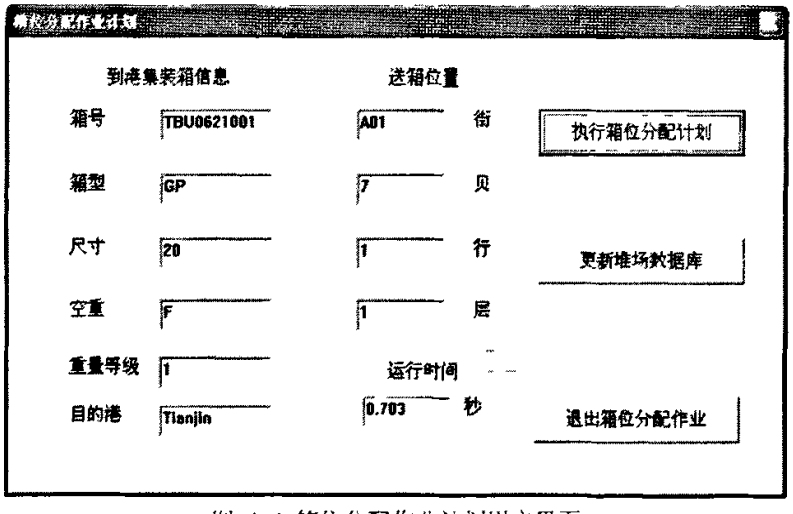


图 4.4 箱位分配作业计划用户界面

Fig. 4.4 User's interface of Container allocation schedule

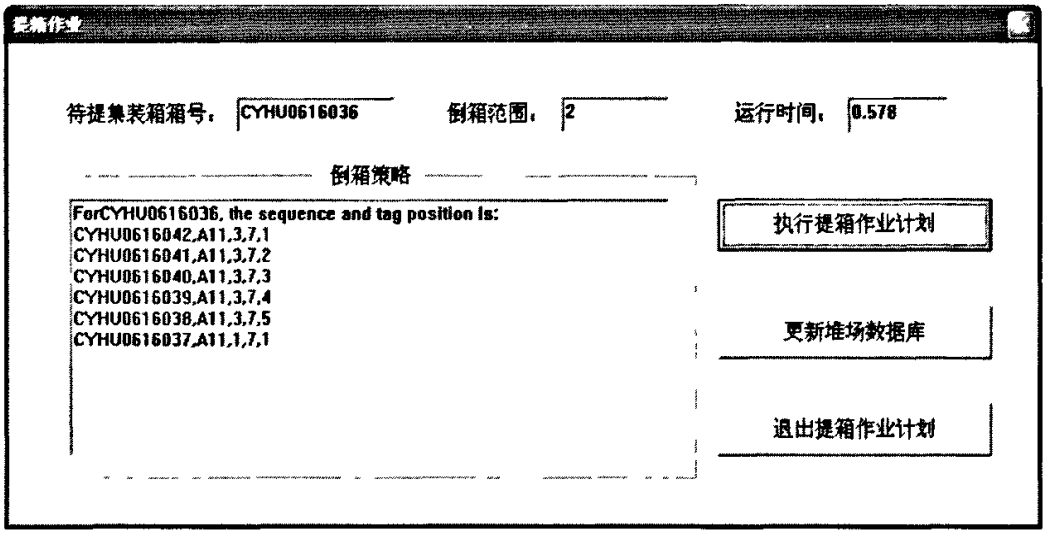


图 4.5 提箱作业计划用户界面

Fig. 4.5 User's interface of Container pick-up schedule

(3) 数据库设计

数据库中存储了执行堆场作业计划需要的所有信息，包括：分组规则，堆场街定义、贝定义、堆场中的在场箱、设备信息、设备作业能力、设备作业成本、设备作业模板。数据库中所有表的设计见附录 A。

(4) 接口设计

系统接口有两种：算法接口和数据库接口。

算法接口包括输入输出接口和功能接口，输入接口接受算法执行所需要的外部参数，输出接口输出算法运行的结果，输入参数和返回结果均以字符串的形式传递；功能接口调用运行算法。

数据库接口完成两种功能：一是读取数据库，拼接成符合算法输入接口格式的字符串参数，二是接收算法返回的结果，写回数据库。

表 4.1 是箱位分配作业算法的接口说明。

表 4.2 是提箱作业算法的接口说明。

表 4.3 是数据库接口说明。

表 4.1 箱位分配作业算法的接口说明

Tab. 4.1 Illustration of interface of Container allocation schedule algorithm

接口名称	功能说明
string addContainer(string strCon)	添加需要放的箱
string addBayInfo(string strBay)	添加计划贝定义基本信息
string addStack(string strDetail)	添加堆场已放箱和各位置放置状态
string addPosition(string strPlanPos)	添加计划位置
string addFacility(string strFac)	添加设备信息
string addAbility(string strAbli)	添加设备作业能力
string addTemplet(string strTemplet)	添加设备作业模板
string addData()	初始化以上所有数据区
void run()	运行箱位分配主算法
void reset()	清空算法所有数据结构
string getCon()	返回箱位分配结果

表 4.2 提箱作业算法的接口说明

Tab. 4.1 Illustration of interface of Container pick-up schedule algorithm

接口名称	功能说明
string addBlock(string strBlock)	添加街定义信息
addOrder(strin strOrder)	添加待提箱信息
string addBayInfo(string strBay)	添加贝定义信息
string addStack(string strDetail)	添加堆场已放箱和各位置放置状态
string addFacility(string strFac)	添加设备信息
string addAbility(string strAbli)	添加设备作业能力
string addFacCost(string strCost)	添加设备作业成本
string specifyInit ()	指定提箱作业初始化
string unSpecifyInit ()	不指定提箱作业初始化
void run()	运行提箱作业主算法
void reset()	清空算法所有数据结构
string getRehandleStra()	返回提箱作业方案(倒箱策略)

表 4.3 数据库接口

Tab. 4.3 Interface of database

接口名称	功能说明
allocRead()	根据界面输入，从数据库中读取信息，转换为箱位分配算法参数
allocWrite()	接收箱位分配算法返回结果，写回数据库
handUpRead()	根据界面输入，从数据库中读取信息，转换为提箱算法参数
handupWrite()	接收提箱算法返回结果，写回数据库

4.2 系统实现

本系统以 Visual C++ 6.0 为开发工具，在 Windows 平台下开发实现。

4.2.1 算法的实现

箱位分配算法和提箱作业算法采用组件技术，在 Visual C++ 下用 ATL COM 开发实现。

组件实际上是一些小的二进制可执行程序，它们可以给应用程序，操作系统以及其他组件提供服务。COM（Component Object Model，组件对象模型）是微软公司为了计算机工业的软件生产更加符合人类的行为方式开发的一种新的软件开发技术，在 COM 构架下，人们可以开发出各种各样的功能专一的组件，然后将它们按照需要组合起来，构成复杂的应用系统。其优点是多方面的：

(1) 可以将系统中的组件用新的替换掉，以便随时进行系统的升级和定制；

(2) 可以在多个应用系统中重复利用同一个组件；

(3) 可以方便的将应用系统扩展到网络环境下；

(4) COM 与语言、平台无关的特性使所有的程序员均可充分发挥自己的才智与专长编写组件模块。

ATL（Active Template Library，活动模板库），其设计旨在让人们用 C++方便灵活地开发 COM 对象。外部系统把用初始信息初始化算法必须的类，传递给算法类的构造函数；然后调用算法类的执行程序执行算法，返回外部系统所要的结果；外部应对返回结果进行解析，得到所要数据。此方案产生可复用的二进制代码，不用任何附加的运行时 DLLs 支持，且便于以后分布式功能的需要。

每个 COM 对象都提供一个名叫 IUnknown 的接口，该接口包含方法 AddRef()、Release()和 QueryInterface()。在实现箱位分配算法和提箱作业算法时，我们加入一个 ATL 的简单对象（Simple Object），该对象提供对 IUnknown 和 IDispatch 的预制支持，可以是可聚集或不可聚集，并且可以使用任何线程模型。

在 ATL 对象中，可以定义属性和方法，下面以箱位分配算法的 ATL 对象定义为例说明。

```
class ATL_NO_VTABLE CAutoAllocDll :
        public CComObjectRootEx<CComSingleThreadModel>,
        public CComCoClass<CAutoAllocDll, &CLSID_AutoAllocDll>,
        public IDispatchImpl<IAutoAllocDll, &IID_IAutoAllocDll,
&LIBID_COMAUTOALLOCLib>
{
```

```
public:
    CAutoAllocDll()    {   }
DECLARE_REGISTRY_RESOURCEID(IDR_AUTOALLOCDLL)
DECLARE_PROTECT_FINAL_CONSTRUCT()
BEGIN_COM_MAP(CAutoAllocDll)
    COM_INTERFACE_ENTRY(IAutoAllocDll)
    COM_INTERFACE_ENTRY(IDispatch)
END_COM_MAP()
// IAutoAllocDll，对象中定义的方法
public:
    STDMETHOD(getConAmt_dll)(/*[out,retval]*/ long *rs);
    STDMETHOD(reset_dll)();
    STDMETHOD(getSecondCon_dll)(/*[out,retval]*/BSTR *rs);
    STDMETHOD(getFirstCon_dll)(/*[out,retval]*/BSTR *rs);
    STDMETHOD(getSecondErr_dll)(/*[out,retval]*/BSTR *rs);
    STDMETHOD(getFirstErr_dll)(/*[out,retval]*/BSTR *rs);
    STDMETHOD(run_dll)();
    STDMETHOD(addData_dll)(/*[in]*/BSTR strCon,
                           /*[in]*/BSTR strBay,
                           /*[in]*/BSTR strStack,
                           /*[in]*/BSTR strPlanPos,
                           /*[in]*/BSTR strFac,
                           /*[in]*/BSTR strTemplet,
                           /*[in]*/BSTR strAbili,
                           /*[out,retval]*/BSTR *rs);
// IAutoAllocDll，对象中定义的属性
private:
    SAutoAlloc sAlloc;      //箱位分配算法主类对象
};
```

该类中定义的接口与箱位分配算法主类 SAutoAlloc 中（表 4.1）的接口一一对应，在实现时正是调用了相应的主类对象接口。

4.2.2 数据库及其接口的实现

(1) 数据库和数据库的连接

数据库选择的是 Microsoft Access 2003，这是一项全面完整的数据库与分析产品，它全面支持 Web 功能，在可扩展性与可靠性方面性能卓越，基于 Access 2003 的应用程序开发速度和事务处理运行速度均相当快捷。Microsoft Access 2003 因其在稳定性、速度、安全性等方面的优点而得到广泛使用。

数据库连接使用 ODBC (Open DataBase Connection，开放式数据库互连)技术。ODBC 是指开放服务结构中有关数据库的一个组成部分，它建立了一个规范，并提供了一组对数据库的标准 API（应用程序编程接口）。使用 ODBC API 编程，有一套专门的方法，称为 ODBC API 编程模型，主要有以下步骤：

① 为 ODBC 分配环境句柄；

② 分配一个连接句柄；

③ 连接到数据库；

④ 用 SQL 命令分配一个语句句柄；

⑤ 执行该命令返回结果集；

⑥ 断开通数据源的连接；

⑦ 释放 ODBC 环境；

使用 ODBC API 的时候，ODBC 管理程序把数据库访问的请求传递给正确的驱动程序，驱动程序在使用 SQL 语句指示 DBMS 完成数据库访问的工作，因此，ODBC 的存在为开发应用数据库程序提供了非常强大的功能和灵活性。

ODBC 与数据库一起工作，必须把数据库注册到 ODBC 驱动程序管理器，这通过定义一个 DSN 来完成。一个 DSN 信息至少应该包括：数据库文件名，系统，文件夹等信息，同时要给数据源命名。使用 ODBC 访问数据库必须声明这个注册的 DSN。

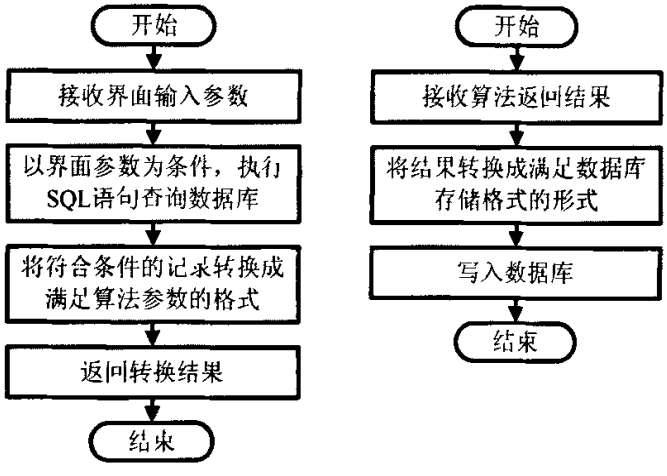
(2) 数据库接口

数据库接口封装在一个类 DBInterface 中，该类定义如下：

```
class CReadData   {
    public:
        CReadData();                    //构造函数
        virtual ~CReadData();           //析构函数
        int allocRead();                //箱位分配算法读数据库接口
        void allocWrite();              //箱位分配算法写数据库接口
        int handUpRead();               //提箱算法读数据库接口
        void handUpWrite();             //提箱算法写数据库接口
        string get~();                  //私有变量读取接口
```

```
    void set~(string);          //私有变量写入接口
private:
    string strCon;              //待分配集装箱（箱位分配算法使用）
    string strBay;              //贝定义（两个算法共同使用）
    string strStack;            //堆场在场箱（两个算法共同使用）
    string strPos;              //计划位置（箱位分配算法使用）
    string strFac;              //设备（两个算法共同使用）
    string strAbl;              //设备作业能力（两个算法共同使用）
    string strTempt;            //设备作业模板（箱位分配算法使用）
    string strOrder;            //待提箱订单（提箱算法使用）
    string strRange;            //倒箱范围（提箱算法使用）
    string strBlock;            //街定义（提箱算法使用）
    string strCost;             //设备作业成本（提箱算法使用）
};
```

读写数据库接口的实现原理如图 4.6 所示，这是两个互逆的过程。



读数据库接口实现流程　　写数据库接口实现流程

图 4.6 数据库接口实现原理

Fig. 4.6 Implementation of database interface

5 基于仿真的集装箱堆场大门作业计划系统性能测试

集装箱堆场大门作业计划系统是一个实时作业计划系统，对箱位分配算法和提箱优化算法的实时响应速度有较高要求。该系统测试需要收集、整理大量的测试数据，分析测试结果，工作量相当大。本文将离散事件系统仿真和软件测试相结合，提出了基于仿真的信息系统性能测试框架，对集装箱堆场大门作业计划系统进行性能测试。

5.1 基于离散事件仿真的信息系统性能测试

5.1.1 基于仿真的信息系统性能测试

性能测试不仅能够验证系统是否满足响应时间、吞吐量等性能指标的要求，还可分析系统可能存在的瓶颈。对于这类信息系统而言，性能测试很重要，但又存在以下困难：

(1) 用于性能测试的数据应覆盖系统实际运行的所有状态，以确保系统运行的稳定性和可靠性，因此既需要表征系统平均工作量的数据，也要表征系统极大工作量的数据，而采用一般的测试方法搜集和输入数据的工作量过大。

(2) 一次输入顾客服务特征参数和系统状态，只能考察系统某一时刻或某一状态下的运行性能；若要考察系统在长期连续运行下的性能，则需要记录每次运行的性能参数，观察其统计特性。

离散事件系统仿真和软件测试的结合有较大应用价值[60]，文献[61]基于排队模型的仿真建模技术，提出了一种新的性能测试框架和一种仿真模型参数估计算法；文献[62]分析了测试实时软件的三种基本方法，论证了硬件在回路仿真测试平台解决方案在解决测试实时软件问题中的优势。另外，仿真在军事[63]、电子[64][65]、通信[66]等领域的软件测试中也有相关应用，多是用于随机发生测试数据，而仿真统计分析等其它功能的利用，有待进一步完善。

5.1.2 测试框架

本文提出一种基于离散事件仿真的信息系统的性能测试方法，把待测信息系统融入离散事件仿真系统，利用仿真产生测试数据，并通过仿真的统计分析功能，对系统在连续运行下的性能进行评价。测试框架总体结构如图 5.1 所示。

测试数据和实体到达分布参数可由历史数据中提取相应的特征而得到，也可根据测试目标生成。在实体产生时刻，测试数据以实体属性的形式随之产生，待实体得到服务时，传入被测系统。待测系统通过 DB 接口获取系统状态参数，作出领域相关的服务决策，同时写入数据库，运行结束后返回运行时间给仿真系统，仿真系统继续运行。

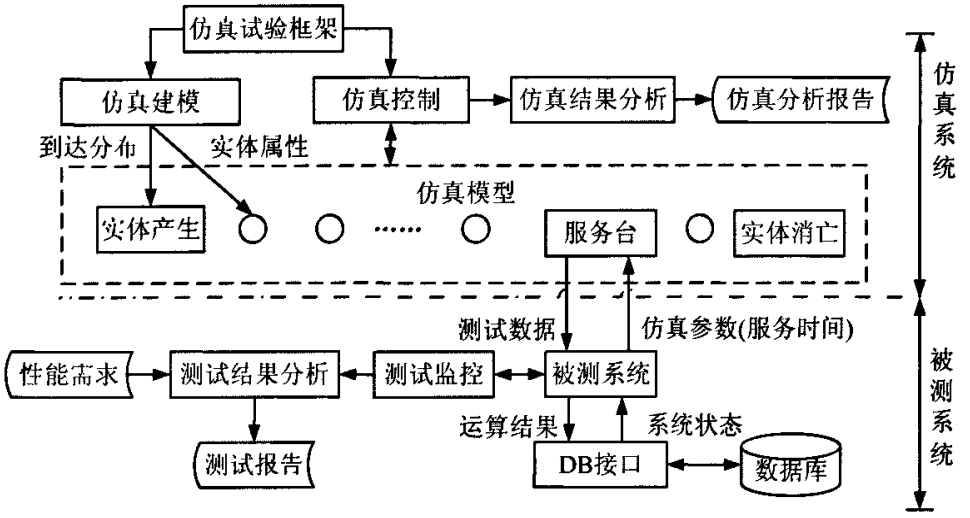


图 5.1 基于仿真的测试总体结构

Fig. 5.1 General architecture of Simulation-based testing

该框架形成了虚实结合的结构，一方面，仿真系统中原本为模拟值的服务时间，在该框架下通过实时统计被测系统运行时间得到，因此该参量由真实值代替模拟值。另一方面，被测系统的数据输入原本由人工在系统外部进行，该框架下由仿真系统完成，因此由虚拟系统代替真实系统。

5.2 仿真系统分析与设计

系统仿真可分为连续系统仿真和离散系统仿真，连续系统的内部状态随时间连续变化，任意两个时间点上系统的内部状态都不相同，因此必须确定每个时刻系统的内部状态，解出系统的内部状态与时间的函数关系。离散系统的内部状态变化是随机的，同一个内部状态可以向多种状态转变，因此很难用函数形式来描述系统内部状态的变化，通常所关心的是系统内部状态变化的统计规律。

5.2.1 仿真系统总体结构

仿真系统以离散事件系统仿真原理设计，包括仿真试验框架、仿真建模、仿真控制、仿真结果分析等模块。

(1) 仿真试验框架。该模块定义了仿真运行的次数、每次仿真运行的时间等仿真试验的参数，它负责设定仿真运行的初始状态和终止条件。

(2) 仿真建模。系统仿真的初始阶段，该阶段从研究领域中辨识实体、资源等，以仿真模型的形式描述系统运行的过程，研究模型中的重要参数。

(3) 仿真控制。仿真控制管理事件表和活动表，实现仿真运行的事件调度和活动安排，该模块主要包括 2 个子模块：时钟管理和条件检查，前者负责寻找将来最早发生的事件或者活动，推进仿真时钟；后者负责管理条件活动和实体等待的解除。

(4) 仿真结果分析。该模块负责统计并计算仿真过程中系统的统计指标，如系统平均队长、最大队长、实体的平均等待时间、服务台的利用率等。在统计的基础上，分析这些统计量的统计特性。

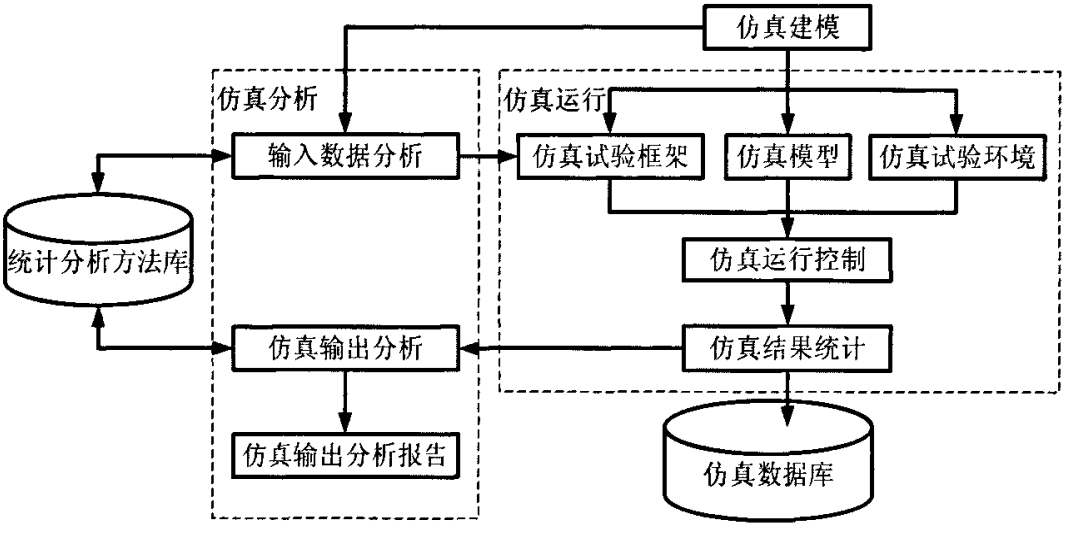


图 5.2 仿真系统的整体结构

Fig. 5.2 Infrastructure of simulation system

5.2.2 仿真运行控制

首先，我们介绍仿真运行控制的基础，主要包括：事件、实体和活动。事实上，仿真运行控制在本质上是对以上三者的控制。

(1) 事件和事件表

事件说明仿真中的某种事件在某时刻发生，每个事件所带的信息为：事件发生时间和事件发生目标。事件表是事件对象的集合，它能使流程管理以正确的顺序处理事件，计划事件和撤销事件由事件管理负责，通过事件表完成。

(2) 实体和实体表

实体是系统中的活动部件，活动由实体的产生而引起，当系统中的实体全部消亡后，系统活动也将随之停止。实体表是实体对象的集合，保存了有关实体的所有信息，包括实体号、产生时间、优先级等。当实体产生时，实体信息插入实体表；当实体经历活动时，实体表就在不断地被修改；当实体消亡时，表中相应实体信息被删除。

(3) 活动和活动表

活动表明实体活动的状态或者活动将要发生的时间，仿真过程是实体从一个站到另一个站移动的过程，活动就描述了实体在它的生命周期不同时刻的状态。实体的状态分为三种：激活、预期和等待。激活是活动状态，而预期和等待均为非活动状态。活动表记录了所有实体活动，它和实体表相对应，即表中序号为 i 的活动表示实体表中第 i 个实体的活动状态。

仿真控制是在仿真试验框架下对仿真运行过程的控制，是系统仿真的核心。仿真试验中设定了仿真运行的次数，每次仿真运行又有不同的环境变量，因而仿真控制可分为多次仿真控制和单次仿真控制，其运行流程如图 5.3 所示。

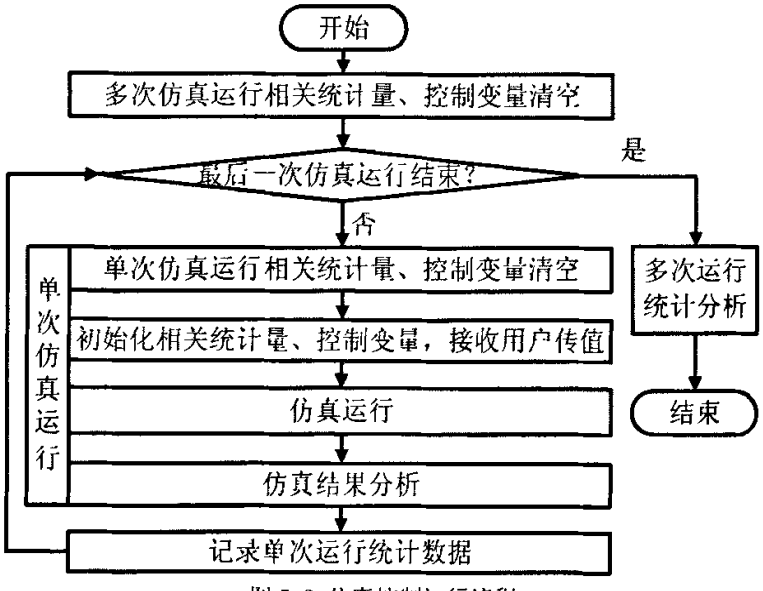


图 5.3 仿真控制运行流程

Fig. 5.3 Flow chart of simulation control

可见，所谓多次仿真就是反复运行单次仿真的流程，并记录每次仿真的统计数据，待仿真结束后进行统一的多次仿真统计分析。因而仿真控制的关键就在于每次仿真的流程控制，确切地说，是图中“仿真运行”环节。

仿真运行控制主要包括两大部分：安排用户事件和流程管理。用户事件是指用户在仿真开始之前就可以确定发生时间和发生目标的事件，往往是初始实体的产生事件，对仿真状态造成较大改变的关键事件等。

流程管理就是仿真策略的实现，仿真系统采用近程交互策略实现流程管理，进程交互法结合了事件调度和活动扫描，因而流程管理主要包括两大部分：时钟管理和条件检查。时钟管理的功能是寻找下一个事件或预期的活动实体，扫描预期活动，将仿真时间推进到下一最早发生的事件或活动。条件检查的功能是寻找阻塞实体，检测实体激活的条件是否满足，若满足，则选择相应策略，激活实体，执行相应活动。仿真运行流程图如图 5.4 所示。

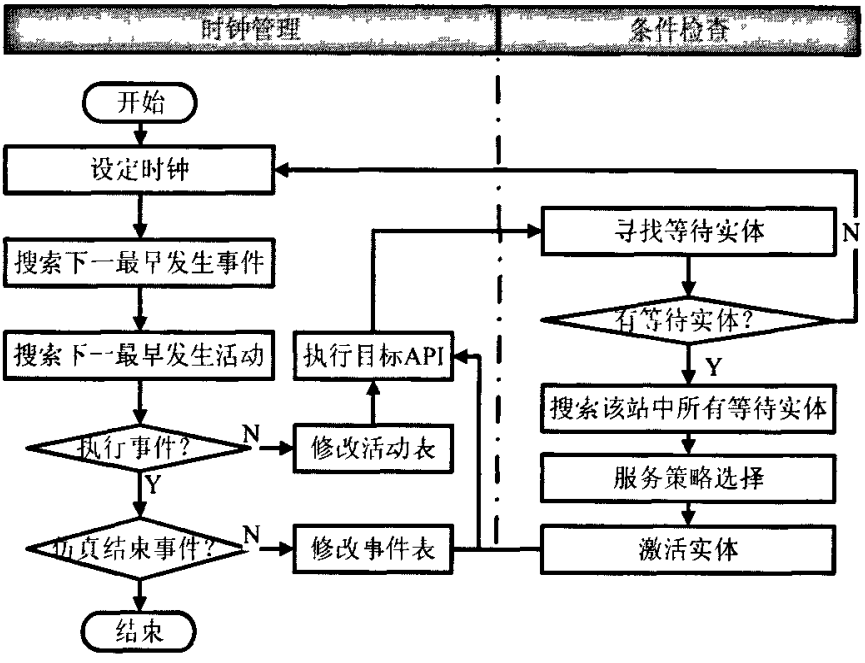


图 5.4 仿真运行流程管理

Fig. 5.4 Flow management of simulation run

在仿真系统调用测试系统，执行仿真测试功能的仿真运行控制流程如图 5.5 所示。

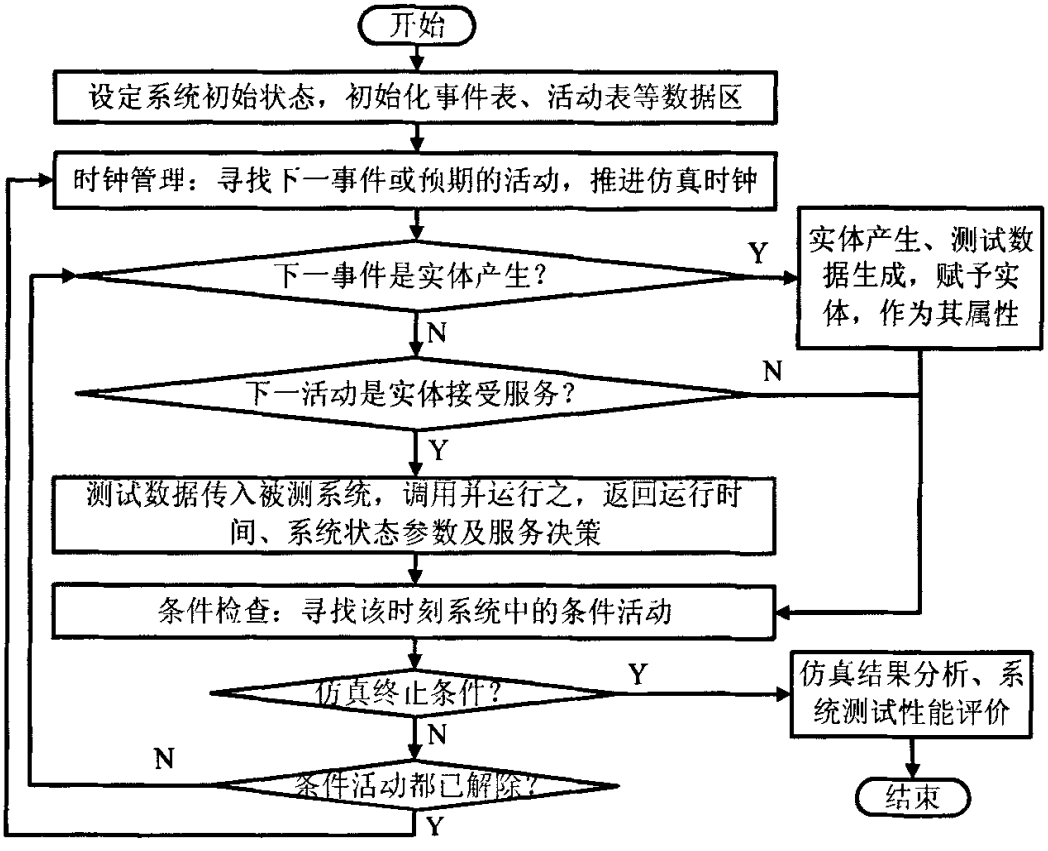


图 5.5 执行仿真测试功能的控制运行流程

Fig. 5.5 Flow chart of simulation run with testing embeded

5.2.3 用户接口及其实现方法

主要的用户接口包括 userEvent()、simModel()和 simAnalysis()。

(1) userEvent()

userEvent()用于安排用户事件，实现该接口只要调用 API 接口 event()，指定事件发生的时间和目标。用户事件往往是安排第一个实体的产生，由于实体产生由源指定，因此用户在安排实体产生事件时需要指定相应源号。因此 event()有 3 个参数：事件时间、事件目标、源号。

(2) simModel()

simModel()给用户仿真建模，仿真系统采用近程交互法进行仿真运行控制，建立仿真模型只需用建模资源类的功能 API 描述进程，实现 simModel()接口。仿真中的进程由

一系列事件和活动构成，在仿真模型中它们通过相应的功能 API 实现。每个 API 均有一个目标代码，作为 API 接口的唯一标识，通过该标识，事件，活动的地址被保存，仿真控制进行目标选择时，就可以找到相应的 API，处理相应的活动。

(3) simAnalysis()

simAnalysis()用于仿真结果分析，用户根据自身需要定义统计量，这些统计量相关的基础数据的获得，是通过在仿真模型的产生数据的 API 前后分别调用仿真数据统计类的 arrive()和 depart()接口而实现的。在 simAnalysis()中，用户调用 endBin()，结束统计工作，而后利用统计类中的基础数据求得统计量。

5.3 基于仿真的大门作业计划系统性能测试

5.3.1 港口大门作业模型

集装箱港口大门作业模型如图 5.6 所示。图中左侧为入口通道排队模型，右侧为出口通道排队模型。

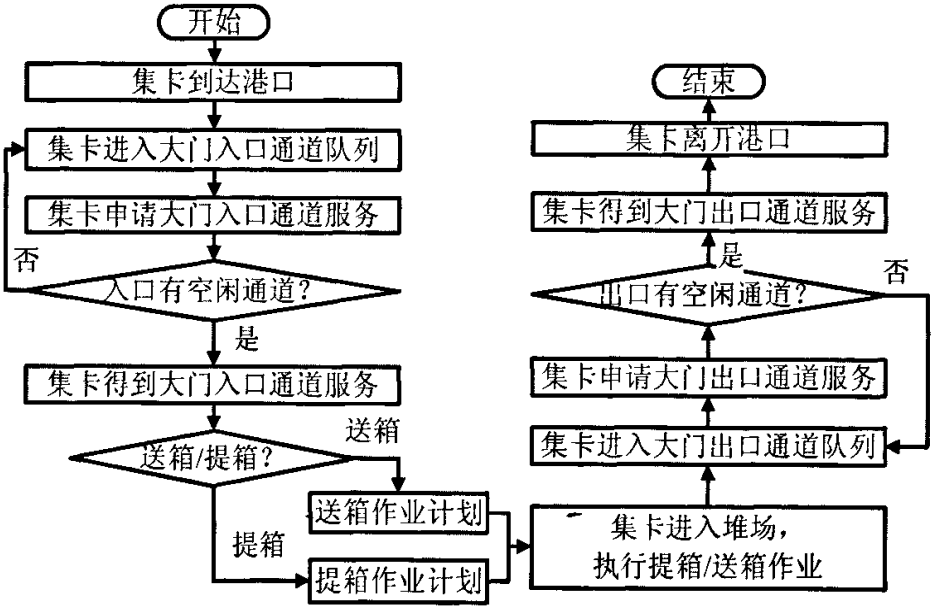


图 5.6 港口大门作业模型

Fig. 5.6 Operation model of container port gate

图 5.6 所示的港口大门作业模型描述如下：

(1) 集卡随机到达港口大门，随后进入大门入口通道队列，向入口通道申请服务。

若入口没有空闲通道，则集卡返回等待队列，下一次继续申请服务；若入口有空闲通道，则集卡得到服务。

(2) 到港集卡分为两类：空车和重车，空车向港口发出提箱作业请求，重车发出进箱作业请求，大门服务台在服务阶段根据集卡类型执行相应作业计划。

(3) 大门入口通道服务结束后，集卡进入堆场，按照大门作业计划给定的指令，执行提箱/进箱作业。

(4) 作业过程结束后，集卡准备离开港口，进入大门出口通道对列，向出口通道申请服务；若出口没有空闲通道，则集卡返回等待队列，下一次继续申请服务；若出口有空闲通道，则集卡得到服务。

(5) 在出口处没有作业计划，大门对集卡的服务无需根据集卡类型加以区分，服务结束后，集卡离开港口。

由此可见，入口通道和出口通道分别是一个单队列多服务台的排队模型，所以港口大门作业模型是由两个单队列多服务台排队模型构成的组合模型。

5.3.2 测试方案和数据准备

本方案利用基于离散系统仿真的方法测试大门系统性能，主要指标如下：

(1) 提箱/进箱模块运行的响应时间统计特征。由于提箱和进箱模块位于港口大门入口通道处，因而对系统的响应时间有较高要求。

(2) 大门服务系统总体服务水平，主要指入口通道的平均队长、平均等待时间等。

我们以中国北方某港口的历史数据为基础，拟合了提箱集卡和进箱集卡的到达规律，集卡到达数量按一天中不同的时间段而不同。以 1 小时为采样间隔，集卡到达数量如图 5.7，分布参数如表 5.1 所示：

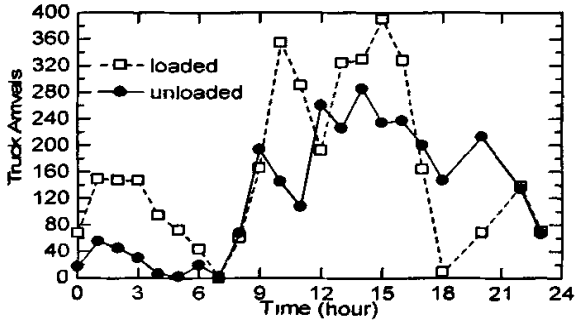


图 5.7 一天内按小时统计的到达外卡的数量分布图

Fig. 5.7 Operation model of container port gate

表 5.1 集卡到达时间间隔分布参数

Tab. 5.1 Distribution parameter of out trucks' arrival time interval

时间段	分布类型	提箱参数值	进箱参数值
0~6	负指数分布	λ=0.0083	λ=0.0361
6~12	负指数分布	λ=0.037	λ=0.0592
12~18	负指数分布	λ=0.0631	λ=0.0722
18~24	负指数分布	λ=0.0222	λ=0.0139

大门通道服务时间分布函数$f_1(t)$，$f_2(t)$，$f_3(t)$服从正态分布函数，如表 2 所示，参数μ、σ分别表示正态分布的均值及方差。其中，f_1为入口通道为提箱集卡的服务时间，f_2为入口通道为进箱集卡的服务时间，f_3为出口通道的服务时间（出口处对提箱/进箱集卡的服务近似等同）。

表 5.2 大门通道服务时间分布函数

Tab. 5.2 Distribution parameter of service time of gate

分布函数	分布类型	单位	参数	
			μ	σ
$f_1(t)$	Normal	秒	40	4
$f_2(t)$	Normal	秒	30	4
$f_3(t)$	Normal	秒	20	4

外卡在集装箱堆场的装卸时间分布采用经验分布函数$f_4(t)$，如式(5.1)所示。

$$f_4(t) = DISC((0.9,9),(0.95,21),(0.99,36),(1,60)) \tag{5.1}$$

$f_4(t)$由四组数字组成，每组由两个数字组成，分别表示累加的百分比与平均服务时间。例如：第二组数据(0.95,21)表示一辆卡车在堆场接受服务的平均时间长度为 21 分钟的概率为 5%(0.95- 0.9 = 0.05)。

根据港口大门作业模型、测试方案及整理的数据，利用离散事件仿真工具提供的 API 接口，建立港口大门作业的仿真模型，实现 userEvt()和 simModel()接口，程序代码见附录 B。

5.3.3 测试结果及分析

根据上述分布参数建立仿真模型，仿真 10 次，每次仿真周期为 1 天，对该大门作业服务系统性能进行测试，分别统计执行提箱模块和进箱模块的系统响应时间，通过仿真分析得到的统计特性分别见图 5.8 和图 5.9 所示，图中置信度均为 0.95。

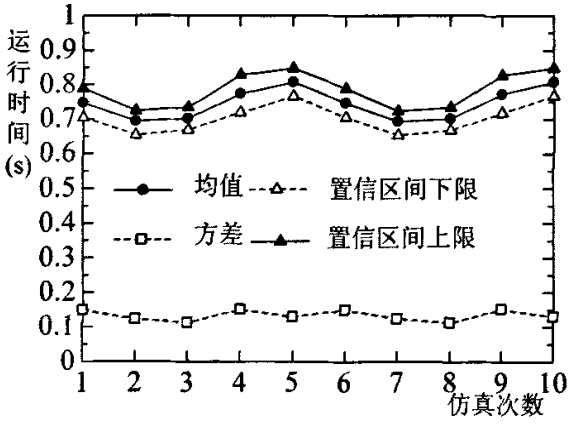
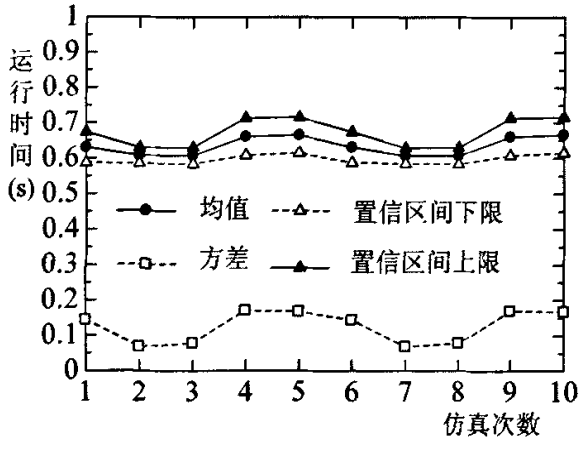


图 5.8 提箱作业运行时间统计特性

Fig. 5.8 Statistic feature of executive time of pick-up operation



5.8 进箱作业运行时间统计特性图

Fig. 5.8 Statistic feature of executive time of Storage space allocation operation

本例中大门系统取的入口通道数量为 6 个，仿真系统统计得到的大门系统服务性能指标如表 5.3 所示：

表 5.3 系统服务性能指标

Tab. 5.3 Service capability index of system

通道	MWT	MQL	MUP	TST1	TST2
1	17.8	1.7	42.7%	5694.4	859.1
2	18.1	1.9	43.2%	5656.7	842.6
3	17.6	1.6	42.1%	5673.9	855.3
4	17.9	1.8	43.3%	5688.3	866.2
5	18.0	1.7	43.7%	5675.4	849.6
6	17.6	1.7	43.2%	5681.8	869.5

MWT：平均等待时间(s)　　MQL：平均队长(个)　　MUP：平均利用率

TST1：提箱运行总时间(s)　　TST2：进箱运行总时间(s)

以上结果是从离散事件仿真系统的仿真结果分析功能得到的，因此，本文所提出的基于离散事件系统仿真的系统性能测试方法，具有如下优点：

(1) 可通过合理设计测试方案，发生有效的测试数据，覆盖算法中的程序分支，保证测试的全面性。测试方案可以由用户设计，输入分布及其参数可以根据测试需要而修改，既可以从历史数据中拟合得到输入分布，也可以由用户设计将来某一可能发生的分布，模拟系统在大量实体到达、不同堆场状态下的系统运行性能。

(2) 极大地提高了测试的效率。测试数据可以随机生成，用离散事件仿真系统这种“软件”代替人执行测试工作，加快了测试速度。

(3) 减少测试结果分析的工作量，提高测试分析结果的可信度。可以方便地得到被测系统的性能指标，对测试得到的数据，用户可以直接查看仿真系统分析得到的结果，而不需要重新分析。同时，也避免了用户分析过程中由于不了解仿真数据特点而引起的误差。

结　论

本文运用知识工程、遗传算法、系统仿真等理论和技术，对集装箱堆场作业计划中的提箱作业计划和箱位分配计划问题进行了研究，研究的主要结论如下：

(1) 针对箱位分配作业问题，通过分析集装箱箱位分配作业的规则，说明采用基于知识的方法的必要性，并构建了基于知识的箱位分配模型，该模型包括分配区域划分、模式及规则匹配、最优箱位选择 3 个部分。实例结果表明：用该方法进行的箱位分配计划，有利于堆场的后续作业，并可减少设备作业的移动距离。

(2) 针对集装箱堆场提箱作业问题，首先，将提箱作业优化问题归结为包含倒箱优化和最短路搜索两部分的多阶段决策问题，并建立了提箱作业优化的数学模型，该模型以作业总成本最小为目标函数，以作业规则函数为约束条件。其次，提出两层结构的嵌套遗传算法，外层实现倒箱优化，内层实现最短路搜索。对遗传算法的总体结构特点、编码解码方法和适应度函数设计作了重点讨论。为确保收敛到最优解，引入最优个体保留的方法；为加快算法的收敛速度，引入父代参与选择的方法。最后，通过实际算例对本算法的正确性、有效性进行验证，结果表明：嵌套遗传算法可用作提箱作业优化算法，且相对于 SGA 具有较快的收敛速度。

(3) 按照离散事件系统仿真的基本原理，以进程交互法实现仿真控制，设计开发了离散事件系统仿真工具，提供 API 接口。该软件包采用面向对象的方法设计，提供了各种仿真建模类库及其功能 API 和用户接口，实现仿真试验和仿真控制的分离。

(4) 采用基于仿真的测试方法，将箱位分配作业计划算法和提箱作业优化算法的 COM 组件嵌入仿真系统中，对上述两个算法模块进行性能测试。基于离散事件仿真的测试方法将被测信息系统融入仿真系统，测试数据由仿真系统产生，解决了测试数据的来源问题，并且可利用仿真系统的分析功能评价系统长期运行性能，不但提高了测试效率，也便于系统性能的统计分析。

在上述研究中，主要创新点在于：

(1) 将知识工程、规则推理的方法应用于集装箱堆场箱位分配作业计划；

(2) 针对提箱作业这样的具有层次性的、嵌套结构的优化问题，提出两层结构的嵌套遗传算法；

(3) 更进一步地，该算法结构可以从两层进一步扩展到 n 层，只要每一层都可以提出明确的目标函数，那么各层可以共享一个遗传算法流程，又拥有各自的适应度函数，求得最终的优化结果。

(4) 将离散事件系统仿真与软件测试相结合，提出基于离散事件仿真的系统性能测试框架，并实现了对箱位分配作业计划和提箱作业计划模块的测试，该方法也可用于对具有和港口大门作业计划系统类似特征的其他信息系统的性能测试中。

然而，本文的各项工作内容有的存在值得继续深入研究的难点，有的可以同其他相关领域结合扩展深入，均有待完善和进一步发展。

(1) 在箱位分配作业计划问题方面，首先，本文所采用的方法是将计划人员的经验、堆场、设备的固有特点结合应用，而没有采取解析模型和优化方法，因而系统的输入，也就是各种规则的定义对箱位分配的结果起着决定性的作用。换言之，对知识的获取、加工处理并不是系统可以自动实现的。同时，在推理方法、对知识的解释方面，也需要进一步完善。其次，由于船舶到港时间、装船配载的相关信息的滞后性特点，使得本方法并不能保证装船作业时不发生倒箱作业，而只是尽量少发生。

(2) 提箱作业计划问题方面，本文只考虑了提取一个箱、一台设备、在一条街内倒箱作业的情况，对更为复杂的问题，如：多个箱的提取顺序的优化，多条街到箱问题，多台设备的协调、调度问题，可以在此基础上作进一步的分析。另一方面，对于遗传算法本身，包括算法结构、参数设置、收敛性等问题，仍需进一步的讨论，以改进算法的性能。

(3) 在仿真基本功能方面，本文所设计开发的离散事件仿真工具对于一般的排队和库存问题适用，如本文中的单队列多服务台排队问题。但对于较为复杂的，实体之间、站之间存在较为复杂的逻辑关系的问题，还缺乏相应模块，需要进一步扩充和完善。

参 考 文 献

[1] http://www.asiadcp.com/club/html/2005-05/1133.htm
[2] http://data.wswire.com/htmlnews/2006/04/03/673345.htm
[3] http://www.tfqh.com/show_detail.jsp?news_id=867230
[4] http://china.cangchu.com.cn/Info/273/Index.shtml
[5] 真虹. 集装箱运输学. 人连：大连海事大学出版社. 1999.
[6] Yun, Won Young, Choi, Yong Seok. A simulation model for container terminal operation analysis using an object-oriented approach. International Journal of Production Economics. 1999, 59(1-3): 221-230.
[7] 李建忠. 集装箱港口资源配置问题研究:(博士学位论文). 上海：上海海事大学. 2005
[8] De Castilho B. Daganzo C. F. Handling strategies for import containers at marine terminals. Transportation Research. 1993. 27B (2): 151-166.
[9] Kim K H, Kim H B. Segregating space allocation models for container inventories in port container terminals. International Journal of Production Economics, 1999, 59:415-423.
[10] Preston Peter, Kozan Erhan. An approach to determine storage locations of containers at seaport terminals. Computers and Operations Research, 2001, 28 (10): 983-995.
[11] Ebru K. Bish. A multiple-crane constrained scheduling problem in a container terminal. European Journal of Operational Research, 2003, 144(1):83-107.
[12] Sculli D, Hui C.F. Three dimensional stacking of containers. OMEGA 16, 1988, 85-594.
[13] Taleb-Ibrahimi M, De Castilho B, Daganzo C. F. Storage space vs handling work in container terminals. Transportation Research, 1993, 27B(1):13-32.
[14] Kim K H, Bae J W. Re-marshaling export containers in port container terminals. Computers and Industrial Engineering, 1998, 35(3-4):655-658.
[15] Kim K H, Park Y M, Ryu K R. Deriving decision rules to locate export containers in container yards. European Journal of Operational Research, 2000, 124(1):89-101.
[16]Kim K H, Park K T. A note on a dynamic space-allocation method for outbound containers. European Journal of Operational Research, 2003, 148(1):92-101.
[17] Kim H K, Jong S L. Satisfying Constraints for Locating Export Containers in Port Container Terminals. Lecture Notes in Computer Science, 2006, 3982: 564-573.
[18] E.mcdowill, gmartin d, cho, west. A study of maritime container handling. Oregon State University, Sea grant college program ads Corvallis, 1985, Oregon 97:1-10.
[19] Kap Hwan Kim. Evaluation of the number of rehandles in container yards. Computers and Industrial Engineering, 1997, 32(4): 701-711.
[20] Ping Chen, Zhaohui Fu, Adrew Lim. The yard allocation problem. Eighteenth national conference on artificial intelligence, 2002, 56-65.

[21] Zhang Chuqian, Liu Jiyin, Wan Yat-wah et.al. Storage space allocation in container terminals. Transportation Research Part B: Methodological, 2003, 37(10): 883-903.
[22] 徐剑华.用择箱指数法优化集装箱货场的利用率和取箱效率.港口装卸,1991,72: 46-51.
[23] 郝聚民,纪卓尚,林焰.混合顺序作业堆场 BAY 优化模型.大连理工大学学报,2000,40(1):102-105.
[24] 杨淑琴,张运杰,王志强.集装箱堆场问题的一个数学模型及其算法.大连海事大学学报,2002,28:115-117.
[25] Kim K H, Kim D Y. Group storage methods at container port terminals. The Materials Handling Engineering Division, 75th Anniversary Commemorative, 1994:15-20.
[26] Kim K H, Park K T. A dynamic space allocation method for outbound containers in carrier-direct system. Proceedings of the 3th Annual International Conference on Industrial Engineering Theories, Applications and practice 2, 1998: 859-867.
[27] Kim K H, Kim H B. The optimal determination of the space requirement and the number of transfer cranes for import containers, Computers and Industrial Engineering, 1998, 35(3):427-430.
[28] Kim K Y, Kim K H. A routing algorithm for a single transfer crane to load export containers onto a containership. International Journal of Production Economics, 1999, 59(1-3):425-433.
[29] Kim K H, Bae J W. A Dispatching Method for Automated Guided Vehicles to Minimise Delays of Containership Operations. International Journal of Management Science, 1999, 5(1):1-25.
[30] Kozan E, Preston P. Genetic algorithms to schedule container transfers at multimodal terminals. International Transactions in Operational Research, 1999,6(3):311-329.
[31] Eley, Michael. Solving container loading problems by block arrangement. European Journal of Operational Research, 2002, 141(2):393-409.
[32] Ng W C, Mak K L. Yard crane scheduling in port container terminals. Applied Mathematical Modelling, 2005,29(3):263-276.
[33] Linn Richard, Liu Ji-yin, Wan Yat-wah et.al. Rubber tired gantry crane deployment for container yard operation. Computers and Industrial Engineering, 2003,45(3):429-442.
[34] Y G Chung, S U RandHawa, E D McDowell. A simulation analysis for Transtainer based container handling facility. Computers & Industrial Engineering, 1988, 14(2):113-125.
[35] Lai K K, Lam K. A study of container yard equipment allocation strategy in Hong Kong. International Journal of Modeling and Simulation, 1994, 14(3):134-138.
[36] Zhang Chuqian, Wan Yat-wah, Liu Jiyin et.al. Dynamic crane deployment in container storage yards. Transportation Research Part B: Methodological, 2002, 36(6): 537-555.
[37] Richard J Linn, CHU-QIAN ZHANG, A Heuristic for dynamic yard crane development in a container terminal, IIE Transactions, 2003,35(2): 161-174.

[38] Gambadella L M , Rizzonli A E, Zafallon M. Simulation and planning of an intermodal container terminal. Simulation, 1998, 71(2):107-116.

[39] Lu Chen, Li-feng Xi, Jian-guo Cai et al. An integrated approach for modeling and solving the scheduling problem of container handling systems. Journal of Zhejiang University - Science A. 2006, 7(2):234 - 239.

[40] 真虹.港口生产调度过程优化.上海:上海科学技术文献出版社,1999.

[41] 张新艳.港口集装箱物流系统规则与仿真建模方法的研究与实现:(博士学位论文).武汉:武汉理工大学,2002.

[42] 王永庆.人工智能原理与方法.西安:西安交通大学出版社,1998.

[43] Ebru K. Bish et al. Dispatching Vehicles in a Mega Container Terminal. OR Spectrum, 2005, 27(4):491-509.

[44] Kim K H, Lee K M, Hwang H. Sequencing delivery and receiving operations for yard cranes in port container terminals. International Journal of Production Economics, 2003, 84(3):283-292.

[45] Erhan Kozan, Peter Preston. Mathematical modelling of container transfers and storage locations at seaport terminals. OR Spectrum, 2006.

[46] Dirk C M, Holger Orth. The allocation of storage space for transshipment in vehicle distribution. OR Spectrum, 2006:1 - 23.

[47] 李书臣,赵礼峰.仿真技术的现状及发展.自动化与仪表,1999,14(6):1-5.

[48] 梁静国. 管理决策仿真. 哈尔滨：哈尔滨工程大学出版社，2003.

[49] 鲁建厦，方荣，兰秀菊. 国内仿真技术的研究热点——系统仿真学报近期论文综述.系统仿真学报,2004,16(9):1910-1913.

[50] Ballis A, Abacoumkin C. A container terminal simulation model with animation capabilities. Journal of Advanced Transportation, 1996,30(1):37-57.

[51] Kosan, E. Optimizing Container Transfers at Multimodal Terminal. Mathematical and Computer Modeling, 2000,31(10-12):235-243.

[52] Gambardella, Mastrolilli, Rizzoli et al. An optimization methodology for intermodal terminal management. Journal of Intelligent Manufacturing, 2001,12(5-6):521-534.

[53] Chun Jin, Xinlu Liu, Peng Gao. An Intelligent Simulation Method Based on Artificial Neural Network for Container Yard Operation. Lecture Notes in Computer Science, 2004, 3174: 904-911.

[54]张泰. 基于本体的知识表示和知识共享. 江苏经贸职业技术学院学报, 2006,1:76-78.

[55]刘增锁，艾敬. 产生式规则在考试评分系统中的应用研究.计算机技术与发展,2006,16(7):162-164.

[56]Maojun Li., Shaosheng Fan., An Luo.: A Partheno-genetic Algorithm for Combinatorial Optimization. Lecture Notes in Computer Science, 2004, 3316:224-229.

[57] Miguel Rocha, Carla Vilela, José Neves. A Study of Order Based Genetic and Evolutionary Algorithms in Combinatorial Optimization Problems. Lecture Notes in Computer Science, 2000, 1821:601-611.

[58] Gomez-Albarran Ma de las Mercedes, Fernandez Pampillon Cesteros Ana Ma, Sanchez-Perez Juan Manuel. A routing strategy based on genetic algorithms. Microelectronics Journal (Incorporating Journal of Semicustom ICs), 1997, 28(6):641-656.

[59] Davies Cedric, Lingras Pawan. Genetic algorithms for rerouting shortest paths in dynamic and stochastic networks. European Journal of Operational Research, 2003, 144 (1):27-38.

[60] M. Zambaldi, W. Ecker. How to bridge the gap between simulation and test. Test Conference, 2004. Proceedings. ITC 2004. International, 2004:1091 - 1099.

[61] 欧阳荣彬,郭陟,顾明.基于排队模型的软件性能测试框架研究.计算机工程,2006,32(3):73-75.

[62]谈琳,罗永红.实时软件的仿真测试平台的研究.系统仿真学报,2005,22(1):247-250.

[63]顾浩,周玉芳,岳岗.指挥自动化仿真测试技术研究.系统仿真学报,2002,14(12):1666-1670.

[64]钟德明,刘斌,阮镰.嵌入式软件仿真测试环境软件体系结构研究.北京航空航天大学学报,2005,31(10):1130-1134.

[65]Wen, C.H.-P, Wang, L.C, Kwang-Ting Cheng. Simulation-based functional test generation for embedded processors. High-Level Design Validation and Test Workshop, 2005. Tenth IEEE International, 2005:3-10.

[66]邓尔风.面向对象的方法在移动电话通信仿真测试环境中的应用.计算机仿真,1999,16(4):71-74.

[J]

附录 A 堆场数据库表设计

表 A.1 街定义

Tab. A.1 Block definition

字段名称	字段类型	字段长度	说明
blockName	文本	50	街名
beginRow	文本	50	起始行
endRow	文本	50	最终行
rowAmt	文本	50	行数
tierAmt	文本	50	贝数
rehandleMode	文本	50	翻倒规则（L：左边，R：右边，B：两边）

表 A.2 贝定义

Tab. A.2 Bay definition

字段名称	字段类型	字段长度	说明
block	文本	50	街名
bay	文本	50	贝号
rowAmt	文本	50	定义的行数
tierAmt	文本	50	定义的层数
oprtMode	文本	50	作业方式
startRow	文本	50	中间到两边作业时，作业起始行
orientFlag	文本	50	中间到两边作业时，优先作业方向
weightRule	文本	50	轻重原则
lapFlag	文本	50	同重量可否压箱标志
areaID	文本	50	区域号

表 A.3 设备信息

Tab. A.3 Facility

字段名称	字段类型	字段长度	说明
facID	文本	50	设备编号
facType	文本	50	设备类型
top	文本	50	最大作业高度（层）
scope	文本	50	最大作业跨度（行）
locBlock	文本	50	设备所在位置街名
second	文本	50	设备所在位置贝号

表 A.4 设备作业能力

Tab. A.4 Facility operation ability

字段名称	字段类型	字段长度	说明
facID	文本	50	设备编号
facType	文本	50	设备类型
oprtTier	文本	50	作业目标所在层
gsFlag	文本	50	取/放箱标记
first	文本	50	第一行最高堆箱数
second	文本	50	第二行最高堆箱数

表 A.5 设备作业成本

Tab. A.5 Operation cost of facility

字段名称	字段类型	字段长度	说明
facID	文本	50	设备编号
facType	文本	50	设备类型
conSize	文本	50	集装箱尺寸
isFul	文本	50	集装箱空重
handTime	文本	50	抓取时间
handCost	文本	50	抓取成本
upTime	文本	50	提升一层时间
upCost	文本	50	提升一层成本
downTime	文本	50	下降一层时间
downCost	文本	50	下降一层成本
rowTime	文本	50	平移一行时间
rowCost	文本	50	平移一行成本
bayTime	文本	50	平移一贝时间
bayCost	文本	50	平移一贝成本

表 A.6 设备作业模板

Tab. A.6 Operation template of facility

字段名称	字段类型	字段长度	说明
facID	文本	50	设备编号
facType	文本	50	设备类型
width	文本	50	模板宽度
height	文本	50	模板高度
order	文本	50	作业顺序编号
row	文本	50	模板行号
tier	文本	50	模板层号

表 A.7 箱位分配规则

Tab. A.7 Group rule

字段名称	字段类型	字段长度	说明
conType	文本	50	集装箱型
size	文本	50	尺寸
port	文本	50	目的港
areaID	文本	50	堆放区域
facID	文本	50	作业设备

表 A.8 堆场堆存状态

Tab. A.8 Stack configuration status

字段名称	字段类型	字段长度	说明
areaID	文本	50	区域号
block	文本	50	街名
bay	文本	50	贝号
row	文本	50	定义的行数
tier	文本	50	定义的层数
conNum	文本	50	箱号
size	文本	50	尺寸
type	文本	50	箱型
port	文本	50	目的港
filled	文本	50	空箱/重箱
weight	文本	50	重量等级
isFull	文本	50	是否有箱
isEmpty	文本	50	是否为空
isSealed	文本	50	是否封存

附录 B 港口大门作业仿真模型

1. 用户事件的安排

```
void SimApp_portGate::userEvt(){
    event(firstHand,1,1);          //提箱集卡首次到达
    if (recode==1006) return 1;
    event(firstAlloc,2,2);         //放箱集卡首次到达
    if (recode==1006) return 1;
    if (firstHand<firstAlloc)    {
        beginWorkTime=firstHand;
    }
    else    {
        beginWorkTime=firstAlloc;
    }
    event(simexp->getTime(),100,0);
    if (recode==1006) return 1;
    return 0;
}
////////////////////////////////////////////////////////////
```

2. 仿真模型

```
void SimApp_portGate::simModel(){
do {
    if(target!=0) {
        NADDR=target;
        target=0;
    }
    switch(NADDR) {
    case 1:    {  //提箱集卡到港
            iz=(int) (exponential ( mean1,min1,max1,1)+0.5);
            if(recode==1006)      break;
            genera(iz,maxEntityAmt,1,1);
            if(recode==1006)      break;
```

```
        binObj[1].arrive(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        //确定到达集卡的属性
        TX[LTX].setFreePara(1,111);      //集卡性质：提箱
        TX[LTX].setFreePara(2,10);       //待提箱序号
        TX[LTX].setFreePara(3,2);        //倒箱范围
        target=3;
        break;
    }
case 2:{   //进箱集卡到港
        iz=(int) (exponential ( mean2,min2,max2,1)+0.5);
        if(recode==1006)      break;
        genera(iz,maxEntityAmt,1,2);
        if(recode==1006)      break;
        binObj[1].arrive(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        TX[LTX].setFreePara(1,222);      //集卡性质：进箱
        TX[LTX].setFreePara(2,TX[LTX].getIdentif()); //进箱的箱号
        long typeOrd = (long)rander(0,2.9999,5); //箱型
        TX[LTX].setFreePara(3,typeOrd);
        long sizeOrd=(long)rander(0,1.9999,1); //尺寸
        TX[LTX].setFreePara(4,sizeOrd);
        long emptyOrd=(long)rander(1,1.9999,5); //空重
        TX[LTX].setFreePara(5,emptyOrd);
        long weightOrd=(long)rander(1,5.9999,1); //重量等级
        TX[LTX].setFreePara(6,weightOrd);
        long portOrd=(long)rander(1,5.9999,11); //目的港
        TX[LTX].setFreePara(7,portOrd);
        target=3;
        break;
    }
```

```
case 3:    {   //集卡向入口大门请求服务
mfacObj[1].mseize(3,1,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1005||recode==1006)      break;
        binObj[1].depart(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        target=4;      //继续进入下一模块，准备接受服务
        break;
    }
case 4:        {   //大门为集卡服务
        binObj[3].arrive(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        //准备接受服务
mfacObj[1].msetup(3,5,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1005||recode==1006) break;
    }
case 5:    {   //服务
        if (TX[LTX].getFreePara(1)==111){   //提箱服务时间
            iz=(int)(erlang(mean3,min3,max3,1,1)+0.5);
            if(recode==1006) break;
        }
        if (TX[LTX].getFreePara(1)==222){   //进箱服务时间
            iz=(int)(erlang(mean4,min4,max4,1,1)+0.5);
            if(recode==1006) break;
        }
        //服务模块
mfacObj[1].mwork(iz,5,0,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006||recode==1005)      break;
    }
case 6:{ //服务决策,在这里嵌入提箱、放箱程序模块
        if (TX[LTX].getFreePara(1)==111) { //执行提箱模块
            m_conNum=readData.getHandUpCon(TX[LTX].getFreePara(2));
            if (m_conNum!="") {
                HandUpRun();
```

```
            }
        }
        if (TX[LTX].getFreePara(1)==222)    {   //执行放箱箱模块
            m_contype=conTypeVec[TX[LTX].getFreePara(3)]; //箱型
            m_connum= TX[LTX].getIdentif();//箱号
            if (TX[LTX].getFreePara(4)==0)  {   //尺寸
                m_size="20";
            }
            else    {
                m_size="40";
            }
            if (TX[LTX].getFreePara(5)==0)  {   //空重
                m_isempty="F";
            }
            else    {
                m_isempty="E";
            }
            m_weight= TX[LTX].getFreePara(6); //重量等级
            m_port=destPortVec[0]; //目的港
            AllocRun();
        }
        //集卡服务结束，离开箱柜
        binObj[3].depart(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
mfacObj[1].mknock(3,7,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006||recode==1005)  break;
    }
case 7:    {   //集卡进入堆场接受设备作业
        if (TX[LTX].getFreePara(1)==111)    {   //提箱集卡延时
            iz=(int)(erlang(mean5,min5,max5,1,1)+0.5);
            if(recode==1006)  break;
        }
        if (TX[LTX].getFreePara(1)==222)    {   //进箱集卡延时
```

```
            iz=(int)(erlang(mean6,min6,max6,1,1)+0.5);
            if(recode==1006) break;
        }
        advance(iz,8);
        if(recode==1006||recode==1005)      break;
    }
case 8:    {  // 释放设备，清空入口大门现场
mfacObj[1].mclear(&TX,&AL,LTX,LAL,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006 )     break;
        binObj[2].arrive(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        TX[LTX].setFreePara(5,T); //实体离开作业堆场，到出口大门报到时间
        TX[LTX].setPrReturn(0);
        target=11;
        break;
    }
case 11:   {  //集卡向出口通道请求服务
mfacObj[2].mseize(11,1,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1005||recode==1006)      break;
        binObj[2].depart(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        target=12;     //继续进入下一模块，准备接受出口通道服务
        break;
    }
case 12:   {// 大门出口为集卡服务
        binObj[4].arrive(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
        //准备接受服务
mfacObj[2].msetup(3,13,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1005||recode==1006) break;
    }
case 13:   {  //服务时间
        if (TX[LTX].getFreePara(1)==111)    {  //提箱集卡离港服务时间
```

```
            iz=(int)(erlang(mean7,min7,max7,1,1)+0.5);
            if(recode==1006) break;
        }
        if (TX[LTX].getFreePara(1)==222)    {   //进箱集卡离港服务时间
            iz=(int)(erlang(mean8,min8,max8,1,1)+0.5);
            if(recode==1006) break;
        }//服务
  mfacObj[2].mwork(iz,13,0,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
      if(recode==1006||recode==1005)  break;
   }
   case 14:   {   //集卡服务结束
        binObj[4].depart(1,&TX,LTX,T,&recode,IPRINT,log_str,log_strCh);
   mfacObj[2].mknock(3,18,&TX,&AL,LTX,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006||recode==1005)  break;
      }
   case 18:   {   // 释放设备，清空大门现场
   mfacObj[2].mclear(&TX,&AL,LTX,LAL,T,&recode,IPRINT,log_str,log_strCh);
        if(recode==1006)      break;
        termin();//集卡实体消亡
        if(recode==1005) break;
        break;
      }
   default:
      recode=1006;
      break;
   }
  } while(target!=0);
}
```

攻读硕士学位期间发表学术论文情况

[1] Jianfeng Shen, Chun Jin, Peng Gao. A Nested Genetic Algorithm for Optimal Container Pick-up Operation Scheduling on Container Yards. Advances in Natural Computation, the 2nd International Conference on Natural Computation (ICNC'06), Xi' an, China, September 2006 Proceedings, Part I:666-675. (SCI 检索，论文第三章)

[2] SHEN Jianfeng, JIN Chun, GAO Peng. Algorithm Design on Optimal Container Pick-up Operation Scheduling. The proceedings of The 13th International Conference on Industrial Engineering & Engineering Management (IE&EM' 2006) and The 1st International Conference on Asian Industrial Engineering & Management (AIE&M' 2006),2006(ISTP 检索，论文第三章)

[3] 沈剑峰，金淳，霍琳. 基于离散事件仿真的信息系统性能测试方法. 系统仿真技术及其应用（2006 系统仿真技术及其应用学术会议论文集），2006:8(409-412).(论文第五章)

[4] 沈剑峰，金淳，高鹏. 基于知识的集装箱堆场箱位分配计划研究. 计算机应用研究.（已录用，论文第二章）

[5] 高鹏,金淳,沈剑峰. 集装箱堆场提箱作业计划优化算法研究. Proceedings of the Sixth World Congress on Intelligent Control and Automation (WCICA06). 2006:7332-7337. （EI 检索，论文第三章）

参加科研项目：

1. 企业横向项目：
   《集装箱堆场作业优化算法的软件系统开发》
2. 国家自然科学基金项目(70571008)：
   《综合交通体系下港口物流协调的 HLA 仿真优化方法研究》
3. 国家重点学科“管理科学与工程” 2004 年度交叉项目：
   《基于知识的复杂系统仿真方法研究》
4. 教育部留学回国人员科研启动基金资助项目(教外司留[2005]383 号文件)

致　谢

首先，我以最崇高的敬意向我的导师金淳副教授表示衷心的感谢！感谢他两年半来对我学术上的指导和帮助，感谢他对我生活上的关心，感谢他以师德激励我前进。金老师严谨的治学态度和艰苦奋斗的工作作风给我留下了深刻的印象，在他的言传身教、耳濡目染下，我在读研期间得到了迅速的成长。我深深感到：没有金老师的悉心指导，哪里会有我的研究成果和论文？！导师的认真、仔细和耐心让我不敢懈怠，我真心感谢并祝福我的导师！

我要感谢荣莉莉教授，同她的一次交谈激发了我对研究课题的灵感。我还要感谢党延忠教授、胡祥培教授、邓贵仕教授、王旭平副教授、吴江宁副教授、周宽久副教授等各位老师对我的教导，两年多来，他们的人格魅力和学术修养不断催促我前进。

我还要特别地感谢师兄高鹏博士，他在技术方面给予了我很大支持，在共处的岁月里，我从他身上学到了很多。同时，我还要感谢师姐刘昕露，感谢同学王雪杰、于越，师弟王刚、霍林，师妹赵璐给我的帮助，感谢室友陈坤、吴治宗、刘驰、郭瑞东在生活和学习上对我的关心。

最后感谢我的父母和亲人，他们不但养育了我，而且为我创造了宽松的求学环境，他们的鼓励和支持让我感到温馨！