大连理工大学

硕士学位论文

集装箱堆场作业调度优化问题研究

姓名：高鹏

申请学位级别：硕士

专业：管理科学与工程

指导教师：金淳

20051201

摘　要

港口集装箱堆场是集装箱运输过程中的重要环节，当前在集装箱运输要求快捷化、节约化、简单化及标准化的前提下，要求港口集装箱堆场向自动化、无人化、快捷化方向改进，因此需要提高集装箱堆场作业效率。本文研究的提箱作业和落箱作业是集装箱堆场大门一侧的主要作业之一，作业计划制定的效率直接影响了堆场大门的通过效率。

本文首先通过分析说明落箱作业的影响因素包含了提箱作业时要考虑的影响因素，详细介绍了落箱作业的过程，提出了基于规则的推理过程及方法来解决港口集装箱堆场出口箱箱位分配问题，将作业过程中的制约因素以事实和规则的知识表示形式表示出来，存储到数据库和规则库中，通过推理机制得到最合理的放箱位置。

接着，分析了具有多阶段性决策问题的提箱作业优化问题。将提箱作业优化问题归结为一个包含倒箱优化和最短路搜索两部分的多阶段决策问题。建立了提箱作业优化的数学模型，该模型以提箱作业总成本最小为目标，以规则库中的作业规则为约束条件。根据建立的数学模型，通过对问题状态空间的分析，提出并实现了内外嵌套的两层结构的启发式搜索 A*算法对问题进行求解。

最后，本文进行了实例的验证与比较。通过落箱算法的实例验证，表明了该方法分配箱位形成的堆垛即达到了方便装船作业的目的，又节约了堆场管理成本，从而也说明了基于规则推理过程设计的合理性。然后，对提箱算法做了实例验证并与外层为遗传算法内层为 A*算法的算法方案进行了比较，表明了提箱算法的结果的最优性和搜索的高效性。

本文研究与并发的落箱作业和提箱作业的优化算法不仅具有实际的应用价值，也为同类多阶段优化问题提供的新的优化搜索方法。

关键词：集装箱堆场；启发式算法；遗传算法；提箱作业；自动分配箱位

Research on Optimization Problem on Container Yard Operation Scheduling

Abstract

Port container yard, as an important tache of the process of container transportation, is required to be automated, unmanned and shortcuted as the trend of economization, simplification and standardization in container transportation. Therefore, it is necessary to improve the operation efficiency of container yard. This paper focuses on two main operations of the gate side of container yards: pick-up operation and storage space allocation operation on the gate side of container yards, whose scheduling efficiency has direct impact on the efficiency of truck's passing though the gate.

This paper explained the factors of pick-up operation were included in which of storage space allocation firstly. Then, it was proposed to sovle the problem of storage space allocation in container yard by rule-based reasoning, the constraints factors of this operation were denoted as fact and rule, stored in the database and rule base respectively. The best position was gained by reasoning.

Then, analyzed the pick-up operation process on the gate side of container yard for import containers, the optimization problem was attribute to be a muti-stage decision making problem. A mathematical model which aims to minimize the total cost of whole pick-up operation process is established, it was a multi-phase optimization model comprised of two sub-models: rehandling operation scheduling and the shortest path searching. A two-layer A* heuristic algorithm with the inner A* algorithm structure embedded into the outer one was put forward and realized, the inner one was responsible for searching the shortest path and the outer one for optimizing rehandling strategy.

Evaluation and verification are proposed at last. A container allocation operation example shows that not only the container stack allocated according to the above method is convenient to shipment loading, but also in favor of saving the yard management cost. For pick-up operation, after comparing A* algorithm with Genetic algorithm for the same problem, the result shows that A* algorithm has better performance in optimization strategy and efficiency.

The optimization algorithm of this paper, not only has it great value for application, but also supply a new method for similar problems.

Key Words: Container Yard; Heuristic Algorithm; Genetic Algorithm; Pick-up Operation; Container Allocation

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：高鹏 日期：2006.1.8

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：高鹏

导师签名：金涛

2006 年 1 月 8 日

1 绪论

1.1 研究背景

集装箱码头主要包括岸壁、前沿、集装箱堆场、集装箱货运站、指挥塔、维修车间、大门等固定设施。集装箱专用码头堆场可采用的设备有轮胎式龙门起重机、轨道式龙门起重机、集装箱半挂车、跨运车、正面吊运机、叉车[1,2]。

图 1.1 为集装箱码头平面布置图，其中，1——岸桥；2——集装箱拖挂车；3——轮胎式龙门起重机；4——加油站；5——电力站；6——货运站；7——控制室；8——维修车间；9——门房；10——泊位；11——集装箱堆场。

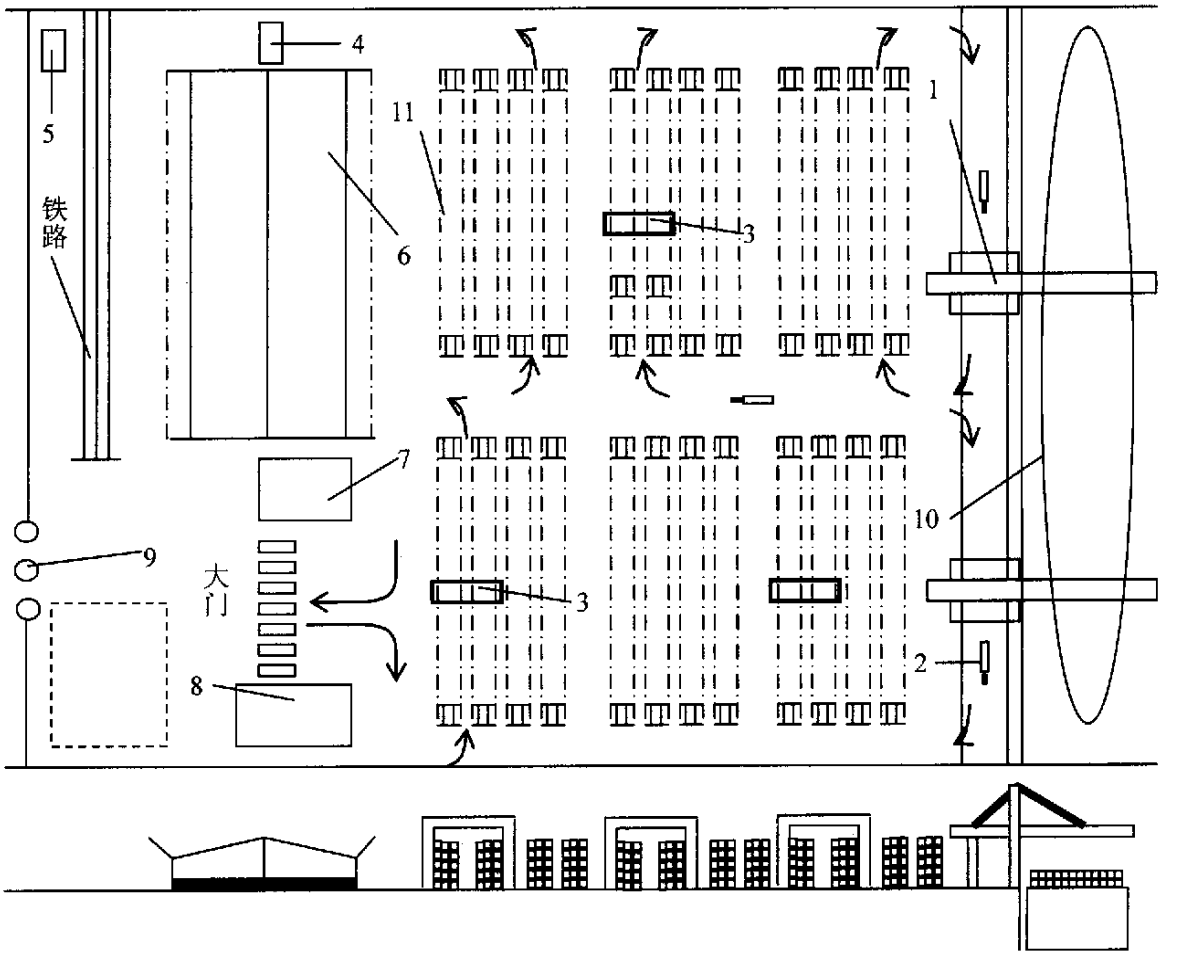

图 1.1 集装箱码头平面布置图

Fig.1.1 Plane Layout of Container Yard

1.1.1 集装箱码头堆场概述

集装箱码头堆场是供装卸船舶堆放集装箱的场所，同时也是临时保管和向货主交接集装箱的地方。在堆场上对集装箱进行分区分类堆放，并按照集装箱的规格尺寸，在堆场上划出“箱位”线，标明号码。尽管业务模式的差异使得各类堆场场站的服务功能不尽相同，但所有业务功能都是在堆场核心服务功能之上发展而来，因此他们的关键业务基本一致，堆场的主要业务工作是办理集装箱的装卸、转运、装箱、拆箱、收发、交接保管、堆存、捆扎、掏载、搬运、以及承揽货源等，此外，还受理集装箱的修理、冲洗、熏蒸和有关衡量等工作[1,2]。

港口集装箱堆场通常被分为三个部分：集装箱前方堆场（marshalling yard）、集装箱后方堆场（container yard）、空箱堆场（van pool）。

集装箱前方堆场是指在靠近集装箱码头前沿处，为加速船舶装卸作业，暂时堆放集装箱船直接装卸的集装箱的场地。其作业内容是：当集装箱船到港前，有计划有次序地按积载要求将出口集装箱整齐地集中堆放，卸船时将进口集装箱暂时堆放在码头前方，以加速船舶装卸作业。

集装箱后方堆场是集装箱重箱或空箱进行交接、保管和堆存的场所。集装箱后方堆场是集装箱装卸区的组成部分。是集装箱运输“场到场”交接方式的整箱货办理交接的场所（实际上是在集装箱卸区“大门口”进行交接的）。

集装箱空箱堆场是专门办理空箱收集、保管、堆存或交接的场地。它是专为集装箱装卸区或转运站堆场不足时才予设立。这种堆场不办理重箱或货物交接。它可以单独经营，也可以由集装箱装卸区在区外另设。

有些国家对集装箱堆场并不分前方堆场或后方堆场，而统称为堆场。

一般地，堆场由多个街（block）组成，每个街有连续的贝位（bay）组成（一般为 40-60 个贝位），每个贝位通常包括 6-8 行（row）。堆场的堆高为层数（tier）。在堆场上的街贝行层都通过一定的编码表示，其中街通常由字母和自然数（A01,A02,B01,B02）共同表示，贝行层通常由自然数（1,2,3...）表示。在堆场中，通常 20 尺的集装箱占用 1 个贝位，其所在贝号为奇数，40 尺的集装箱占用 2 个贝位，其所在贝号为偶数。大多数的堆场都不允许集装箱混贝放置，即把 20 尺的箱和 40 尺的箱放在同一贝位上。堆场中每个箱位都可以通过其所在的街贝行层的编码来唯一确定。如给定一个堆场编码(A010010302)表示该箱位的位置是 A01 街 001 贝 03 行 02 层，而且可以判断出这是 20 尺的箱；堆场编码(B030060504)表示该箱位的位置是 B03 街 006 贝 05 行 04 层，而且是 40 尺的箱。大多数集装箱堆场按功能划分为不同的区域，例如空箱堆存区、拆装箱区、制冷区、残箱区、修箱区、重箱堆存区、

暂存区等，每个堆场区域有相邻的横向的街组成。这样便于场桥移动和减少场桥占用集卡作业线时间。堆场堆放示意图如图 1.2 所示。

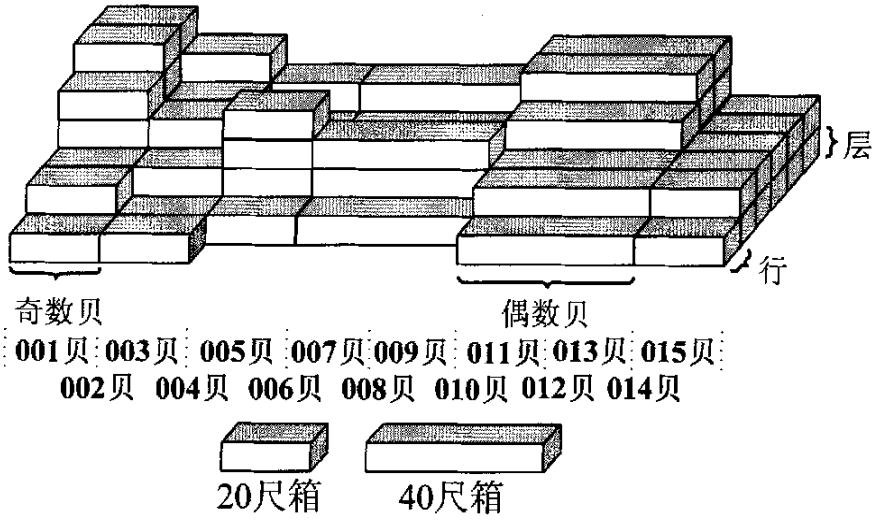

图 1.2 堆场堆放示意图

Fig.1.2 Schematic Drawing of Terminal Stacking

1.1.2 大门作业概述

大门是集装箱码头的出入口，通过大门交接集装箱、集装箱货物和集装箱货物的各种单据。大门是场站系统管理中最关键的环节之一，是很多场站操作得以顺利进行的前提之一。其管理的核心是对集装箱和货物进出场站进行高效，准确的控制。大门的工作主要是：控制车辆进出场；对于进场的集装箱检查箱体是否完好，若有破损则需要录入破损记录；录入进场车辆和货物信息；对出场的货物和集装箱进行核对；打印入站操作条；打印出场货物交接单等。

大门的主要作业流程如下：

（1）进场箱作业

对进场的集装箱通过大门时，由地秤自动测定其重量，根据该重量和运输的目的地，靠计算机确定后方堆场内该集装箱应该放置的箱位位置，然后由大门交给卡车司机一张带有箱位号码的门票（slip），卡车司机必须按指示的计划把集装箱放在指定的箱位上。

（2）出场箱作业

内陆承运人从堆场提取集装重箱时，空卡车到达大门，提交提货单给大门，计算机确定提货单内指定的集装箱的箱位位置，计算最优提箱计划，然后由大门交给卡车司机一张带有箱位号码的门票（slip），卡车司机必须按指示的计划到指定的箱位位置提取集装箱，

提取集装箱后，卡车到大门处经审查放行离开堆场。提空箱一般根据单证上的要求与大门已存信息进行发箱。

1.2 港口集装箱堆场作业研究现状

集装箱运输是货物在集装箱内进行运送的一种新颖的运输方式。因为采用这种运输方式具有装卸作业效率高、加速集装箱车船周转时间、节约费用、理货手续得到简化，便于采用计算机管理、实现标准化作业等优点，使得仅有几十年经验的现代化的集装箱运输，却在科技进步的推动下取得了飞速的发展。当今世界集装箱运输已经实现了设备化和硬件的标准化，目前正朝着自动化和软件的标准化方向发展。近年来随着各种运输方式联合运输的发展，集装箱联运，特别是海运和铁路联运也迅速发展起来。随着全球集装箱运输业的迅猛发展，作为集装箱联运大物流过程的一个重要环节，港口货物吞吐量日益增长，港口集装箱作业效率的高低，直接关系到整个运输过程的效率和成本，关系到船公司的效益和货主的切身利益，因此如何对港口集装箱物流系统进行更加合理有效地规划与设计，以最大限度地发挥其作业能力，是目前急需解决的问题。

目前国内外在集装箱码头主要的组织管理手段有包括：集装箱码头信息系统(CTIS)、集装箱码头模拟系统(CTSS)、故障诊断软件包、集装箱码头综合管理系统、电子防摇控制系统(HWAC)等，应用效果较好的如美国的洛杉矶集装箱码头、荷兰阿姆斯特丹集装箱码头、韩国釜山港集装箱码头、新加坡港等。

近年来，随着集装箱运输业的发展需要，以及计算机仿真软件的进步，研究者们多利用计算机模拟技术和优化算法对港口集装箱物流系统进行研究，这些研究主要集中在运营环节及装卸工艺系统方面的研究。其中运营环节发生在码头前沿、水平搬运、堆场作业以及大门作业，因此研究可以分为在确定合理的泊位利用率、装卸设备资源的合理配置及调度、堆场作业、装卸工艺流程的选择等部分。

下面对堆场作业以及大门作业方面的国内外的研究情况进行叙述。

1.2.1 关于堆场内装卸设备资源的合理配置及调度的研究

港口的主要任务是装卸。在港口集装箱物流作业过程中，堆场是码头作业中最复杂的部分，堆场的资源调配在很大程度上决定了码头的效率，因此堆场管理是场站管理的核心之一，在堆场管理中占有重要地位。集装箱的装卸效率的高低，直接决定了使通过港口的集装箱物流更经济、更合理，决定整个作业过程的效率。因此，对集装箱码头装卸工艺系统进行优化，选择合适的码头装卸工艺系统是非常必要的。在我国现有的集装箱专用码头，几乎全部采用了轮胎式龙门起重机作业，再配合部分正面吊、叉车等设备作业，水平运输配备集卡和底盘车进行作业。在设备资源的配置和调度优化方面的研究如下：

1991 年香港的 China light & Power 公司，曾委托 Sandwell Inc.利用模拟模型研究了增加何种设备或设施，以及以何种方式增加这些设备或设施，才能以更加经济的方式适应港口吞吐能力的增长要求[3]。Ballis 和 Abacoumkin 建立了一个包括由于设备不匹配而引起的交通堵塞和延迟的仿真模型，他们使用这个模型来评价一个码头的堆场设计、设备数量、集卡到达、运行规则[4]。Kozan 研究过利用火车疏运集装箱的码头的吞吐量，他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[5]。K.H.Kim 和 J.W.Bae 采用混合整数规划模型研究将集装箱分配给自动导向车辆（AGV）的调度问题[6]。Ebru K. Bish 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[7]。Ebru K.Bish 还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，其主要成果是解决了调度的 NP 难题，并发了一种启发式的算法使每艘船的服务时间达到最小[8]。R.Van derMeer 研究了以 AGV 为主导的堆场内部运输系统的控制[9]。Zhang 用整数规划模型来研究轮胎式起重机的配置问题，以减少堆场工作的延迟，并通过拉格朗放松启发式理论来获得接近最佳解决方案[10]。K.Y.Kim 和 K.H.Kim 运用整数规划来最小化跨运车在堆场的运输时间[11]。在国内，张新艳在她的博士论文中，忽略场桥的影响，用基于生物进化策略对堆场上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[12]。天津大学的别社安教授，提出了运用循环网络模拟方法将排队论、网络计划技术、计算机模拟技术结合起来，对港口的营运过程进行模拟，从而得到港口泊位、仓库等的利用情况，对具体的装卸工艺进行模拟，可以得出合理的调度管理方案[13]。真虹通过仿真证明，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，岸桥集中作业时和均匀分配方式相比，更有助于减少船舶在港逗留时间。仿真结果显示，采用自动调配作业方式时，设备利用率最高，船舶等待时间和等待队长最短[14]。

从以上研究可以看出，国内研究比较少，国外研究中目前多用启发式算法和整数规划来解决装卸设备资源的配置和调度优化问题。

1.2.2 关于堆场计划及堆放策略的研究

堆场计划是对集装箱在堆场内进行装卸、搬运、贮存、保管的安排。这是为了更能经济合理地使用码头堆场和有计划的进行集装箱装卸工作而制订的。堆场计划主要内容有确定空箱、重箱的堆放位置和堆高层数；装船的集装箱应按先后到港顺序、集装箱的种类、载重的轻重分别堆放；同一货主的集装箱应尽量堆放在一起。堆放策略主要是指集装箱区域的划分。

Kim 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[15-17]。Talebibrahimi 和 De Castilho 及 Daganzo，分别比较了进口箱和出口箱在不同堆放策略下所需的堆场面积并得出了工作量[18,19]。Lai 和 Lam 使用仿真来比较堆场设备在不同调度策略下的吞吐量、设备使用率、等待时间[20]。Kozan 和 Preston 使用整数规划决定优化的堆放策略和相应的堆场起重机分配，以减少用于一艘船的所有起重机的最大工作时间[21]。Avriel 和 Shields 分别在其研究工作中对到港船舶的装卸箱过程进行模拟，研究了岸边集装箱起重机调度问题，同时通过模拟还可以帮助制定堆场计划以减少翻箱率[22,23]。另外还有人对集装箱在堆场内的堆放情况进行模拟，分析了对于进口箱和出口箱的处理策略问题，主要讨论了集装箱在堆场内的堆放策略以提高堆场的利用率的问题[24,25]。

由于堆场计划制定和堆放策略制定涉及众多作业知识和规则因素，提高了问题的复杂度，因此研究成果还不多。

1.2.3 关于大门处作业的研究

在国内外，主流的集装箱堆场管理都采用了智能大门，其中大多提供提放箱的优化算法，对于进场放箱要根据当前堆场策略计算出最佳的放箱位置，对于出场提箱要计算出最优的提箱作业计划，由于货主到堆场提箱，提箱的时间是随机的，事先并不能准确获知，具有不确定性。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，降低场站整体作业效率，因此有必要对提放箱优化算法的效率进行研究。

目前国内外就此问题展开相关的研究学者为数不多，研究角度各不相同，采用的技术方法也略有差异，但目标基本一致，都取得了一定的成果。K.H.Kim 和 J.W.Bae 假设当前集装箱堆场图可以获得，同时期望的贝布局也可以提供，研究了集装箱重新配置作业问题，将问题分解为 3 个子问题：贝匹配、移动计划、任务排序，提出了一种变当前贝布局为期望贝布局的方法，达到移动最少数量的集装箱和移动最短距离的目标[26]。香港科技大学 Chuqian Zhang 和 Jiyin Liu 等提出了一种变动放箱范围的方法，在每个计划范围中，问题可以分解为两个层次，每个层次可以由一个数学规划模型表示，确定了每个时间段内每个街中放置集装箱的总数，实现了平衡各个街之间的工作量，最小化将集装箱从前方堆场运送到在港船舶之间的距离[27]。Peter Preston, Erhan Kozan 针对不同的集装箱操作计划分析了以最优存储策略为目标的港口系统，建立了以船舶在港时间最小化为目标函数的集装箱箱位分配模型，并用遗传算法获得了最优解[28]。Kim 讨论了在一个贝（bay）内由场桥作业时计算期望倒箱数目的算法[29]。Kozan 和 Preston 把遗传算法用于减少集装箱倒箱和运输时间及

船的在港时间[21]。郝聚民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顺序作业堆场贝优化模型[30]。

1.2.4 问题的提出

从以上的研究表示，存在以下几点问题：

（1）目前在集装箱堆场中使用的作业设备主要以场桥为主，正面吊和叉车为辅，而众多的研究中大多围绕设备资源的配置展开，而很少涉及作业的作业细节；

（2）在堆场的计划的制定的研究中，大多只关心场桥的作业而忽略正面吊和叉车的辅助作用，同时在涉及倒箱作业时，只考虑倒箱数目对作业成本和效率的影响，而实际作业中，倒箱的过程中设备的移动距离是影响作业成本和效率的很重要因素。

因此有必要对多种设备作业时放箱和倒箱的位置选择及具体过程进行研究，从而提高效率降低作业成本。

1.3 研究内容及技术路线

本文从高效性、准确性和实用性出发，对集装箱堆场作业优化的建模、算法的设计与实现方法进行研究。运用人工智能、运筹学方法、面向对象的建模方法，主要通过对堆场大门一侧的进口集装箱提箱和出口集装箱落箱作业问题的建模及分析，提出并实现基于人工智能方法的集装箱堆场大门一侧作业优化算法，并对所提出的算法进行验证。

本文的研究工作按以下步骤进行：

（1）介绍目前世界上集装箱堆场作业优化的研究现状，分析研究的不足之处，提出研究需要解决的问题；

（2）深入了解并分析解决本问题的各种理论方法的技术；

（3）通过对集装箱堆场作业的业务分析，把堆场作业所需的主要领域知识归纳成简单的规则表示形式，建立相应的规则库，提供精确的规则推理机制；

（4）针对堆场的提箱和放箱作业，分别提出提箱和自动分配箱位的优化算法；

（5）自动分配箱位算法主要建立规则库及其推理过程；

（6）通过对提箱作业的分析，建立其数学模型，在自动分配箱位算法的研究的基础上，针对建立的模型提出两层结构的启发算法，并与遗传算法的解决方案进行比较；

（7）最后通过实际算法对算法的效率及最优性进行验证。

2 堆场作业优化算法的理论基础

在港口集装箱物流系统中，在给定的准则下，从诸多可选方案中寻求一个最优的作业计划安排通常是一个组合优化问题，往往具有层次性、多组合性、动态性以及多约束条件等特点。在规模较小时，此类问题可用传统的运筹学方法如分支定界法、动态规划等方法求得最优解，求解时计算相对复杂，而随着问题规模的增大，求解的复杂度也将呈指数增长，因此利用传统的运筹学优化方法显得无能为力。人工智能方法成为了近年来解决多约束、多规则的复杂优化问题的首选，从上一章介绍的国内外研究现状中可以看出，目前的研究多集中在启发式算法、遗传算法等，这些方法都是人工智能中的一部分，本文也将采用人工智能的方法，来解决堆场中集装箱的提箱和落箱问题。

2.1 状态空间概述

人工智能的研究领域很多，但从它们解决现实问题的过程来看，都可抽象为一个问题求解的过程。所谓问题求解就是通过在某个可能的解空间内搜索，以寻找一个能解决某类问题的解，这一搜索操作应在有限步内完成。状态空间法和问题规约法是两种最常用、最基本的问题求解的方法。

问题求解是涉及归约、推断、决策、规划、常识推理、定理证明和相关过程的核心概念。在分析了人工智能研究中运用的问题求解方法之后，会发现很多问题求解方法采用的是试探搜索方法，也就是说，这些方法是通过在某个可能的解空间内寻找一个解来求解问题。这种基于解空间的问题表示和求解方法就是状态空间法，它是以状态和算符（operator）为基础来表示和求解问题的[31]。

状态（state）是为描述某类不同事物间的差别而引入的一组最少变量$q_0,q_1,...,q_n$的有序集合，其矢量形式为：

$$Q=[q_0,q_1,...,q_n]^T \tag{2.1}$$

式中每个元素$q_i(i=0,1,...,n)$为集合的分量，称为状态变量。给定每个分量的一组值就得到一个具体的状态。使问题从一种状态变化为另一种状态的手段称为操作符或算符。算符可为走步、过程 、规则、数学算子、运算符号或逻辑符号等。问题的状态空间（state space）是由问题的全部状态及一切可用的算符所构成的集合。它包含三种说明的集合，即所有可能的问题初始状态集合 S、操作符集合 F 以及目标状态集合 G。因此可把状态空间记为三元状态（S，F，G）。状态空间可用图形式表示，图中的每个节点表示一个状态，图中的有向弧表示算符。

一般用状态空间法这一术语来表示下述方法：从某个初始状态开始，每次加一个操作符，递增地建立起操作符的试验序列，直到达到目标状态为止。寻找状态空间的全部过程包括从旧的状态描述产生新的状态描述，以及此后检验这些新的状态描述，看其是否描述了该目标状态。这种检验往往只是查看某个状态是否与给定的目标状态描述相匹配。不过有时还要进行较为复杂的目标测试。对于某些最优化问题，仅仅找到到达目标的任一路径是不够的，还必须找到按某个准则实现最优化的路径。要完成某个问题的状态描述，必须确定三件事：①该状态描述方式，特别是初始状态描述；②操作符集合及其对状态描述的作用；③目标状态描述的特性。

2.2 搜索技术概述

搜索过程[31]就是问题的求解过程。状态空间图的搜索就是在图中寻找从初始节点到目标节点的路径的方法。

图搜索的一般过程如下：

（1）建立一个只含有起始节点 S 的搜索图 G，把 S 放到一个称为 OPEN 的未扩展节点表中。

（2）建立一个叫做 CLOSED 的已扩展节点表，其初始为空表。

（3）LOOP：若 OPEN 表是空表，则失败退出。

（4）选择 OPEN 表上的第一个节点，把它从 OPEN 表移出并放进 CLOSED 表中，称此节点为 n。

（5）若 n 为一目标节点，则有解并成功退出，此解是追踪图 G 中沿着指针从 n 到 S 这条路径而得到的。

（6）扩展节点 n，同时生成不是 n 的祖先的那些后继节点的集合 M。把 M 的这些成员作为 n 的后继节点添加到图 G 中。

（7）对那些未曾在 G 中出现过的 M 成员设置一个通向 n 的指针，把 M 的这些成员加进 OPEN。对已经在 OPEN 或 CLOSED 表上的每个 M 成员，确定是否需要更改通到 n 的指针方向。对已在 CLOSED 表上的每个 M 成员，确定是否需要更改图 G 中通向它的每个后继节点的指针方向。

（8）按某一任意方式或按某个探试值，重排 OPEN 表。

2.3 启发式算法概述

启发式算法[31]由于能够提供一个较好的搜索方向，实际计算性能较好，广泛应用于各种领域。在人工智能领域，主要研究启发式算法的普遍适用性，而其他学科主要针对具体问题提出启发式算法。现在优化算法包括拉格朗日松弛、模拟退火、遗传算法、禁忌搜索、

神经网络等算法，这些算法涉及生物进化、人工智能、数学和物理科学、神经系统等概念，都是以一定直观基础而构造的启发式算法。

启发式搜索方法就是利用问题本身的特性信息（启发信息）来指导搜索前进的方向，其基本思想是：在搜索的每一步都按照启发式信息$f(n)$的大小对 OPEN 表中的所有节点进行排序，最佳的节点排在最前面，因此称为最佳优先搜索法。这种方法找到的路径接近于最佳。通常把用来描述待扩展节点在问题求解中的重要性的函数称为评估函数，记为$f(n)$，表示为：

$$f(n)=g(n)+h(n) \tag{2.2}$$

其中，$g(n)$表示从初始节点 S 到节点 n 的实际代价，$h(n)$表示从 n 经最佳路径到目标节点 S_g 的估计代价，称为启发函数。启发函数的选择在决定搜索算法的启发能力中起着关键作用。$h(n)\equiv 0$时算法类似于广度优先搜索算法，，虽然保证了算法的可纳性，但搜索效率较低，$h(n)$接近于 $h^*(n)$时，所需要扩展的节点越少，其中 $h^*(n)$为从节点 n 到目标节点集合的最小代价。算法的可纳性是指只要从初始节点到目标节点的路径存在，就一定能找到一条最佳路径。当 $h(n)$等于 $h^*(n)$上较低约束的最高可能值时，则所需要扩展的节点最少，并能保持算法的可纳性。当 $h(n)$大于 $h^*(n)$时，所需要扩展的节点就大为减少，但这时不能保证发现最佳路径。

A*算法是一种启发式搜索方法，是一种有序搜索算法，其特点在于对估价函数的定义上，对于一般的有序搜索，总是选择 $f(n)$值最小的节点作为扩展节点。因此，$f(n)$是根据需要找到一条最小代价路径的观点来估算节点的。A*算法的流程图如图 2.1 所示。

A*算法在最短路搜索方面有着广泛的应用，相对于其他搜索方法（盲目搜索，局部最佳搜索等）在搜索速度和效果方面有着明显的优越性。该算法在地图最短路搜索，物流配送系统、交通系统中最佳路径选择等应用方面取得较好的效果[31,32]。与本课题相关的研究领域中启发式算法主要应用于集装箱配载、车辆调度、路径选择等问题。刘士新等对资源受限工程调度问题进行了模型描述，并总结了启发式算法在这类问题中的应用[33]。刘云忠等对车辆路径问题的进行了模型描述，介绍了在这类问题应用的多种启发式算法[34]。

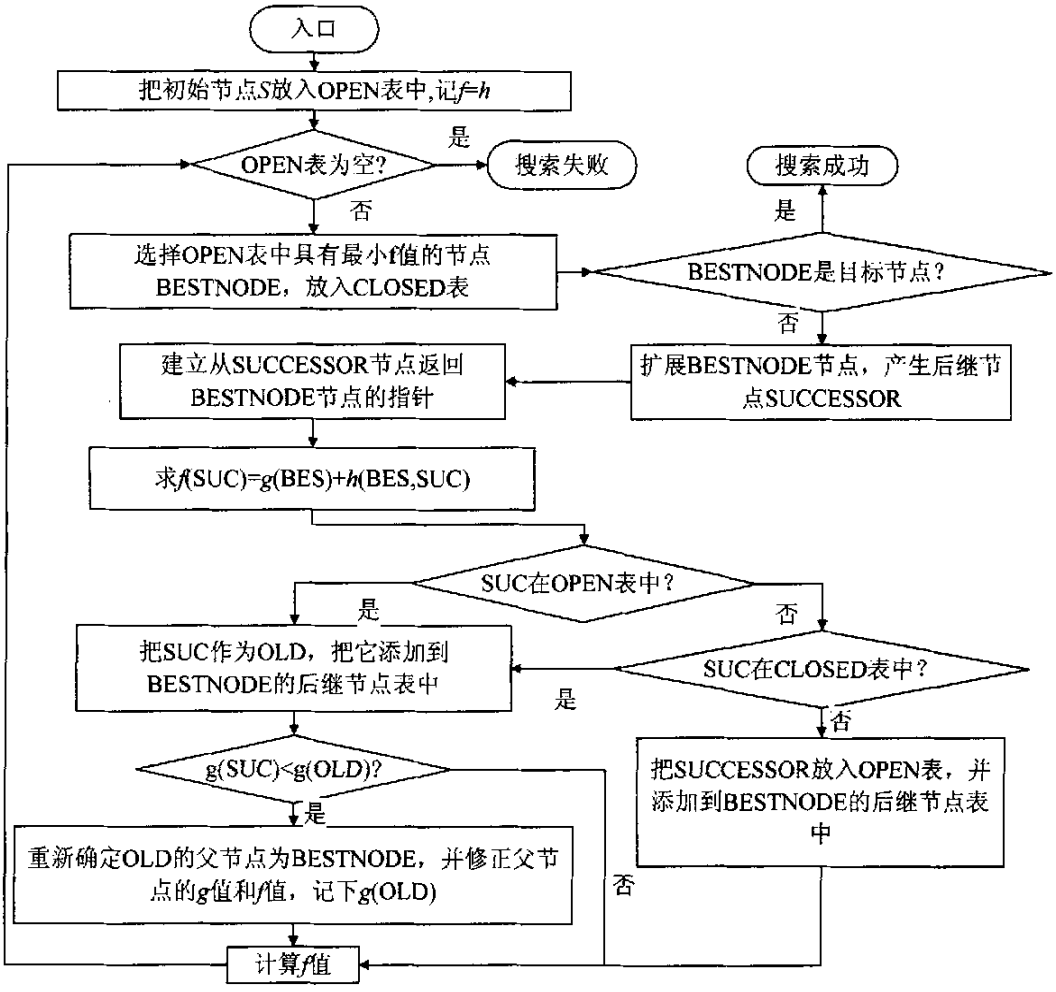

图 2.1 A*算法基本流程图

Fig.2.1 Basic Flow Chart of A* Algorithm

2.4 遗传算法概述

遗传算法是模仿生物遗传学和自然选择机理，通过人工方式构造的一类优化搜索算法，是对生物进化过程进行的一种数学仿真，是进化计算的一种最重要的形式。遗传算法的基本思想是：基于达尔文进化论中的适者生存、优胜劣汰的基本原理，按生物学的方法将问题的求解表示成“种群”，从而构造出一群包括 n 个可行解的种群，将它们置于问题的“环境”中，根据适者生存的原则，对该种群按照遗传学的基本操作，不断优化生成新的种群，这样一代代地不断进化，最后收敛到一个最适应环境的最优个体上，求得问题的最优解。该算法提供了一种求解复杂系统优化问题的通用框架，对问题的种类有很强的鲁棒性，具有智能性、本质并行性、多解性、不确定性、非定向性等特点，现在广泛应用于：函数优

化、组合优化、生产调度问题、自动控制、机器人智能控制、图像处理和模式识别、人工生命、遗传程序设计、机器学习等[35, 32]。

在遗传算法的发展过程中，众多学者针对遗传算法中的不足之处对其进行改进，对编码方式、控制参数的确定、选择方式和交叉机理等进行了深入的研究，提出多种改进方案，如：分层遗传算法、CHC 算法、自适应遗传算法、基于小生境技术的遗传算法、混合遗传算法等。CHC 算法采取跨世代精英选择策略、异物种重组、大变异的方法，强调优良个体的保留[41]。在混合遗传算法研究中，有将遗传算法与模拟退火算法结合，可以解决遗传算法局部搜索能力差，过早收敛的现象[36]。Srinvivas 等提出了一种自适应遗传算法，使得交叉概率和变异概率能够随适应度自动改变[37]。黄聪明等对小生境遗传算法做了改进，算法基于自适应交叉概率算子和变异算子,根据进化代数和群体的适应值，动态调整各个个体的交叉概率和变异概率，并在变异量的确定上引入了梯度的概念[38]。

2.4.1 遗传算法基本概念

（1）种群（Population）和个体（Individual）

遗传算法在求解问题时是从初始给定的多个解开始搜索的，这初始给定的多个解的集合称为一个种群，种群是问题的解空间的一个子集。种群中的每个解称为个体。

（2）染色体（Chromosome）与基因（Gene）、基因组（Gene Group）

对种群中每个个体进行映射变换得到的编码串叫做所谓的染色体，染色体上的每一位叫做基因，染色体上一个有效信息段叫做基因组。

（3）适应度函数（Fitness Function）

适应度函数是种群中各个个体对各自适应环境的程度的一种量度，它通常是用户所提供的目标函数的一个合理的数学变换，而目标函数用能反映个体在种群中优劣程度的数学表达式来担任。

（4）编码（Coding）与解码（Decoding）

许多应用问题的结构很复杂，但可以化为染色体形式表示，这个变换的过程叫做编码。相反地将染色体形式变换为原问题结构的过程称为解码。

（5）选择（Selection）

选择是决定以一定的概率从种群中选取若干对个体的操作。选择的目的是从种群中按一定的标准来选定适合作父代（Parents）的个体，通过杂交后复制（Reproduction）出子代（Child）来。

（6）交叉（Crossover）

交叉是指把两个染色体换组的操作，其用意是将两个个体的染色体的基因进行互换以产生新的后代。

（7）变异（Mutation）

变异是指让遗传因子以一定的概率变化的操作。

2.4.2 遗传算法设计方法

用遗传算法解决问题的基本内容包括编码设计、初始种群产生、适应度函数设计、遗传操作设计、算法参数设计。

（1）编码设计

编码问题是从问题空间到表达空间的映射问题。在进行编码设计时要考虑编码的完备性、健全性、非冗余性。编码方法主要包括二进制编码、Gray 编码、实数编码、浮点数编码、有序串编码、多参数映射编码、可变长编码等。

（2）初始种群产生

产生一定数目的个体组成种群。常用的方法有两种：一是完全随机的产生，适合于问题的解无任何先验知识的情况；二是利用先验知识对问题的解过滤再随机地选择。

（3）适应度函数设计

适应度函数可用目标函数表示，也可以用目标函数变换的方式来定义。适应度函数的设计与遗传算法中的选择操作直接相关，适应度函数设计不当将会导致。在设计适应度函数时应该满足以下条件：单值、连续、非负、最大化；合理、一致性；计算量小；通用性强。由于实际问题的目标函数通常不能满足以上条件，因此需要对目标函数进行转化，另外需要对目标函数的值域作某种映射变化，即适应度的尺度变换，变换方法通常有线性变换、幂函数变换、指数函数变换等。

（4）遗传操作设计

遗传操作包括：选择、交叉和变异三种。

①选择

选择策略对算法性能的影响起到举足轻重的作用，不同的选择策略将导致不同的选择压力，即下一代中父代个体的复制数目的不同分配关系。较大的选择压力使最优个体具有较高的复制数目，从而使得算法的收敛速度较快，但容易出现过早收敛的现象。相对而言，较小的选择压力一般能使群体保持足够的多样性，从而增加了算法收敛到全局最优的概率，但算法的收敛速度较慢。

常用的选择策略有：基于适应值比例的选择（轮盘选择法）、基于排序的选择、基于局部竞争机制的选择、精华保存选择等。

②交叉

交叉是为了能够在下一代产生新的个体，是遗传算法获取新优良个体的最重要的手段。一方面，它使得原来群体中的优良个体的特性能在一定程度上保持，另一方面，它使得算法能探索新的基因空间，从而使新的群体中的个体具有多样性。交叉是遗传算法全局搜索能力的主要算子。交叉通常包括单点交叉、两点交叉、多点交叉、一致交叉、离散交叉、算术交叉。

③变异

变异是对群体中的个体串的某些基因座上的基因值作变动。变异本身是一种局部随机搜索，与选择和交叉结合在一起，保证了遗传算法的有效性，使遗传算法具有局部随机搜索的能力，同时使遗传算法保持种群的多样性，以防止出现非成熟的收敛。变异的方法通常有均匀变异、非一致性变异、自适应变异。

（5）算法参数设计

基本遗传算法运行参数包括：种群大小、染色体长度、进化最大代数、总运行次数、交叉率、变异率等。

2.4.3 遗传算法的求解步骤

遗传算法的基本求解步骤为：

（1）初始化群体；

（2）计算群体上每个个体的适应度值；

（3）按由个体适应度值所决定的某个规则，选择将进行下一代的个体；

（4）按概率进行交叉操作；

（5）按概率进行变异操作；

（6）若没有满足某种停止条件，则转步骤（2），否则进入下一步；

（7）输出群体中适应度值最优的染色体作为问题的满意解或最优解。

算法的停止条件最简单的有两种：①完成了预先给定的进化代数；②群体中的最优个体在连续若干代没有改进或平均适应度在连续若干代基本没有改进。

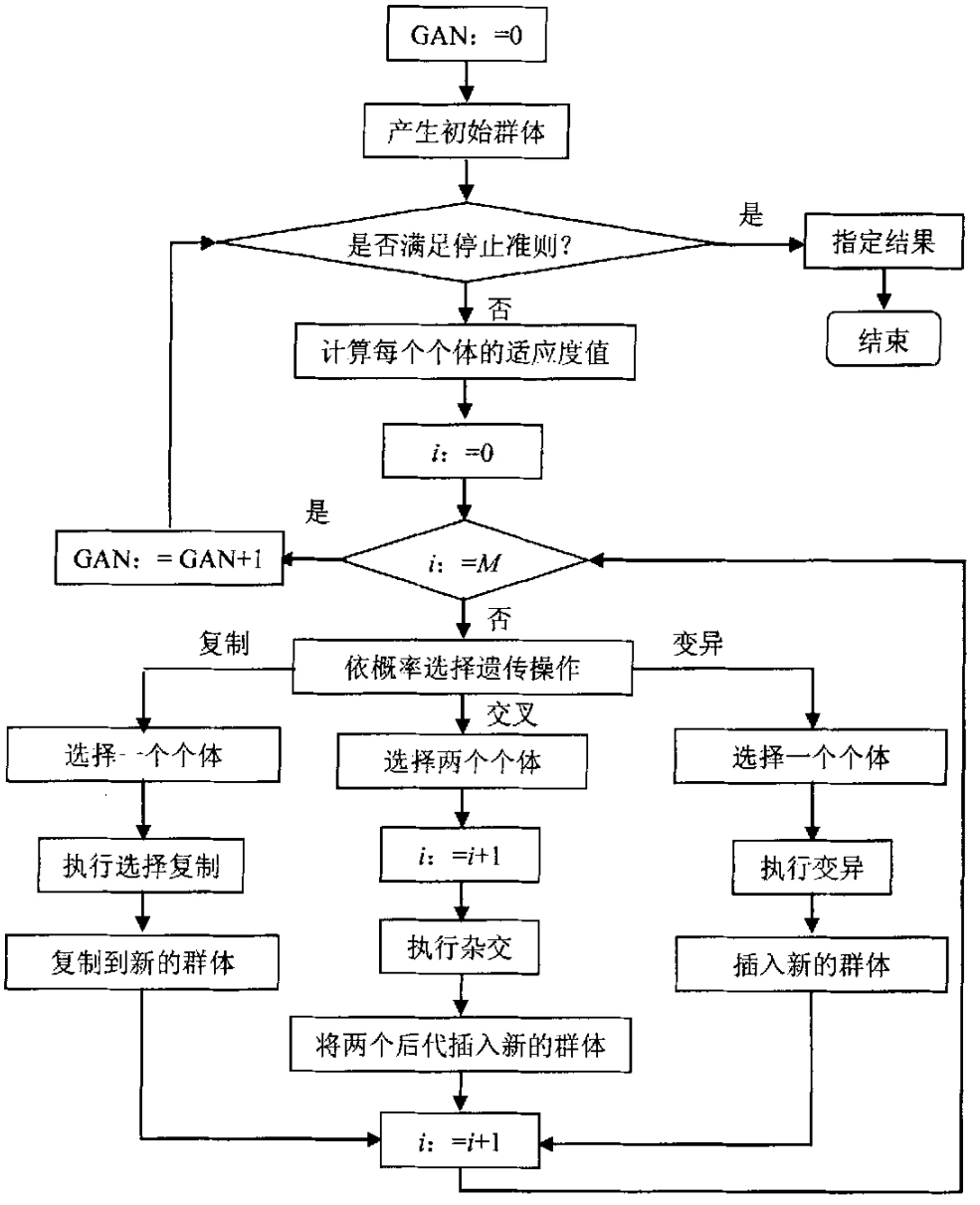

图 2.2 遗传算法基本流程图

Fig.2.2 Basic Flow Chart of GA

2.5 知识表示与应用

对于许多比较复杂的系统和问题，需要借助相应的领域知识来求解。知识的表示在这里起到关键的作用。知识的表示从某种意义上可以看作是数据结构及其处理机制的综合，恰当的数据结构用于存储要解决的问题、可能的中间解答和最终解答以及解决问题涉及之世界的描述，这些都称为符号结构[31, 39, 40]。关于世界的知识可以分为两类：事实和规则。

事实是描述已知的世界状态，是基础性知识，属于静态知识，作为推理的依据。规则是预言外部世界的变化、动作的后果，并从已知的事物演绎未知的事物，是推理的关键。基于规则的问题求解系统通常运用 If→Then 的形式来表示。其中，If 部分可能由几个 if 组成，而 Then 部分也可能由一个或一个以上的 then 组成。在所有基于规则系统中，每个 if 可能与某事实集合中的一个或多个事实匹配，then 部分用于规定放入新事实。在这种系统中每个 if 部分为前提条件或前件，每个 then 部分为结论或后件。表示形式如下：

If　　if　E_1
　　if　E_2
　　⋮
Then　then C_1
　　then C_2
　　⋮

其中 E_1、E_2 为事实，C_1、C_2 为结论。

通常从两个方面来评价知识表示的性能：表示的充分性的表示法效用。表示的充分性指作重要区分和避免不必要的区分的能力，是知识表示的起码要求；表示法效用是表示知识的元素和处理这些元素的操作应能有效的支持使用知识的推理活动，其包括概念效率和计算效率。概念效率指的是知识库应该能够自然的扩充，而不会引起大的变化；计算效率是指推理的有效性，如推理的速度、结论的正确性和有效性等。

基本的知识表示方式有：一阶谓词逻辑、产生式系统、语义网络和框架系统。从知识表示的性能上来看，一阶谓词逻辑具有很好的表示充分性，从而可以应用于各种应用领域，但是知识库较大时推理的效率很低下，因此表示法效用很低；产生式系统中规则是堆积存储，缺乏组织，与一阶谓词逻辑相比表示的充分性降低了，但表示法的概念效率和计算效率相应的提高；语义网络和框架系统虽然克服了前两种表示方式的缺点，但是结构化表示的复杂性，知识库维护代价也相应提高。由此可以看出知识表示的两个方面的性能是相互制约的，往往提高一个方面的性能代价就是牺牲另一方面的性能。所以设计一种表示方式时应根据应用环境和问题特征，对知识表示的这个方面性能做取舍权衡，以能否满足需求为最实用的评价准则。

按知识的作用，计算机处理的知识可分为三类。描述性知识：描述问题求解状况的知识。判断性知识：表示与领域有关的问题求解知识，如推理规则等。过程性知识：表示问题的求解策略，即如何应用判断性知识等进行推理的知识。

在基于知识规则的系统中，除了要具有良好的知识表示方法，还要具有有效的推理解释能力。推理过程就是问题的求解过程，问题求解过程中的每一推理步都要对规则库作穷尽的匹配检查，以选择合适的规则。解释就是把问题的求解过程中激活的使用规则按激活

的次序显示给用户。通常有三种推理方式，即正向推理、逆向推理和双向推理。对于从 if 部分向 then 部分推理的过程，称为正向推理，是从事实或状况向目标或动作进行操作的；反之称为逆向推理，是从目标或动作向事实或状况进行操作的。

专家系统能够利用人类专家的知识与经验处理领域问题。基于规则的专家系统建立在产生式系统的基础上，主要包括知识库、工作存储器、推理机。

知识库以一套规则建立人的长期存储器模型。工作存储器建立人的短期存储器模型，存放问题事实和由规则激发的推断出的新事实。推理机借助于把存放在工作存储器内的问题事实和存放在知识库中的规则结合起来，建立人的推理模型，以推断出新的信息。推理机作为产生式系统模型的推理模块，并把事实与规则的前件进行比较，确定哪条规则能够被激活，通过这些激活的规则，把结论加入工作存储器，并进行处理，直到再没有其他的规则前件与事实相匹配为止。其工作模型如图 2.3 所示。

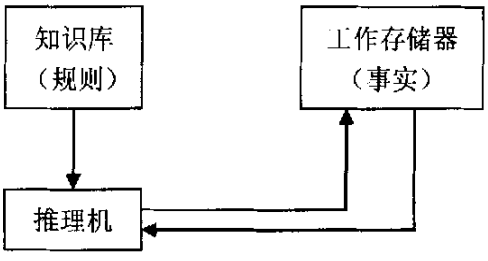

图 2.3 基于规则的工作模型

Fig.2.3 Rule-Based Operation Model

3 堆场作业系统分析及自动分配箱位方法设计

3.1 系统运行环境

本系统嵌入在某物流园集装箱堆场管理系统的大门子系统中。整个集装箱堆场管理系统可分三个层面：操作层面、管理层面、客户层面。管理层负责制定作业计划、调度资源，操作层负责具体执行。图 3.1 中大门子系统为堆场管理系统操作层的一个子功能模块，在外卡进场作业时，需要大门为其提供一个作业成本最低的作业计划。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，也提高了作业成本。在该大门系统中，实际要求外卡的重车通过效率为 60 秒/辆，要求提箱订单处理的响应时间不超过 5 秒，其中包含了作业优化算法的处理时间不超过 2 秒。

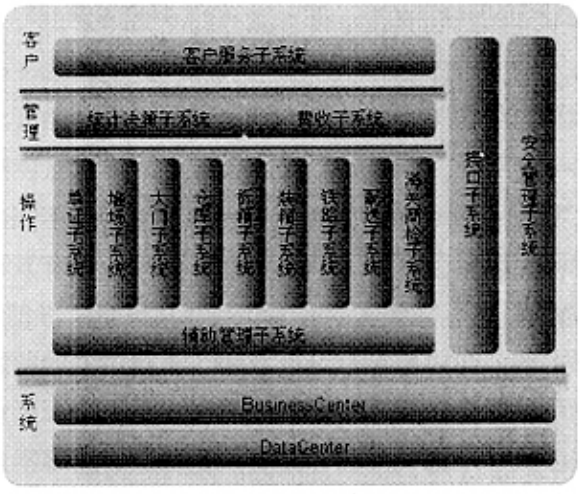

图 3.1 总系统架构

Fig.3.1 General System Architecture

3.2 堆场作业优化系统的结构

堆场作业优化系统的整体结构如图 3.2，该系统主要有 7 部分，系统接口、知识获取模块、知识库、解释模块、数据库、推理机制和算法模块。

系统接口是堆场作业优化系统提供给外部（自动大门）系统的交互接口，用于完成输入输出工作。

知识获取模块的任务是把堆场作业有关规则输入到规则库中，并负责维持规则的一致性及完整性，建立性能良好的规则库。

知识库是知识规则的存储机构，用于存储堆场内部管理所涉及的堆场定义及作业规则及设备作业模式模板等的原理性知识、经验性知识及有关事实，主要包括模板库和规则库。规则来源于知识获取模块，同时它又为推理机提供求解问题所需的知识。知识库管理系统负责对知识库中的知识规则进行组织、检索、维护等。

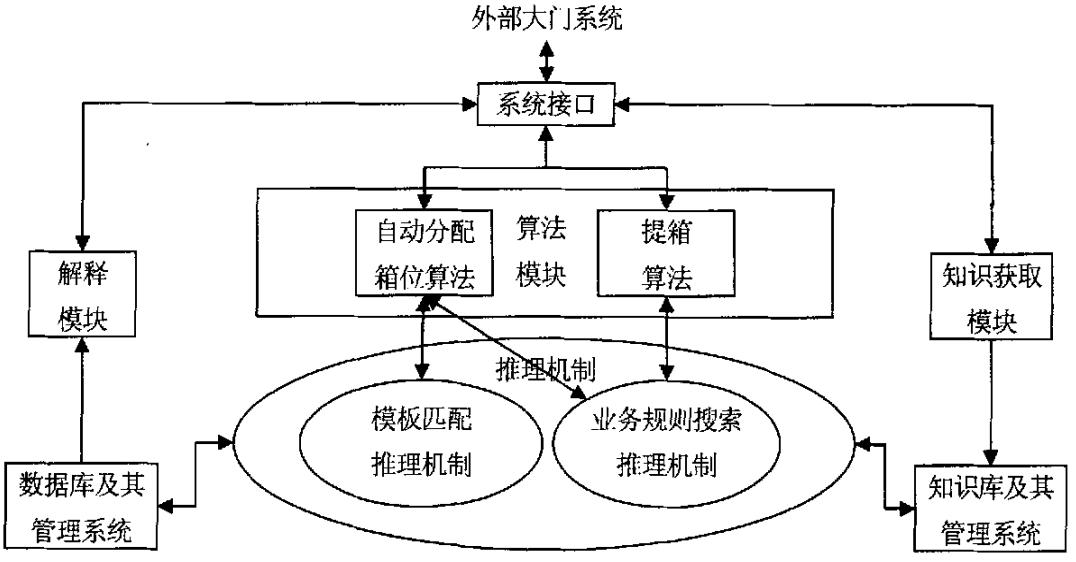

图 3.2 堆场作业优化系统结构图

Fig.3.2 Structure of Stack Operation Optimization System

解释模块跟踪并记录推理过程，当外部大门系统提出询问需要给出解释时，它将根据问题的要求分别作出相应的处理，并把解答通过系统接口输出给外部大门系统。

数据库用于存放堆场中的堆存状态、设备实时位置，堆场中区域、街、贝的相关定义，设备作业能力，作业顺序模板定义等。系统运行过程中得到的中间结果、最终结果、运行信息等也存放在数据库中。

推理机制负责控制并执行对问题的求解，它能根据当前已知事实，利用规则库中的规则，按照一定的推理方法和控制策略进行推理，求得到达堆场的集装箱在某一时刻的堆场状态下所分配的一个位置，根据应用方面和知识的表示不同分成两大类：业务规则搜索和模板匹配。模板匹配服务于分配落箱箱位算法，业务规则搜索服务于提箱和分配落箱箱位两个算法。

算法模块把已知的初始信息提供给相应的推理机制，推理机制通过推理把得到的结论返回给相应的算法模块，算法根据结论得到运行结果返回给外部大门系统。算法模块包括自动分配箱位算法和提箱算法。

3.3 领域知识介绍及其表示

堆场集装箱堆码和搬运作业涉及的知识主要包括堆场、设备的定义，设备在堆场中作业必须遵循的原则，由于这类知识在内容和表达形式上和产生式规则相似，因而采用该种表示方法。上述领域知识可以分别由事实和规则来表示。

3.3.1 集装箱的分类

（1）按规格尺寸分，目前国际上通常使用的干货柜(DRYCONTAINER)有 20 尺集装箱、40 尺集装箱，别外还有应用范围不太广泛的 45 尺集装箱。本文研究内容主要针对 20 尺和 40 尺箱。

（2）按总重分有 30 吨集装箱、20 吨集装箱、10 吨集装箱、5 吨集装箱、2.5 吨集装箱等。

（3）按用途分有干货集装箱、冷冻集装箱(REEFER CONTAINER)、挂衣集装箱(DRESS HANGER CONTAINER)、开顶集装箱(OPENTOP CONTAINER)、框架集装箱(FLAT RACK CONTAINER)、罐式集装箱(TANK CONTAINER)等。本文中所提到的箱型就是按用途进行分类。

3.3.2 集装箱堆场作业设备

港口集装箱堆场装卸搬运设备资源主要包括底盘车、跨运车、叉车、轮胎式龙门起重机、轨道式龙门起重机以及龙门吊，其中一些设备类型可同时用于车辆的装卸作业。[1,2]

（1）底盘车（Chassis）

集装箱堆场的底盘车堆存方式是指将集装箱连同起运输集装箱作用的底盘车一起存放在堆场上，是集卡的拖挂部分。这种堆存方式机动性最大，但是集装箱堆存高度只有一层，而且需要留有较宽的车辆通道，因此需要占用较大的堆场面积，使堆场面积利用率较低。

（2）跨运车（Straddle carrier）

跨运车简称跨车，是一种具有搬运、堆垛、换装等多功能的集装箱专用设备。跨运车承担码头前沿与堆场之间的集装箱水平运输，以有堆场的堆码和进出堆场集装箱拖车的装卸作业。跨运车一般被认为是一种故障率较高的设备，因此维修费用上升。

（3）集装箱叉车（Container forklift）

集装箱叉车是集装箱码头上常用的一种装卸枢机，主要用于吞吐量不大的综合性码头上进行集装箱的装卸、堆垛、短距离的搬运和车辆的装卸作业，也有用于大型集装箱码头堆场的辅助作业。叉车的通用性强，可适用于多货种作业，多用于空箱作业，设备价格较便宜，装卸成本较低。

（4）龙门起重机（Transtainer）

龙门起重机简称龙门吊或场桥，是一种在集装箱场地上进行集装箱堆垛和车辆装卸的设备。龙门起重机有轮胎式和轨道式两种型式。

①轮胎式龙门起重机（Rubber-tired transtainer）：轮胎式龙门起重机主要特点是机动灵活、通用性强，可从一个堆场转向另一个堆场进行作业。

②轨道式龙门起重机（Railmounted transtainer）：轨道式龙门起重机是集装箱码头堆场上进行装卸、搬运和堆垛作业的一种专用设备。一般比轮胎式龙门起重机跨度大，堆垛层数多。轨道式龙门起重机容易实现全自动化装卸，但是只能限制在所设轨道的某一场地范围内进行作业。

（5）正面吊（Front-handling mobile crane）

正面吊是一种目前在集装箱码头堆场上得到越来越频繁使用的专用设备。堆箱层数较高，且可以为多排跨集装箱作业，可以堆存 3-4 层重箱或 7-9 层空箱，因此场地利用率较高，灵活性强，但是需要占用较宽的通道。

3.3.3 堆场作业领域知识表示

（1）事实表示

自动分配箱位算法主要涉及 4 类事实：贝属性事实、堆场堆存状态事实、在场箱信息事实、设备属性事实。

①贝的属性事实指的是堆场中所有贝的自然属性和堆放集装箱的原则的相关定义。包括街名、贝号、最大行数、最大层数、作业模式、作业起始行、作业方向、轻重原则，各属性描述如下：

作业模式。设备在堆场中贝进行作业时的作业方向。包括从小行到大行作业、从大行到小行作业、从中间到两边作业。

作业起始行。当设备从中间到两边作业时，中间行作为划分两边的依据。范围为从 1 行到最大行，取值为 1 行表示从小行到大行作业，取值为最大行号表示从大行到小行作业。

作业方向。设备优先堆放集装箱的方向。取值为优先堆放大行或优先堆放小行。

轻重原则。同一贝同一行的堆垛上堆放集装箱时，上下集装箱堆码的轻重放置原则。取值为重压轻（轻箱在下）或轻压重（重箱在下）。

②堆场堆存状态事实指的是堆场中每个位置的放置状态，位置的状态包括：空位置、已放箱箱号、预分配、封存 4 种情况。

③在场箱信息事实指当前在场箱的基本信息，包括箱号、尺寸、箱型、空重、重量、目的港、货主、箱公司、运输方式。

④设备属性事实指的是设备作业时应当遵循设备本身特点的基本原则，主要包括设备初始位置、设备类型、作业能力和设备堆放顺序。三种类型的设备（场桥、正面吊、叉车）的作业能力和堆放顺序各不相同。设备位置是指设备在堆场中的街贝号，作业能力包括：设备作业的最大高度和宽度。堆放顺序模板包括：模板宽度、模板高度、模板中每一位置的作业顺序号。三种设备由于其机械特点，其作业方式比较如表 3.1 所示。

表 3.1 三种设备作业方式比较

Tab.3.1 Comparison of Operation Mode of Three Operation Equipment

设备类型	正面吊	场桥	叉车
作业方式	可以跨箱作业	直上直下作业	单侧作业
描述	正面吊作业能力通常表示为跨 x 行 y 层是指正面吊最大可以水平方向跨 x 行，垂直方向最高作业 y 层	场桥作业能力通常表示为堆 y 过 $y+1$，是指场桥最高可以提着箱从第 $y+1$ 层通过，箱子最高在第 y 层堆放。落箱分配箱位时，只要不超过场桥作业能力的最大高度和贝的属性中规定的最大层数，可以自上而下落在贝内任意位置；提箱时若存在倒箱则只需倒走目标箱上面压着的箱子即可	只能在街的边侧作业，不可跨箱，而且只能直上直下作业。因此要存在倒箱的情况必须将目标箱的外侧和上面的所有箱移走；落箱时只能放到最外侧且最上面的位置

表 3.2 贝属性实事表示

Tab.3.2 Fact Description of Bay Attribute

街号	贝号	最大行数	最大层数	作业模式	作业起始行	作业方向	轻重原则
A01	001	6	5	小到大	1	先大行	重压轻
B01	003	4	4	大到小	4	先小行	轻压重
A02	005	6	6	中间	2	先大行	重压轻

事实在系统中存放于数据库中，它们均以关系表的形式存储。如表 3.2、表 3.3、表 3.4 所示，贝属性事实、堆场堆存状态事实和在场箱信息事实的存储形式相对简单，设备的作业能力和作业模板顺序的存储形式对推理的效率较为关键。

表 3.3 堆场堆存状态事实表示

Tab.3.3 Fact Depiction of Stack Storing Status

街号	贝号	行号	层号	状态
A01	001	4	2	空位置
A01	003	4	1	T01
A01	005	6	3	预分配
A02	005	6	3	封存

表 3.4 在场箱信息实事表示

Tab.3.4 Fact Depiction of Container Information on a Dtack

箱号	尺寸	箱型	空重	重量	目的港	货主	箱公司	运输方式
T01	20	GP	空	0T	大连	一汽大众	中集	公路
T02	40	FR	重	5T	天津	长春一汽	中集	铁路

表 3.5 正面吊作业模式事实表示

Tab.3.5 Operation Mode of Front-handling Mobile Crane

设备编号	能力跨行	能力跨层	目标位置 T 所在层数	约束行 1 高度	约束行 2 高度
20031001	3	3	1	1	1
20031001	3	3	2	2	1
20031001	3	3	2	0	2
20031001	3	3	3	1	2

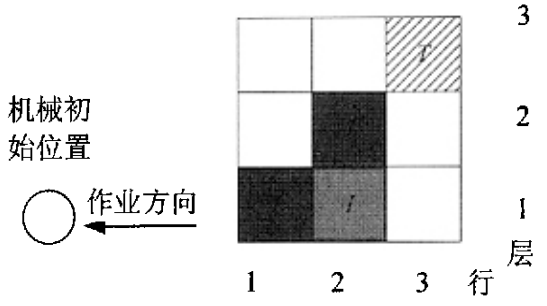

图 3.3 正面吊作业模式图

Fig.3.3 Operation Mode Diagram of Front-handling Mobile Crane

设备属性事实的表示以正面吊为例。正面吊作业能力在数据库中用关系二维表表示，以跨 3 行 3 层为例，其结构设计如表 3.5 所示，以表中第 4 行数据为例，该行数据可用图 3.3 来表示，中的位置 T 表示为将被作业的位置在第 3 层，如果 T 可以被作业，则约束行 1 最高只能第一层有箱，约束行 2 最高只能第 2 层有箱。

（2）规则表示

在本系统中可以分两大类规则：一类是设备作业顺序规则，与设备作业能力有关，属过程性知识；另一类是堆场集装箱堆码和搬运作业业务规则，属判断性知识。

图 3.4 为三种设备作业顺序规则的模板表示。根据设备的作业能力，可用 m×n(m 为模板宽度，n 为模板高度)的矩阵来表示作业顺序，称为设备作业顺序模板。矩阵中存放 1, 2…m×n 的自然数，作为设备对堆场中位置的操作顺序号，外部系统可以设定模板 m、n 值，以及模板中各个位置的序号。

叉车模板：m=1，n 取值为叉车作业能力最高层，模板中序号即位置所在层号。叉车的作业特点决定了只有这样一种垂直作业的模板。

正面吊模板：m=1,2…正面吊可跨过的最大行数，n 取值为正面吊作业能力最高层。典型的模板序号有垂直作业和混合作业。

场桥模板：m=1,2…场桥作业可跨最大行数，典型的模板序号有水平作业、垂直作业和混合作业。

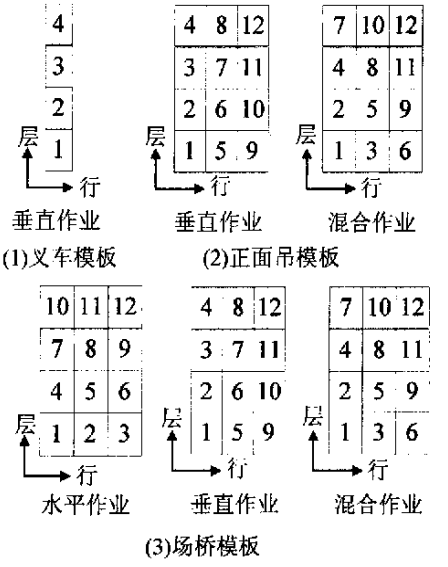

图 3.4 三种设备的作业顺序模板的规则表示

Fig.3.4 Operation Ability and Stacking Sequence rule for Three Kinds of Operation Facility

堆场集装箱堆码和搬运作业业务规则中的各类规则按本文第二章第五节中描述的规则的一般表示形式，表示举例如下表 3.6。

表 3.6 自动分配箱位问题中的各类规则举例

Tab.3.6 Every Kind of Rule Example of Container Automated Allocation Problem

编号	规则类型	规则表示
R1	分组规则	IF 尺寸=20 尺 AND 箱型=普通干货箱 AND 目的港=天津 THEN 箱位分配区域=A01001-A01007
R2	判断某位置的可计划性规则	IF 位置状态=空 AND 位置层号=1 THEN 该位置为计划位置
R3	判断某位置的设备可操作性规则	IF 位置为计划位置 AND 设备为叉车 AND 该位置到设备之间各行不放箱 AND 位置所在层<=叉车作业能力 THEN 该位置设备可操作
R4	判断某位置是否满足重量等级要求规则	IF 该位置设备可操作 AND 该位置在第 1 层 THEN 该位置满足重量等级要求

分组规则是指给到达大门的进场集装箱分配一个堆放区域的原则。堆场通常会根据集装箱的自身属性划分成多个区域，根据划分依据的不同，区域的大小也不同，大的可以包括几条街，小的可能是在一条街内被分成多个区域。因此在对分组规则进行表示时，其规则前件为在场箱信息实事的组合，后件为堆场内某一区域的定义。区域的定义是指堆场中的由街号贝号来划分的一个范围。

位置的可计划性表明某位置是否可以划为计划位置，计划位置指的是堆存状态符合放箱要求的位置，即位置状态为空且在第一层，或者位置状态为空且不在第一层，但位置下方有箱或者预分配。

位置的设备可操作性表明某位置在某一堆场状态下，特定的设备是否能将集装箱放至该处。判断某位置的设备可操作性，需要根据 3 种设备的不同作业特点、堆场的堆存情况来得出结论。

位置是否满足重量等级要求，是指需要放的箱和待判断的位置下方箱的重量等级相比较，看是否能够符合贝属性中的“轻重原则”。

3.4 自动分配箱位问题分析

对于要装船的集装箱进入堆场后，需要为其分配堆场中的临时放置箱位等待装船，船舱配载对集装箱重量等级有一定的要求。理想情况下，堆场中集装箱按照配载图摆放，设备可以依次将集装箱放入船中。然而，载入期内不同重量等级的集装箱到港时间是随机的，产生完全符合装船配载要求的箱位分配结果较困难。而通过等待全部集装箱到达后重新分配位置来产生完全符合配载要求的集装箱堆场的方法，由于堆场面积有限，很难给到达同

一目的港的同类箱组划出完整连续的区域堆放，也难以给到达箱预留缓冲区空间，试图累积到一定数量后重新优化堆垛设计，而且，从成本角度考虑这样做也是不经济的。

自动化大门的自动分配箱位问题就是依照堆场管理的各种原则为要进场集装箱分配一个最优落箱箱位，使得随机到达的集装箱的堆垛符合重量等级堆码要求，方便装船作业，同时降低堆场作业的成本。

因此，对所有进场箱的箱位分配问题实现全局优化的难度较大，本文针对某一时刻到达大门的进场箱信息和堆场信息，根据作业规则和堆场作业设备信息，为集装箱在堆场范围内寻找最优放箱位置。在不倒箱的前提下，优化结果在时间上是局部、瞬时的，在空间上是全局的。

进场集装箱到达自动化大门时，自动化大门自动分配落箱箱位的作业流程为：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自动分配箱位作业计划算法，得到最优目标箱位；③系统打印操作条，给集卡司机；④集卡司机按操作条指示到指定位置；⑤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。

自动分配箱位作业优化的问题模型的前提假设：

①每次需要放的集装箱数量可能为 1 个（20 尺或 40 尺）或 2 个（20 尺）；

②对于 1 个箱只划入 1 个堆场区域，区域中的贝可能在不同街区，同街区内也未必是连续的贝位；

③每个街区只有 1 种类型设备负责该街区的全部作业任务；

④同街区内的所有贝的作业模式相同；

⑤在制订自动分配落箱箱位作业计划期间内，该街内没有其它集装箱进出，即保证堆场状态不会因外部环境而改变。

由上述分析可知，箱位分配问题中涉及的因素之间关系复杂，建立解析模型求解难度较大。基于知识的方法无需系统的定量数学模型，充分利用领域知识和对象的信息，适合于箱位分配问题的系统建模。

3.5 基于规则的自动分配箱位方法的面向对象设计

面向对象技术被认为是计算机科学领域的导向技术之一，它从认识论的观点出发，比较符合人类的思维方式。面向对象的知识表示方法有利于知识的共享和重用，是构造大型知识库的有效手段。

3.5.1 事实与规则类定义

堆场作业问题中的知识（事实和规则）都可用类来表示，事实类 Fact 和规则类 Rule 用 C++定义分别如图 3.4 所示：

```
Class Fact
{                                   //事实类
    private:
    string facts;                   //事实描述
    Fact* next;
    public:
    Fact();
    Fact(string ,Fact*);
    ~Fact();
    string getFact();               //读取事实
    void modify(string);            //修改事实
};
```

```
Class Rule
{                                   //规则类
    private:
        string result;              //规则结论
        bool lastFlag;              //结论性规则标志
        Fact* antecedent;           //规则前件
        Rule* next;
    public:
        Rule();
        Rule(string, bool, Fact*, Rule*);
        ~Rule();
        string getResult();         //读取规则结论
        Fact* getAntecedent();      //读取规则前件
        void modify(stirng,Fact*);//修改规则
};
```

图 3.5 事实类和规则类的 C++定义

Fig.3.5 C++ Language Definition of Fact Class and Rule Class

以这两个类为基础，本系统构造如下几个链表，作为用于推理过程的主要数据结构：

规则库链表（RuleList）：每个节点为一条规则，整个链表构成规则库；

已知事实链表（FactList）：每个节点为一个事实，整个链表构成综合数据库；

结论事实链表（Conclusion）：每个节点的事实都为匹配成功的规则的结论，并且与已知事实不同；

已使用规则链表（UsedList）：每个节点为一条规则，且已经匹配成功。

3.5.2 推理机制

图 3.6 表示基于规则的箱位分配系统的工作过程，该模型可以被描述为一个设备放箱作业的模拟过程：集装箱到达堆场；根据集装箱属性，按照分组规则将其划入某一计划区域；对于区域中每个街区，都有相应设备负责该街区作业；设备提取需要放的箱到某个贝，首先读取负责该贝作业的设备属性和堆场状态信息，将设备作业顺序模板和堆场实时状态匹配，提出箱位分配方案；接着通过规则搜索，验证方案的可行性，如果方案可行，把该方

案放到初始方案表中待评价；然后设备移动到区域中下一个贝，反复上述流程，直到所有贝匹配完毕；评价选优阶段比较所有初始方案中的待选位置，得到最终箱位分配方案。其中关键过程是模板匹配、业务规则搜索和方案评价。

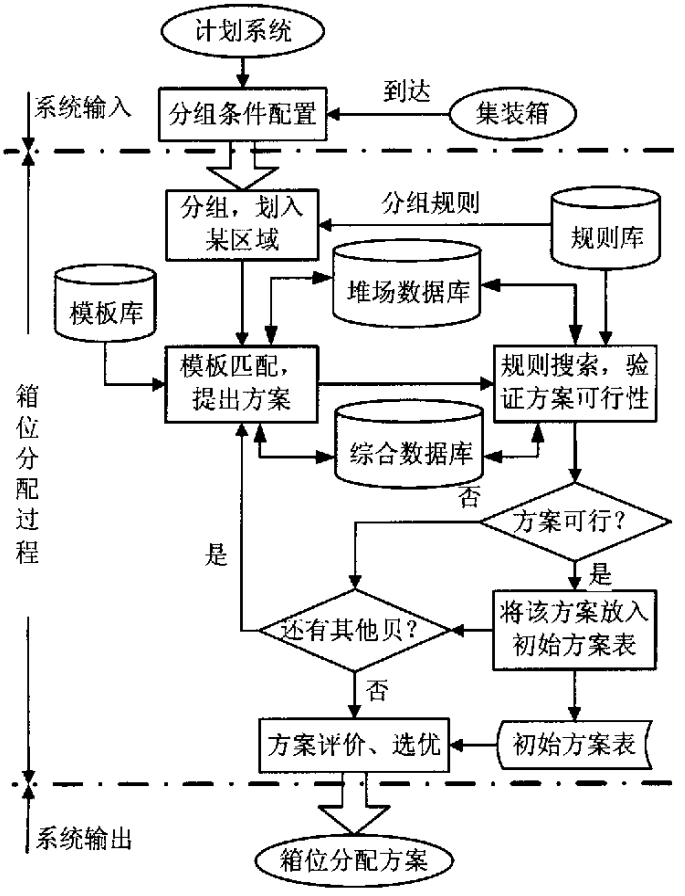

图 3.6　基于规则的箱位分配系统的流程图

Fig.3.6 Process of Rule-Based Storage Space Allocation System

在本章开始时提到的本系统的知识库分为模板库和规则库，因此箱位分配的推理按知识表示形式的不同，也分为模板的匹配和业务规则的搜索两部分。模板的匹配主要指设备作业顺序模板的匹配。业务规则搜索主要针对堆场的作业业务规则和设备作业能力规则进行搜索，按照其相应的作业规则判断某个位置是否可以设备被作业，是否是计划位置，是否满足重量等级要求等。

（1）模板匹配

集装箱划入某区域后，该区域中所有贝和负责每个贝作业的设备随之确定，模板匹配是指将设备的作业顺序模板和堆场实际堆存状态相对比，得出相应箱位分配方案。模板匹

配时，在作业方向上需遵循贝的作业方向，在作业顺序上需遵循设备自身的作业顺序模板。贝的作业方向分为单侧作业和双侧作业，单侧作业是从贝的一侧开始，逐行往另一侧放箱，包括设备从小行到大行作业和从大行到小行作业 2 种情况；双侧作业指设备从贝的中间某行开始，先往一侧作业，完毕后再往另一侧作业。

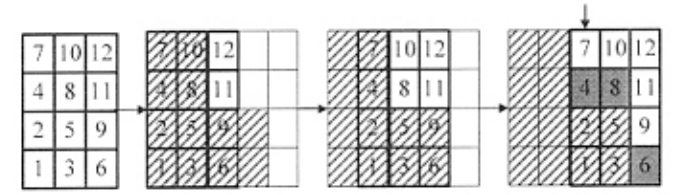

图 3.7 模板匹配过程（小行到大行）

Fig.3.7 Process of Template Match

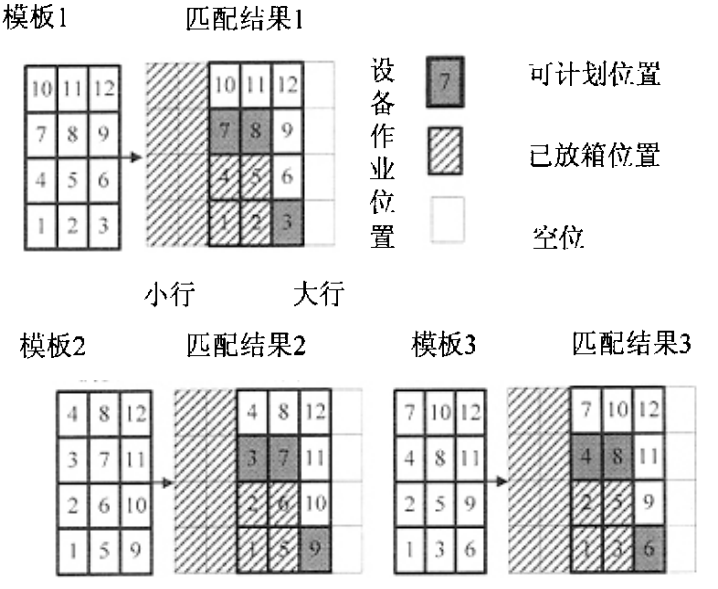

图 3.8 从小行到大行的同一设备的不同模板匹配

Fig.3.8 Different Template Match Results of One Facility

①单侧作业：在这种模式下，设备位置始终在贝的一侧。图 3.7 显示了贝的作业方向为“从小行到大行”，一个模板的匹配过程，最初，模板最小序号所在行从作业起始行开始，逐行寻找未放满箱的一行，使该行和模板最小序号所在行对齐，模板套住范围(图中加粗部

分)的所有位置被赋予相应的模板作业序号，每个被赋予序号的位置是一个备选方案。图 3.8 例举了同一设备的不同模板和某堆场状态相匹配所产生的多种方案。

贝的作业方向为“从大行到小行”下的模板匹配，首先将模板对称翻转，从贝的最大行开始，寻找该贝未放满箱的一行，使该行和模板最右边对齐，模板套住区域的所有位置被赋予相应的模板作业序号。

②双侧作业：其原理和单侧作业相同，但匹配模板的起始行不是贝的最小行或最大行，而是中间某一行。在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）业务规则搜索

在箱位分配模型中，经过作业顺序模板匹配后，每个被赋予作业模板序号的位置形成一个放箱方案，业务规则搜索验证方案的可行性，宜采用逆向推理方式，对放箱方案中的箱位从规则库中选择业务规则，按序号由小到大的顺序逐个判断其是否可以放箱，一旦成功，此轮搜索结束。其推理过程如图 3.9 所示。

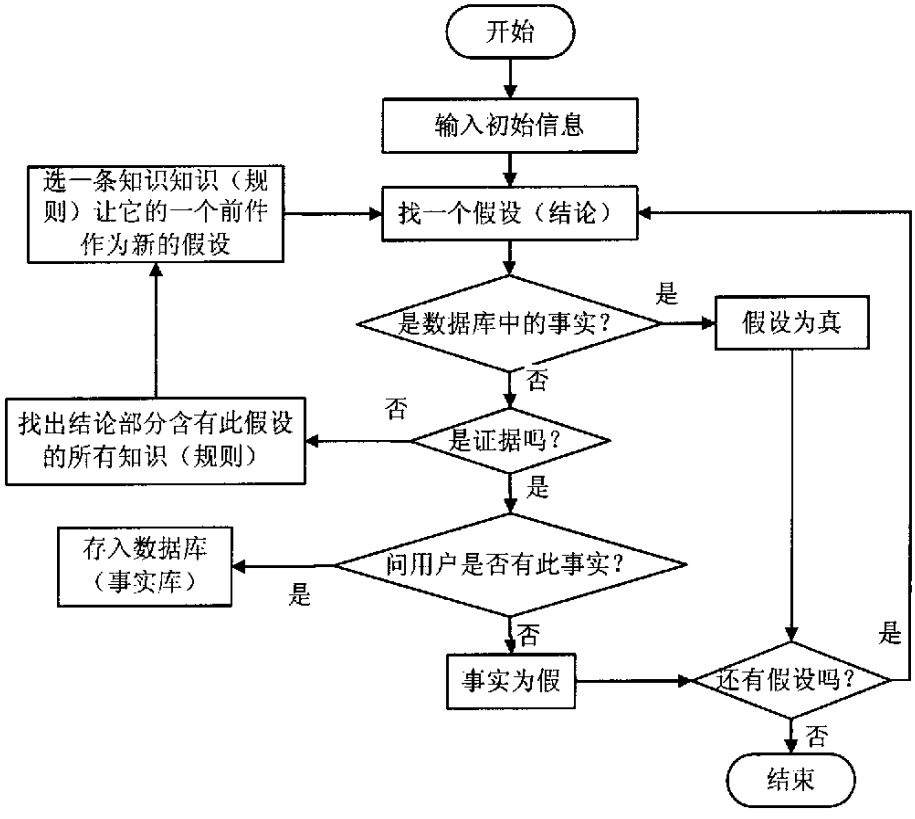

图 3.9 推理流程图

Fig.3.9 Reasoning Flow Chart

以图 3.8 中“匹配结果 2” 为例，模板匹配后，每个位置被赋予相应的作业顺序号。

⑴判断所有位置的可计划性，由于 1、2、5、6 已经放箱，不可计划；4、8、10、11、12 为空，但其下方没有箱，位置悬空，不可计划；其余 3、7、9 可作为计划方案。

⑵判断设备对位置的可操作性。若设备为场桥，3 个位置均可作业，因而均可作计划；若设备为叉车(最高 5 层)，位置 3 不可作业，因而只有 7 和 9 可作为计划方案。

⑶判断需要放的箱和位置下方箱的重量，是否满足重量等级要求。假设步骤⑵设备为场桥，若待放箱重量大于位置 2 的箱重量，则位置 3 可作为计划方案；若待放箱重量大于位置 6 的箱重量，则位置 7 可作为计划方案。

（3）方案优劣评价

所谓方案优劣，是指以贝为单位，集装箱的堆放状态是否有利于后续操作，包括该贝的后续箱位分配作业和该贝放满后的装船作业。评价方案的优劣，最重要的指标是：

①集装箱的堆放是否严格按作业顺序模板进行；

②堆垛形成的集装箱重量等级分布是否合理。

对指标①，在模板匹配的时候可以记录下放箱方案中待选择位置，计算该位置序号和最优位置序号之差，值越大，方案越差。对指标②，在规则搜索时记录待放箱和目标位置下方箱的重量等级之差，值越大，方案越差。我们采用公式 3.1 表示方案的优劣：

$$\min \text{opt} = a(N_{obj} - N_{min}) + b(W_{obj} - W_{under}) \tag{3.1}$$

其中，N_{obj}：目标位置模板序号；N_{min}：模板匹配得到可行方案的最小模板序号；W_{obj}：待放箱重量等级；W_{under}：目标位置下方箱重量等级；a、b：均为常量，分别代表指标①和指标②在模型中的权重。opt 值最小的为最优箱位，若 opt 最小值位置有多个，取设备移动距离最小的位置为最优箱位，可用公式 3.2 表示。

$$\min \text{move} = | \text{bay}_{fac} - \text{bay}_{opt_i} | \tag{3.2}$$

其中，pos_{fac} 为设备所在贝，pos_{opti} 为 opt 值最小的目标箱位所在贝。

4 提箱作业优化算法设计

4.1 提箱作业问题模型

提箱作业是大门处作业的重要组成部分。空集卡到达大门，提交给大门提箱订单，大门根据提箱订单为集卡提供提箱作业计划，集卡根据大门提供的提箱作业计划进场提箱。提箱作业计划是指堆场中要提取某个集装箱的全部倒箱作业步骤的方案，包括倒箱的顺序及待倒箱的放箱目标位置。本文研究的提箱作业优化算法是指在制定提箱作业计划时对倒箱作业步骤方案进行优化，从而得到倒箱作业的最优的放箱目标位置，以减少倒箱成本或提高倒箱效率。

为了保证堆场信息、系统数据库、算法内部数据的一致性，图 4.1 表示的是提箱算法的被调用过程及与系统实时状态的关系。

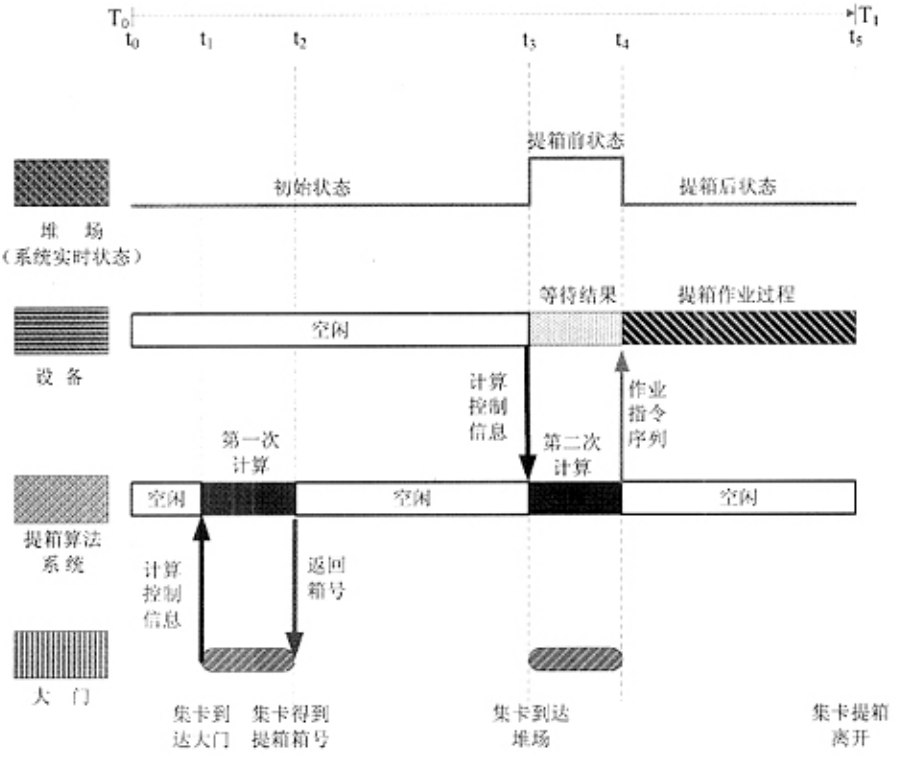

图 4.1 时序图和作业过程图

Fig.4.1 Time Sequence and Operation Process of System

4.1.1 前提条件

提箱作业是一个复杂的作业问题，涉及很多制约因素，因此为了简化问题，突出问题的核心部分，本文在进行提箱作业优化时设定以下的前提条件：

（1）在堆场中的一条街中的给定几个贝的范围内进行提箱作业，在给定的范围中不考虑放箱中的分组规则的因素。

（2）制定提箱作业计划时必须事先指定一种类型的堆场作业设备进行作业，该设备类型为待提箱所在街的负责提箱作业任务的设备类型。

（3）在制定提箱作业计划时，以待提箱所在街的当前状态为基准，假设在制定提箱作业计划过程中，该街内没有其它集装箱进出，保证街的状态在提箱过程中除了进行当前待提箱的倒箱作业外不会被其他作业而改变。

（4）当倒箱位置不够时，把待倒箱放到缓冲区中，提出待提箱后，再把待倒箱还原，当把待倒箱放到缓冲区时，此方案设为最差方案，因此除非确实找不到倒箱位置，才考虑使用缓冲区。

4.1.2 模型描述

（1）符号定义

模型中的主要符号定义如下：

b：街的最大贝数；

r：街的最大行数；

t：街的最大层数；

n：全部待提箱个数；

$s_i\,(sb_i,sr_i,st_i)$：待倒箱 i 的原始位置，(sb_i,sr_i,st_i)为其在街中的位置坐标；

S：全部待倒箱的原始位置的集合，$S=\{s_1,s_2,s_3,\ldots,s_n\}$，按倒箱先后顺序排列；

$d_{ij}\,(db_{ij},dr_{ij},dt_{ij})$：第 i 个倒箱阶段时，待倒箱 i 的可以一个放置的位置，$(db_{ij},dr_{ij},dt_{ij})$ 为其在街中的位置坐标；

D_i：进行第 i 个倒箱阶段时，满足作业规则的可以放待倒箱 i 的目标位置集合，$D_i=\{d_{i1},d_{i2},d_{i3},\ldots,d_{ij},\ldots,d_{im}\}$；

H_i：第 i 个倒箱阶段，把待倒箱 i 从 s_i 倒至 d_{ij} 的作业最小成本（重车作业）；

R_i：第 i 个倒箱阶段，设备空车从 d_{ij} 移动到 s_{i+1} 返回最小成本（空车作业）；

C：全部倒箱作业的总成本 $\sum(H_i+R_i)$；

X：表示模型的一组可行解，$X=\{d_{1k_1},d_{2k_2},\ldots,d_{nk_n}\}$，$d_{ik_i}$ 为表示待倒箱 i 最终被放置的位置；

B_i：表示第 i 个倒箱阶段，街上集装箱的堆放状态集合，即街中每个位置的放箱情况，$B_i=\{p_1,p_2,...,p_j,...\},1\le j\le b\times r\times t$，每个位置 p_j 由四个参数表示(e_j,b_j,r_j,t_j)，e_j 表示该位置是否已放箱，b_j、r_j、t_j 分别为该位置的坐标。

（2）模型的描述

在求提箱方案时，可以根据待提箱的所在位置、街的初始堆放状态及设备作业能力，可以得到 n 和 S，对每个待倒箱进行的倒箱过程如下：

①倒第 i 个箱时，当前街的状态为 B_i，把第 i 个箱从原位置 s_i 放到 d_{ij} 位置，得到作业成本 H_i，当前街的堆放状态被改变；

②设备空车从 d_{ij} 返回到下一个要倒箱的所在位置 s_{i+1}，得到返回成本 R_i。

如图 4.2 所示，由于设备在移动集装箱时的路径有多种选择，且移动方向不同、设备重车（带集装箱）和空车（不带集装箱）移动单位距离时成本也不同，进行优化时，需要考虑作业成本最小的移动路径，因此 H_i 和 R_i 应为每次移动最小成本。在这个过程中，由于设备作业能力的制约，前一个箱的放置 d_{ij} 使街的状态发生改变，因此需要重新得到下一个待倒箱的可放位置的集合 D_{i+1}，即 d_{ij} 选择的不同，D_{i+1} 也不同，这使问题具有多组合性及动态性。因此对于提箱过程中的倒箱优化是一个多阶段决策问题，把每次倒走一个箱并且设备空车返回下一倒箱所在位置这个过程看作一个阶段，求使提出待提箱的总成本 C 最小的一个解 X，即求每个待倒箱的最终放置位置。

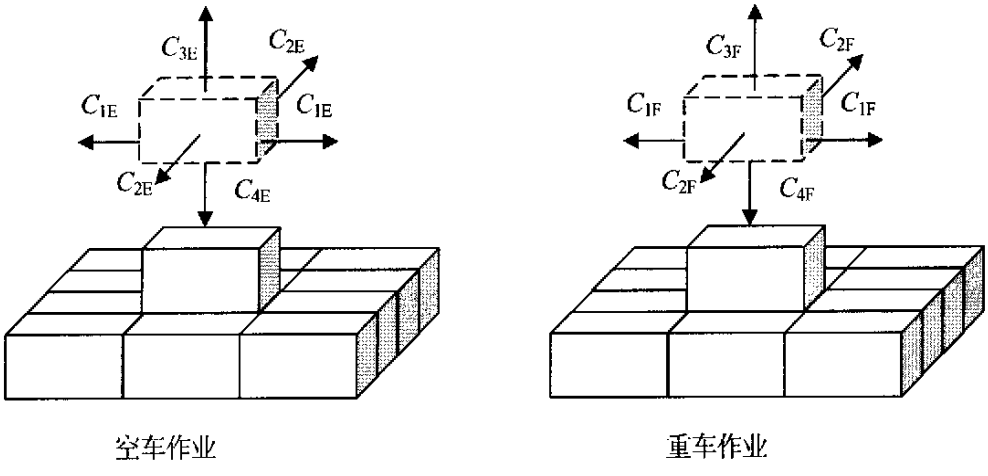

图 4.2 设备移动方向和成本图

Fig.4.2 Move Direction and Cost of Facility

在倒箱优化多阶段决策问题的状态空间中，倒第 i 个箱时，设该问题的当前状态为 $Q_i=[B_i,s_i,D_i]^T$。状态转移算符可以表示为公式 4.1。$T_i(D_i,d_{ij})$表示从 D_i 中选择一个 d_{ij} 为目标位置放待倒箱，并更新街的堆放状态为 B_{i+1}。

$$B_{i+1}=T_i(D_i,d_{ij}) \tag{4.1}$$

对于同一种设备，当移动一个箱或空车移动时，街的位置状态可以用一个有向图表示。在这个有向图中，每个节点对应街的一个位置，边表示从两个相邻节点路径，边的权值表示移动的成本。在求 H_i 和 R_i 时，就是在该有向图中找出从初始位置节点到目标位置节点的一条最短路径。

由此可以看出，整个提箱作业计划模型实际是由两个子模型组成，一是倒箱优化模型，一是倒箱过程中最短路搜索模型，其中，最短路搜索嵌套在倒箱优化模型中。

4.1.3 提箱作业规则描述及约束条件

实际提箱作业中，所考虑的制约因素通过规则来描述，从而确定模型的可行解中的位置应该满足的约束条件，主要规则如下：

规则 1：设备（叉车、正面吊、场桥）类型不同其作业方式（移动路径）不同，在街的状态相同的情况下能够作业的位置也不同。倒箱时的目标位置及设备移动路径必须是设备能够作业的。

规则 2：20 尺箱和 40 尺箱不能放在同一贝上。

规则 3：对于不可压的箱型（超限箱、危品箱、开顶箱、框架箱等），上面不可以再放集装箱。

规则 4：一个提箱订单中的所有箱被称为预约箱，在倒箱时目标位置下面有预约箱时不可放箱。

规则 5：倒箱时的目标箱位不能悬空，也不可以有其它箱。

规则 6：在规定的倒箱范围内进行倒箱。在进行倒箱时，放箱的位置不能超过规定倒箱范围的贝、街定义的最大行和最大层。

4.1.4 模型表示

提箱优化模型可由公式 4.2 表示。其中，约束中 f_1、f_2、f_3、f_4、f_5 分别表示根据 4.1.3 中的作业规则得到的规则约束函数，在求 H_i 和 R_i 时的 g 为作业成本最小值计算函数，是一个作业最短路搜索模型，其目标函数可由公式 4.3 表示。其中，$C_{1E(F)}$、$C_{2E(F)}$、$C_{3E(F)}$、$C_{4E(F)}$ 分别为设备空车（重车）移动一贝、一行、上移一层、下移一层时作业成本的权值。

Mu_i、Md_i、Mr_i、Mb_i分别为设备作业时从初始位置移动到目标位置的路径 i 的上移、下移、移动行和移动贝的位移量。

$$
\begin{aligned}
\min \quad & C=\sum_{i}^{n}\left(H_i+R_i\right) \\
& H_i=g\left(sb_i,sr_i,st_i,db_i,dr_i,dt_i\right) \\
& R_i=g\left(db_i,dr_i,dt_i,sb_{i+1},sr_{i+1},st_{i+1}\right) \\
& f_1\left(db_i,dr_i,dt_i\right)=1 \\
& f_2\left(sb_i,sr_i,st_i,db_i,dr_i,dt_i\right)=1 \\
s.t. \quad & f_3\left(db_i,dr_i,dt_i\right)=1 \\
& f_4\left(db_i,dr_i,dt_i\right)=1 \\
& f_5\left(db_i,dr_i,dt_i\right)=1 \\
& db_i\le b,dr_i\le r,dt_i\le t \\
& i\in\left(1,2,3,...,n\right)
\end{aligned}
\tag{4.2}
$$

$$
\begin{aligned}
\min \quad & H_i=C_{1F}Mb_i+C_{2F}Mr_i+C_{3F}Mu_i+C_{4F}Md_i \\
\min \quad & R_i=C_{1E}Mb_i+C_{2E}Mr_i+C_{3E}Mu_i+C_{4E}Md_i
\end{aligned}
\tag{4.3}
$$

4.2 提箱算法设计

4.2.1 A*算法设计

针对以上分析所得两层结构的模型，在此提出了如图 4.3 所示的两层嵌套的 A*算法。该算法中外层 A*算法用于求解倒箱优化模型，对倒箱优化多阶段决策问题的状态空间进行搜索，内层 A*算法求解最短路搜索模型，对作业最短路问题的状态状态空间进行搜索。

（1）倒箱优化模型的多阶段决策 A*算法的设计

倒箱优化模型的多阶段决策 A*算法，是整个提箱作业计划算法的核心部分。

对于具有 n 个待倒箱的倒箱优化模型，其对应问题的状态空间可以看作如图 4.4 所示的一个 n 层的树型结构，表示了 n 个倒箱阶段，为了便于搜索，设一个虚拟的初始节点作为树的根节点 S，表示为第 0 层。此根节点表示搜索的开始。

在图 4.4 所示的树型结构中除根节点以外的其它节点对应一个可行解 X 中的一个分量，它应该保存三个信息：①在树型结构中当前所处的层数，对应一个倒箱阶段；②可行解 X 中的分量在街中的位置坐标，此位置应符合通过从知识库提取规则经过推理机制而得到的前述作业规则约束条件；③指向父节点的指针。假设当前节点为 N(图中 $d_{j_1j_2\cdots j_i}$)，在树型结构中的第 i 层，根据父节点指针可以得到从根节点到 N 的唯一的一条路径，表示为

S-N_1-N_2-…-N，此路径表示把第 1 个要倒的箱放到 N_1(图中d_{j_1})所对应的街中的位置，第 2 个要倒的箱放到 N_2(图中$d_{j_1j_2}$)所对应的街中的位置，…，第 i 个要倒的箱放到 N 所对应的街中的位置，可行解为 $X=\{d_{j_1},d_{j_1j_2},...,d_{j_1j_2\cdots j_i},...,d_{j_1j_2\cdots j_n}\}$。

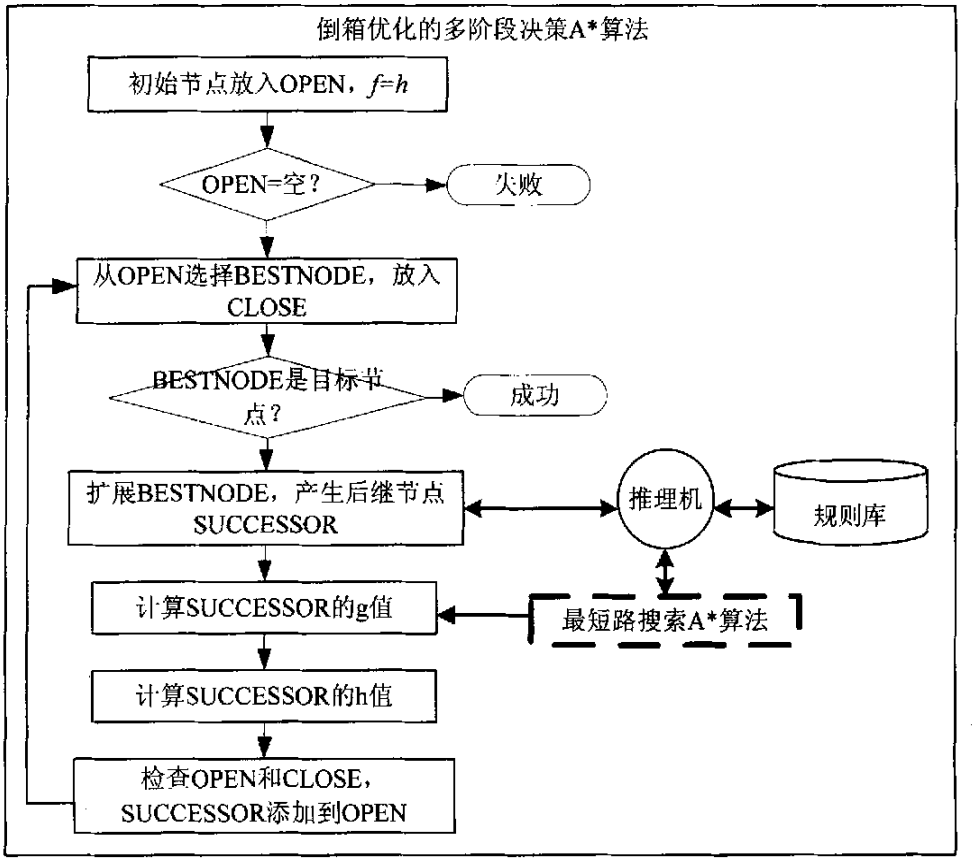

图 4.3 两层 A*算法结构图

Fig.4.3 Structure of Two-layer A* Algorithm

设节点 N 的 g 值为 g_n，表示前 i 个待倒箱从原位置移动到目标位置的实际作业成本总和。由以上分析可知，在倒箱优化多阶段决策中，每个阶段分两步，第一步为把待倒箱从原位置移到目标位置，第二步是设备放下待倒箱后，设备空车从目标位置移到下一个待倒箱的所在位置。因此，g_n 应该由两部分组成：移动箱的成本 g_{n1} 和空设备返回成本 g_{n2}。当前节点的父节点的 g 值为 g_p，则当前节点的 g_n 值：

$$g_n = g_p + g_{n1} + g_{n2} \tag{5.4}$$

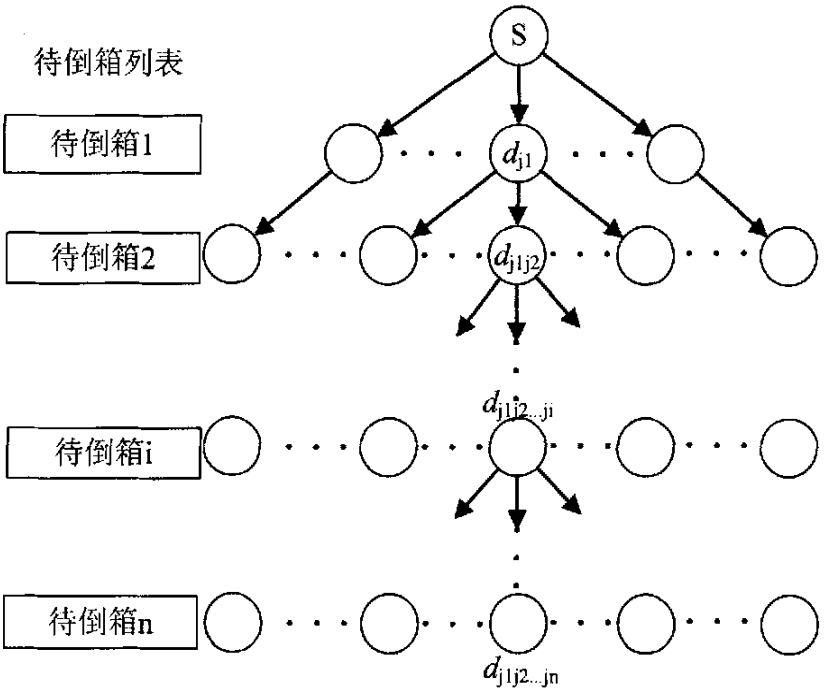

图 4.4 倒箱优化模型的树形结构

Fig.4.4 Tree Structure of Container Rehandling Optimization Model

其中，g_{n1} 和 g_{n2} 通过成本最短路搜索 A*算法求得，两者不同之处在于计算移动箱的成本 g_{n1} 时初始位置为该箱的原所在位置 s_i，目标位置为要放箱的位置 $d_{j_1j_2\cdots j_i}$；g_{n2} 计算空设备返回成本时初始位置为待倒箱的放箱的位置 $d_{j_1j_2\cdots j_i}$，目标位置为下一个要倒箱的所在位置 s_{i+1}。

当节点 N 被扩展时，首先从规则库中提取相应的作业规则，通过推理机制，把数据库中输入的数据与规则相匹配，判断出可以作业的位置，按前述的数据结构组织成 N 的后继节点集合中的一个元素，后继节点的集合记为 $SN=\{N_1', N_2', ..., N_n'\}$，SN 中的每一个元素的层数为 $i+1$，表示的位置为在第 i 个箱放到节点 N 所对应的位置时，当前街的状态下，要倒第 $i+1$ 个箱时的可放箱的一个位置，其父节点指针指向 N。

记 h 为树型结构中当前节点到目标节点的估计成本，设当前节点对应的位置坐标 (b_0,r_0,t_0)，目标节点对应的位置坐标(b_1,r_1,t_1)，则 h 表示为式 4.5 所示。

$$h=\begin{cases}C_1|b_0-b_1|+C_2|r_0-r_1|+C_3|t_0-t_1|, t_0\le t_1\\ C_1|b_0-b_1|+C_2|r_0-r_1|+C_4|t_0-t_1|, t_0>t_1\end{cases}\qquad C_i=C_{iE}+C_{iF}\quad i=1,2,3,4 \qquad (4.5)$$

问题的状态空间的大小可以由倒箱总数目 n 和第一个待倒箱初始可放空位置的总数 m 估算出来为 m^n，因此随着 n 和 m 的增大，状态空间会成指数级增长，所以算法中设计作业成本的估计值 f 为式 4.6 所示。通过权值 w 来调节 f 的大小，目的是为了保证算法的搜索运行效率，同时要保证算法的可纳。根据经验 w 值应与当前节点所在的树树型结构中的层数成反比，而与 n 和 m 成正比，从而可以使 h 更接近 h^*。

$$f = g_n + w * h \tag{4.6}$$

当从 OPEN 表中选出的 BESTNODE 在树型结构中的层数如果是第 n 层，则此节点为目标节点，表示的意义是第 n 个待倒箱的倒箱过程结束，也是全部倒箱过程结束，且总作业成本 f 值最小。

（2）最短路搜索模型的 A*算法设计

最短路搜索模型的 A*算法嵌套在倒箱优化 A*算法中，通过前面模型描述部分的堆场图中搜索最短路，计算移动箱的成本和空设备返回成本。

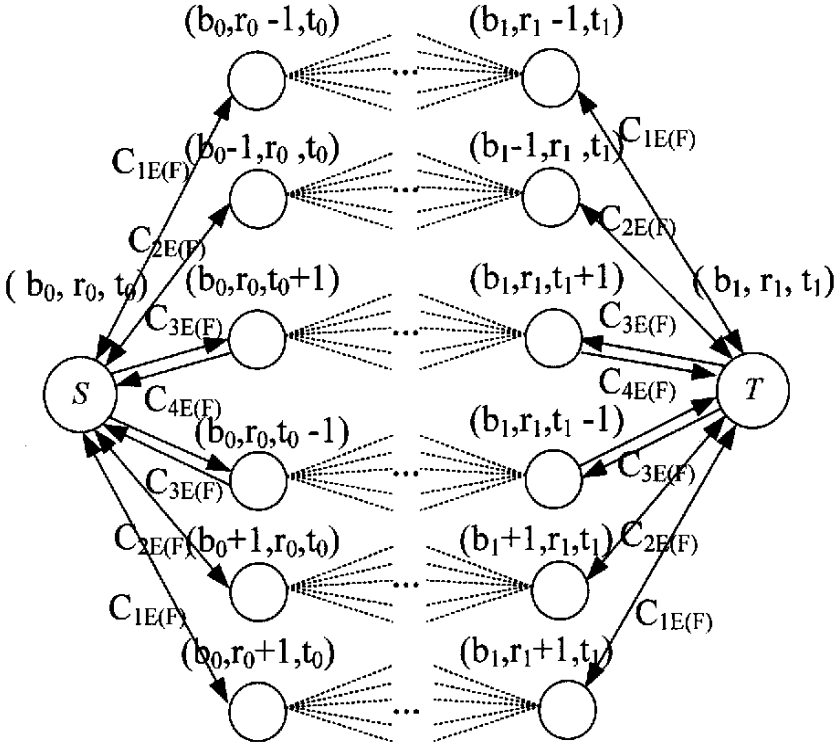

图 4.5 作业最小成本路径搜索图

Fig.4.5 Search Graph of Operation Route

图 4.5 为作业路径搜索图，表示倒箱作业中从初始位置 S(b_0,r_0,t_0)到目标位置 T(b_1,r_1,t_1)的最短路搜索图。图中的每个节点表示街中的一个位置，根据坐标可以唯一标识一个节点，节点保存四个信息：①位置坐标；②位置是否可以被设备作业标志；③位置是否为空；④指向从初始位置搜索到当前位置的路径上，前一个位置节点的指针。在实际堆场堆放中，每个位置最多会有上、下、左、右、前、后 6 个与之相邻的位置节点。节点之间的设备移动成本权值在平移时相同，上下移动时不同。根据作业要求，如果搜索到某位置，由规则推理出设备不可作业，或已经放箱，则通向此位置的成本为∞，在 A*算法中此位置节点将不被当作子节点来扩展。图 4.5 中设节点 N (b_N,r_N,t_N)的 g 值为 g_n 表示从初始位置移动到当前节点位置的成本，设当前节点父节点 P(b_P,r_P,t_P)的 g 值为 g_p，从父节点移动到当前节点的成本为 g_{n1}，则当前节点 g_n 值如式 4.7 所示。

$$g_n = g_p + g_{n1} \tag{4.7}$$

因为节点 P 与 N 在该图中为相邻节点，因此坐标差为单位位移量所以有式 4.8：

$$g_{n1} = \begin{cases} C_{1E(F)}, b_N \neq b_P \\ C_{2E(F)}, r_N \neq r_P \\ C_{3E(F)}, t_N \leq t_P \\ C_{4E(F)}, t_N > t_P \end{cases} \tag{4.8}$$

在此最短路搜索模型的 A*算法中，设 f= g_n+h ，其中，h 表示当前节点位置到终点位置的估计成本，按公式(4.5)计算。

4.2.2 遗传算法设计

（1）初始数据区

①目标待提箱对应一组待倒箱，记作向量$\mathbf{CV} = [c_1, c_2, ..., c_n]$，向量中的每个分量表示一个待倒箱，倒箱的顺序与下标顺序相同，n 为待倒箱的总个数；

②**CV** 中的每个待倒箱 c_i 都对应着一组倒箱作业时的可放箱的目标位置，记作向量$\mathbf{TPV_i} = [tp_{i1}, tp_{i2}, ..., tp_{im}]$，向量中的每个分量表示一个可放箱目标位置，其中 m 为第一个待倒箱 c_1 的可放箱目标位置的个数，c_1 后面的待倒箱 c_i 可放箱目标位置的个数最多会有 m 个，不足 m 个时出缓冲区位置填充；

③一个待倒箱 c_i 从其对应的可倒位置 $\mathbf{TPV_i}$ 中选出一个位置 tp_{ij} 来放置 c_i，所有 n 个待倒箱选出的 n 个目标位置组成一个最终的倒箱方案，记作向量$\mathbf{FS} = [fp_1, fp_2, ..., fp_n]$，其中 $fp_i = tp_{ij}$；

（2）遗传算法编码和解码设计

染色体采用自然数编码，染色体中设有 n 位基因，每一位基因的取值为 1~m 之间的自然数。每个染色体为一个个体，表示一个最终的倒箱方案；染色体中基因的位置号与 CV 中的待倒箱顺序号相对应，即第 i 位基因与待倒箱 c_i 相对应；基因的值与 $\mathbf{TPV_i}$ 中的位置 tp_{ij} 的下标 j 相对应；因此如果第 i 位基因的值为 j，则表示在一个倒箱方案中待倒箱 c_i 选择的放箱目标位置为 tp_{ij}。染色体编码方式如图 4.6 所示。

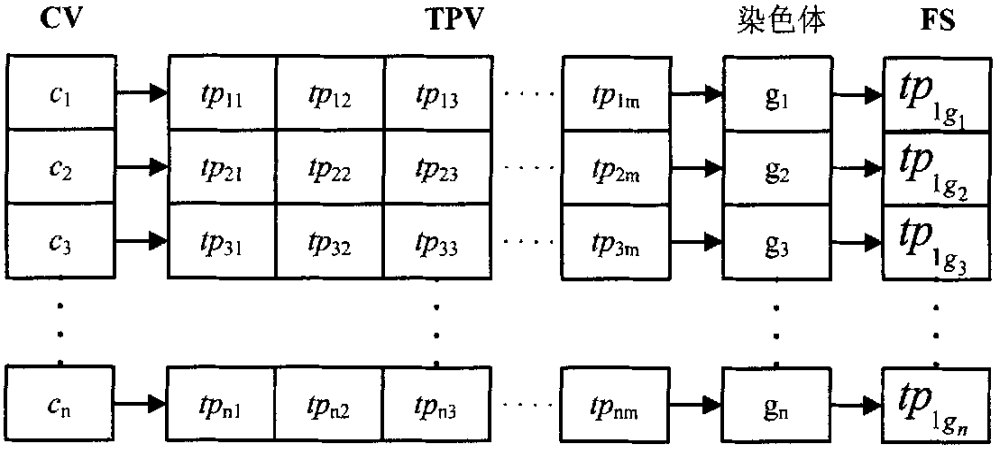

图 4.6 遗传算法编码结构图

Fig.4.6 Coding Structure of GA

（3）初始群体的产生

把堆场作业要求作为约束规则，算法每次运行时，要根据这些约束规则，确定初始数据区的参数，即需要确定倒箱的个数 n，以及第一个待倒箱的可放置目标位置个数 m，然后根据前述编码规则，随机产生可行解作为一个体，如若 n=3，m=8，则随机产生的个体的编码可能为[5,3,5]，意义为从 $\mathbf{TPV_1}$ 中选择 tp_{15}，从 $\mathbf{TPV_2}$ 中选择 tp_{23}，从 $\mathbf{TPV_3}$ 中选择 tp_{35}。群体的数目通过运行算法时的参数设定来确定。

（4）遗传算法适应度函数设计

遗传算法的适应度函数应该与提箱问题模型中的目标函数公式 4.2 相对应。由于目标函数包括倒箱成本和提箱成本，因此在计算倒箱成本时，需要多次调用内层的 A*算法来求作业最小成本和返回最小成本。当染色体中的基因位数为 n 时，就要调用 $2n$ 次 A*算法。提箱问题是一个求最小值问题，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如公式 4.9 所示：

$$Fit(f(x)) = c_{\max} - f(x), f(x) < c_{\max} \tag{4.9}$$

式中 C_{max} 为 $f(x)$的最大值估计，其值通过运行算法时的参数设定来确定。

（5）遗传操作设计

遗传操作包括选择、交叉、变异，相应的实现方法设计如下：

①选择操作。选择操作采用选择方法中最基本的方法——轮盘赌选择法（roulette wheel selection）。在该方法使得种群中的每个个体都有机会根据其与适应度相对的概率被选择，某个体 i 被选择的相对概率 p_i 由公式 4.10 计算，其中，f_i 为个体 i 的适应度值。

$$p_i = f_i / \sum f_i \tag{4.10}$$

$$r_i = p_1 + p_2 + \cdots + p_i \tag{4.11}$$

可以根据计算出来的概率$\{p_i, I=1,2,\ldots,n\}$把圆盘分成 n 份，其中第 i 扇形的中心角为 $2\pi p_i$，针对第 i 个个体计算从 1 到第 i 个体的累计相对概率为 r_i 由公式 4.11 求得，旋转圆盘 n 次，每次产生一个[0，1]的随机数 r，满足 $r_{i-1} < r \le r_i$ 时，则个体 i 被选中加入新的种群。

根据概率的计算公式可以看出，适应度越大的个体，越容易被选择，在下一步参加交叉的可能性就高。

②交叉操作。交叉操作采用两点交叉法。在进行交叉操作时，随机的设置染色体上的 2 个位置，然后将这 2 个位置进行交叉，生成新的 2 个个体。设两个父串分别为 $x=[x_1,x_2,\ldots,x_n]$ 和 $y=[y_1,y_2,\ldots,y_n]$，由随机数发生器，产生两个[1，n]的随机数 i、j，选择第 i、j 位作为 2 个交叉位（其中 $1<i<j<n$），生成的新的后代为 $x' = \left[x_1, x_2, \ldots, x_i, y_{i+1}, y_{i+2}, \ldots, y_j, x_{j+1}, \ldots, x_n\right]$ 和 $y' = \left[y_1, y_2, \ldots, y_i, x_{i+1}, x_{i+2}, \ldots, x_j, y_{j+1}, \ldots, y_n\right]$。

③变异操作。变异操作采用的是均匀变异方法。在父代向量中随机选择第 $k(k \le n)$个分量，然后在其定义的区间$[a_k,b_k]$中均匀随机地取一个随机数 z_k^1 代替 z_k 得到 Z，其数学描述为：父代为 $S = (v_1, v_2, \ldots, v_n)$，变异后产生的子代为 $Z = (z_1, z_2, \ldots, z_n)$，变异关系为公式 4.12，其中 $z_i, z_i^1 \in [a_i, b_i], i = 1,2,\ldots,n$。在本问题中，变异的值范围$[a_i,b_i]$为[1,$m$]。

$$v_i = \begin{cases} z_i, i \ne k \\ z_k^1, i = k \end{cases} \tag{4.12}$$

（6）遗传算法流程

遗传算法流程主要步骤包括初始群体产生、选择、交叉、变异，流程图如图 4.7 所示：

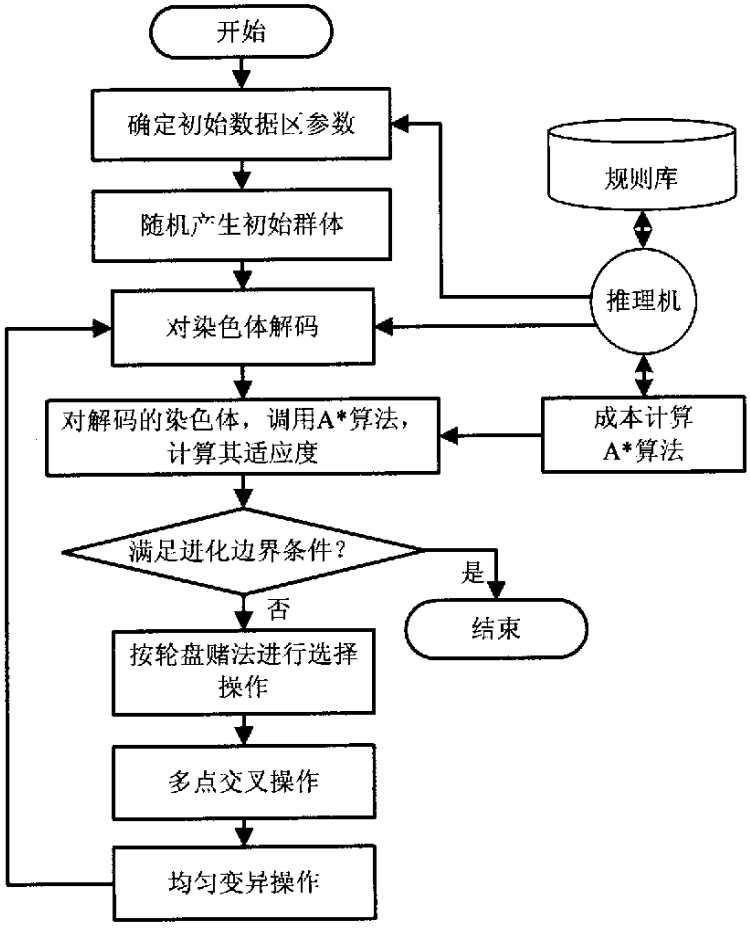

图 4.7 遗传算法流程图

Fig.4.7 Flow Chart of GA

5 系统的实现、验证及比较

5.1 系统的实现

5.1.1 系统实现的方案比较

（1）动态连接库接口方案

外部系统用初始信息初始化算法所必需的类，传递给算法类的构造函数；然后调用算法的类的执行程序运行算法，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。

此方案运行速度快，I/O 数据交换可靠，但与主系统联系紧密，需要联合调试。

（2）ATL COM 方案

ATL——活动模板库（The Active Template Library），其设计旨在让人们用 C++方便灵活地开发 COM 对象。

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。

COM 即组件对象模型，是 Component Object Model 取前三个字母的缩写。COM 是一种跨应用和语言共享二进制代码的方法，它提倡源代码重用。COM 通过定义二进制标准解决了 C 语言编写 DLLs 和 MFC 扩展 DLLs 的在使用中受限的问题，即 COM 明确指出二进制模块（DLLs 和 EXEs）必须被编译成与指定的结构匹配。这个标准也确切规定了在内存中如何组织 COM 对象，COM 定义的二进制标准与编写模块所用的语言是无关的，因为结果二进制代码为所有语言可用。而且从理论上讲 COM 不是 Win32 特有的，它可以被移植到 Unix 或其它操作系统。

为了创建 COM 对象并从这个对象获得接口，必须调用 COM 库的 API 函数，CoCreateInstance()。当调用 CoCreateInstance()时，它负责在注册表中查找 COM 服务器的位置，将服务器加载到内存，并创建所请求的组件对象类（coclass）实例。调用 Release()方法通知 COM 对象使用完成。

5.1.2 系统的实现

本系统应用面向对象的程序设计方法，在 VC 环境下进行编译开发，在 Windows2000 平台下进行测试。

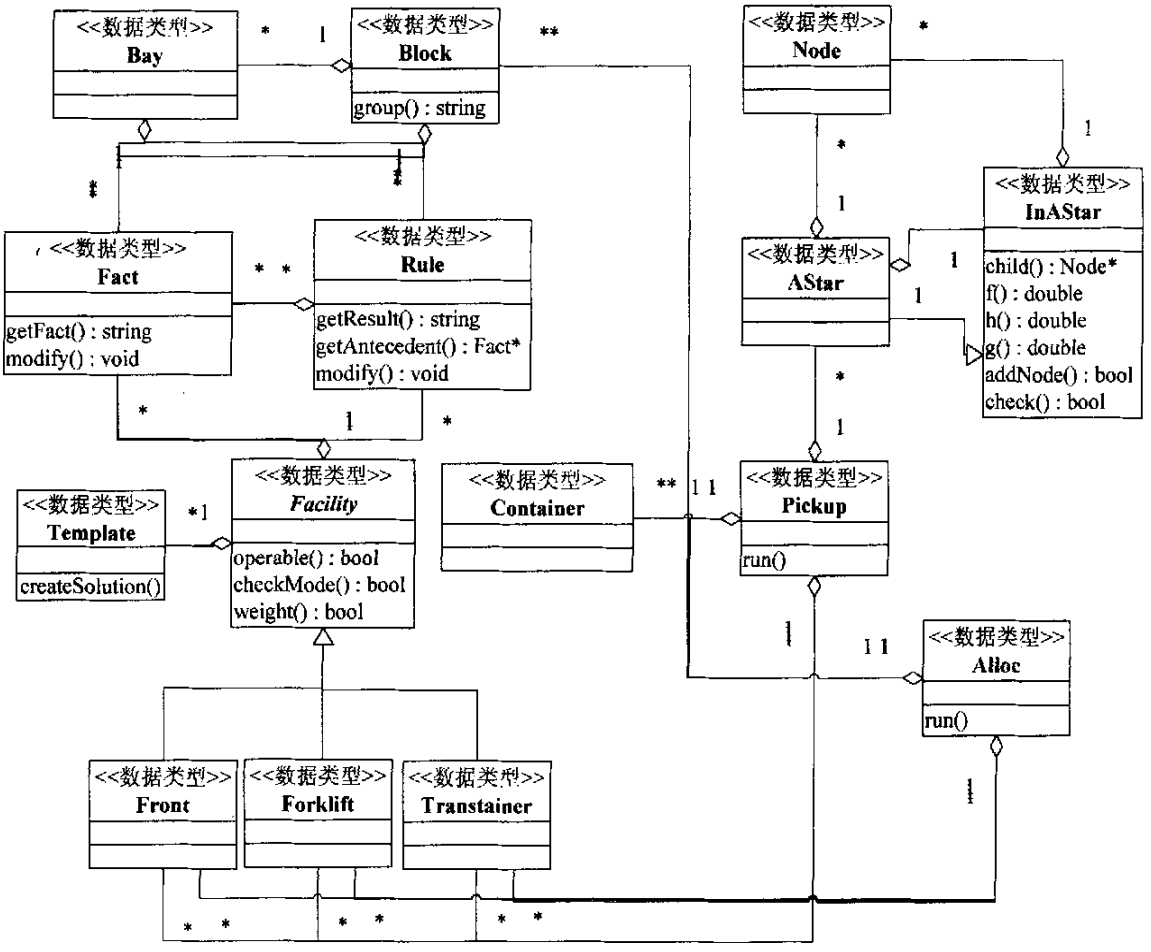

图 5.1 系统类关系图

Fig.5.1 System Classes

系统按照图 5.1 所示的为基于 UML 的 C++类关系设计。

（1）Pickup 类主要实现提箱算法，提供 run()接口供外部系统调用。

（2）Alloc 类主要实现落箱算法，提供 run()接口供外部系统调用。

（3）InAStar 类主要提供提箱算法中的内层 A*搜索功能。AStar 类继承 InAStar 类，主要提供提箱算法中的外层 A*搜索功能。chile()提供节点扩展功能。addNode()为向 open 表添加子节点方法。check()检查扩展节点是否在 open 表或 closed 表中。f()、h()、g()分别为计算节点 f、h、g 值的函数。Node 保存 A*算法中节点信息。

（4）Container 保存堆场中集装箱基本属性。

（5）Bay 保存一个贝的基本属性。Block 保存一条街的基本属性。因此一个 Block 对象包含多个 Bay 对象。group()提供落箱分组功能。

（6）Facility 设备抽象类。提供 operable()、checkMode()、weight()三个纯虚方法。Front、Forklift、Transtainer 都继承 Facility 类，并根据各自的作业规则实现 Facility 类的虚拟方法。

（7）Fact 和 Rule 为事实类和规则类，一个事实类或规则类的对象对应一个事实或一条规则。Facility 类、Bay 类和 Block 类中都可以包含多个事实或规则。

（8）Template 为设备作业顺序模板类，一个 Template 对象对应一个模板，一个 Facility 类可有多个 Template 对象。

5.2 自动分配箱位算法的验证

本文所采用的算例为 30 个随机到达的划入同一区域、重量等级服从均匀分布 U(1,3)的集装箱(箱号/重量等级)，按到达顺序依次为：T01/1，T02/2，T03/3，T04/1，T05/2，T06/3，T07/3，T08/2，T09/1，T10/3，T11/3，T12/2，T13/3，T14/1，T15/3，T16/2，T17/3，T18/2，T19/1，T20/1，T21/3，T22/1，T23/2，T24/3，T25/1，T26/2，T27/2，T28/1，T29/1，T30/2。

设定区域为 A01005，A02007，即 A01 街 5 贝、A02 街 7 贝，且该区域中最初没有放任何集装箱，区域中贝的定义如下：

(1)A01005,A,1,1,6,4,1：表示从小行到大行作业，6 行 4 层，落箱原则为重压轻。

(2)A02007,A,6,2,6,4,1：表示从大行到小行作业，6 行 4 层，落箱原则为重压轻。

作业设备为场桥，设备初始位置分别为 A01001，作业顺序模板如图 3.8 模板 2 所示。采用本方法的得到的箱位分配堆垛如图 5.2 所示。

A01005贝

	1	2	3	4	5	6
4	T10/3	T21/3				
3	T03/3	T13/3	T25/1			
2	T02/2	T12/2	T20/1	T26/2		
1	T01/1	T09/1	T19/1	T22/1		

A02007贝

	1	2	3	4	5	6
4				T27/2	T17/3	T07/3
3				T23/2	T15/3	T06/3
2		T29/1	T24/3	T16/2	T11/3	T05/2
1	T30/2	T28/1	T18/2	T14/1	T08/2	T04/1

图 5.2 堆放优化结果

Fig.5.2 Storage Optimization Result

上述条件下，使用不同设备所产生的箱位分配结果略有差异，但由结果可以看出，通过该种方法所产生的堆场堆垛，符合重量等级要求，方便后续装船作业；优化的同时考虑了设备移动距离和作业成本。

5.3 提箱算法的验证与比较

表 5.1 为各类设备作业的成本权值，是一种根据实际作业需要设定的相对值。

表 5.2 为堆场中一个拥有 12 贝，4 行，4 层的街的集装箱位置的初始状态表。其中，Txx 表示箱号（xx 为数字），没有标箱号的位置为空位置。表中粗方框表示的箱 T30(7,1,2)为待提箱。

表 5.1 设备作业成本的权值

Tab.5.1 Weight of Equipment Operation Cost

设备		成本设定			
类型	状态	C_1	C_2	C_3	C_4
正面吊	重车	15	13	10	6
	空车	10	8	0	0
叉车	重车	16	14	8	5
	空车	10	8	0	0
场桥	重车	20	11	5	3
	空车	15	6	0	0

表 5.3 给出了针对待提箱 T30(7,1,2)，倒箱范围为以 T30 所在的 7 贝为基准的左右 4 个贝之内，使用三种不同的设备分别作业时本算法的优化结果。表 5.3 中，搜索节点数为 A*算法中搜索过的在 OPEN 表和 CLOSE 表中的节点总数，估计节点总数为倒箱优化多阶段决策 A*算法的估计总节点数，从两种节点数的比较可以看出，本算法的搜索效率较高。

为进一步验证本算法的正确性和有效性，采用外层为遗传算法 GA，内层为 A*算法的算法组合（GA+A*算法组合），将它和本算法的 A*+A*算法组合进行比较验证。外层遗传算法的主要参数如下：群体规模 20；交叉概率 0.6；变异概率 0.05，运行 100 代。表 5.3 中的 GA 优化成本行为 GA+A*组合的优化成本，实际最优成本为经过验证的本问题的最优值。实际最优成本下的最优倒箱方案和 A*+A*及 GA+A*组合的倒箱方案结果也都相同（见表 5.3 的待倒箱及目标位置行）。说明本算法的优化结果具有最优性。

表 5.2 堆场箱位的初始状态表

Tab.5.2 Initial State Table of Stack

行 \ 贝 / 层		1	3	5	7	9	12
1	4				T32		T45
	3	T03			T31		T44
	2	T02	T12	T21	**T30**	T38	T43
	1	T01	T11	T20	T29	T37	T42
2	4						
	3	T06					T48
	2	T05	T14	T23	T34		T47
	1	T04	T13	T22	T33		T46
3	4						
	3						
	2	T08		T25		T40	T50
	1	T07	T15	T24	T35	T39	T49
4	4		T19				
	3		T18	T28			
	2	T10	T17	T27			T52
	1	T09	T16	T26	T36	T41	T51

表 5.3 T30 提箱作业优化结果表

Tab.5.3 Pick-up Operation Optimization Result for Container T30

设备类型	正面吊	叉车	场桥
搜索节点数	24	24	14
总节点数	125	128	196
待倒箱及目标位置	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)
A*优化成本	374	814	228
GA 优化成本	386	831	246
实际最优成本	374	814	228

三者最优成本值的差异主要是由各自成本计算方法上的差异造成的，这是相对值，只对算法寻优提供依据。

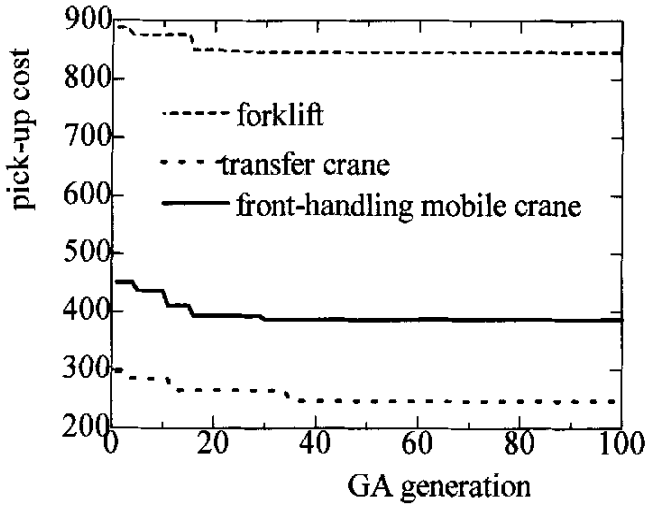

图 5.3 GA+A*算法组合的进化过程

Fig.5.3 Evaluation Process of GA Algorithm Combined with the A* Algorithm

图 5.3 显示了 GA+A*组合算法在上述堆场条件和设备成本权值下，三种设备分别提取 T30 箱时，遗传算法进化 100 代的提箱作业成本平均值的演变过程。

图 5.4 显示了对于采用正面吊作业情况下，待倒箱数 n 分别为 2 和 4 时，随着问题搜索空间增大，分别采用 GA+A*组合和 A*+A*组合两种方案时，在得到相同最优化结果的条件下，两种算法在运行时间上的变化情况。

本算例情况下，m 最大值一般不超过 30，因此取倒箱目标位置数 m=3,6, …,30 时，群体规模 *popsize*=20，遗传代数 *gen*=100。两种方案运行时间比较如下：

①n=2 时，GA+A*组合运行到 m=27 以后，运算时间超过 2s，不满足作业要求；A*+A*组合直到 m=30 一直可以满足作业要求。

②n=4 时，GA+A*组合已无法满足作业要求；A*+A*组合在 $m\leq 28$ 时，运算时间<2s，可以满足作业要求。

另外从图 5.4 中还可以看出，随着 n 变大两种算法的运行时间的变化曲线的斜率也相应变大，且变化趋势基本相同。

由于算法搜索空间为 m^n，且有：

①A*+A*算法组合中外层 A*调用内层 A*的次数在最坏情况下为 $2\times m^n$ 次，是关于待倒箱数 n 的指数函数；但从表 5.3 中的结果可以看出：A*+A*算法组合的搜索节点数远比估计节点数小，其搜索效率较高，因而调用次数远小于 $2\times m^n$ 次。

②GA+A*算法组合中外层 GA 调用内层 A*的次数为 $2n\times popsize\times gen$，是关于待倒箱数 n 的线性函数。而对于遗传算法来说，要得到最优解，则在问题空间增大时，*popsize* 和 *gen* 也必然随之增大，因此调用次数也必然增大。

由此可见，GA 和 A*算法相比，在相同的 n 值和 m 值的前提下，调用内层 A*算法的次数的计算方法是不同的，而该调用过程是优化算法运行过程中花费时间较多的一个部分，两者运行时间的差别由此产生。但 A*算法的运行结果较好。

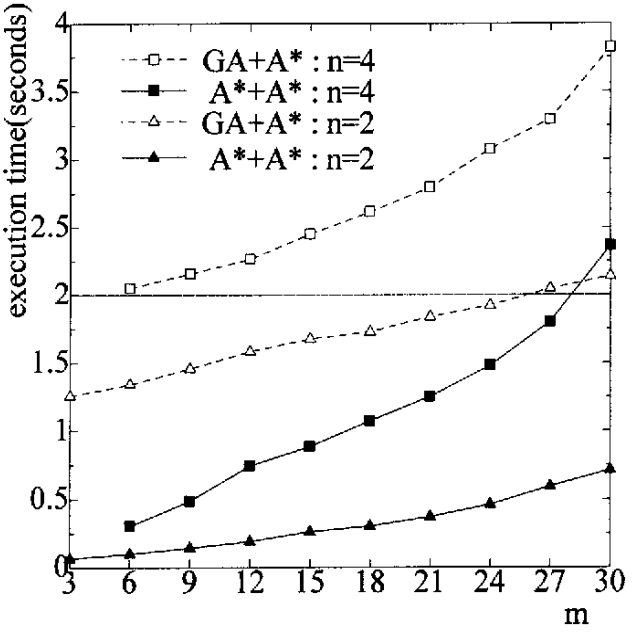

图 5.4 问题搜索空间增大时两种算法运行时间的比较

Fig.5.4 Comparison of Two Algorithm Running Time with Increasing of Problem Search Space

结　论

本文的主要工作

本文研究了集装箱堆场大门一侧集装箱作业的优化问题，主要结论如下：

（1）根据堆场作业的主要领域知识建立了事实数据库和作业规则库，用于堆场大门作业优化算法求解过程。

（2）提出了基于规则的自动分配箱位方法模型，阐述自动分配箱位方法、规则库、推理机制的相互作用关系。

（3）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，该优化模型以作业总成本最小为目标，以能满足多种设备实际作业要求的作业规则函数为约束条件。

（4）通过对问题状态空间的分析，提出并实现了基于两层嵌套结构的 A*算法的集装箱堆场提箱作业启发式优化算法，其中外层 A*算法用于求解倒箱优化子模型，内层 A*算法用于求解最短路搜索子模型。

（5）通过实际算例对提箱和落箱算法的正确性、效率性进行了验证，表明本算法的优化结果为最优方案，在运行效率上也较优，可满足实际系统作业的要求。

本文在理论上，对于很难用传统的运筹学手段求解的港口集装箱作业多阶段决策的优化问题，提出了基于人工智能的解决方法；在实际应用领域上，以实际项目为背景，以实际需求为对象进行研究，具有针对性和实用性。

进一步的研究工作

下一步研究将在本研究基础上展开以下工作：

（1）把作业范围扩展到整个堆场区域，这就需要考虑作业设备在堆场上的行走路径优化问题。

（2）应用仿真工具，对整个堆场的作业进行仿真建模，同时把优化算法嵌入到仿真模型中，为实际问题提供决策依据。

（3）本文建立的规则推理过程以精确推理为主，下一步将提供模糊推理的功能。

参 考 文 献

[1]日本海上集装箱协会,集装箱运输业务手册编委会编.刘鼎铭,王义源译.集装箱运输业务手册（上册）.北京:人民交通出版社,1992.

[2]真虹.集装箱运输学.大连:大连海事大学出版社,2002.

[3]King D H, Radomske B A and Manocha G S.Recent Advances in Simulation Models for Bulk Terminal Design. Bulk Solids Handling, 1993, 13(1) :23-27.

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

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

[6]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.

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

[8]Ebru K, Bish. A multiple-crane-constrained scheduling problem in a container terminal. European Journal of Operational Research, 2003,144(1):83-107.

[9]Van der Meer J R. Operational Control of internal Transport ERIM Ph.D. Series Research in Management, 2000.9,series 1.

[10]Zhang C. Dynamic crane deployment in container storage yards: [PhD thesis].Hong Kong: Department of Industrial Engineering and Engineering Management, Hong Kong University of Science and Technology, 2000.

[11]Kim K Y, Kap H K. A routing algorithm for a single straddle carrier to load export containers onto a containership. International Journal of Production Economics 59,1999:425-433.

[12]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.

[13]别社安, 孙锡衡. 动态随机循环网络模拟方法及堆石坝体的施工模拟.水利水电技术，1998,(12): 14-16.

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

[15]Kim K H, Kim D Y. Group storage methods at container port terminals. MH-2, The Materials Handling Engineering Division, 75th Anniversary Commemorative Volume, 1994: 15-20.

[16]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.

[17]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.

[18]Taleb-Ibrahimi M, De Castilho B, Daganzo C F. Storage space vs handling work in container terminals. Transportation Research, 1993.27B(1):13-32.

[19]De Castilho B, Daganzo C F. Handling strategies for import containers at marine terminals. Transportation Research, 1993. 27B(2): 151-166.

[20]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.

[21]Kozan E, Preston P. Genetic algorithm to schedule container transfers at multimodal terminals. International Transactions in Operational Research 6, 1999: 311-329.

[22]Avriel M, Penn M, Shpire N, et al. Stowage planning for container ships to reduce the number of shifts. Annals of Operations Research,1998: 55-71.

[23]Shields J J. Container stowage: a computer-aided preplanning system. Marine Technology, 1984, 21(4):370-383.

[24]Kim K Y, Kim K H. A routing algorithm for a single transfer crane to load export containers onto a containership. Computers and Industrial Engineering, 1997, 33 (3) :673-676.

[25]Kim K H, Park Y M, Ryu K R. Deriving decision rules to locate export containers in container yards. European journal of Operational Research,2000:89-101.

[26]Kim K H, Bae J W. Re-marshaling export containers in port container terminals. Computers and Industrial Engineering, Volume 35, Issue 3-4, December, 1998, Pages 655-658.

[27]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.

[28]Preston P, Kozan E. An approach to determine storage locations of containers at seaport terminals. Computers and Operations Research, 2001,28(10):983-995.

[29]Kim K H. Evaluation of the number of rehandles in container yards. Computers and Industrial Engineering, 1997, 32(4): 701-711.

[30]郝聚民, 纪卓尚, 林焰. 混合顺序作业堆场 BAY 优化模型研究. 大连理工大学学报, 2000, 40(1): 102-105.

[31]蔡自兴, 徐光祐. 人工智能及其应用. 北京:清华大学出版社, 2000.

[32]李陶深. 人工智能. 重庆：重庆大学出版社，2002.

[33]刘士新, 王梦光, 唐加福. 资源受限工程调度问题的优化方法综述. 控制与决策, 2001, (16):647-651.

[34]刘云忠, 宣慧玉. 车辆路径问题的模型及算法研究综述. 管理工程学报, 2005, 19(1):124-130.

[35]王小平, 曹立明. 遗传算法-理论、应用与软件实现. 西安:西安交通大学出版社, 2002.

[36]付永锋. 一种改进的遗传算法. 长春师范学院学报, 2003,22(2): 9-12.

[37]Srinvivas M, Patnaik L M. Adaptive Probabilities of Crossover and Mutations in GAs. IEEE trans. On SMC, 1994, 24(4):656-667.

[38]黄聪明, 陈湘秀. 小生境遗传算法的改进. 北京理工大学学报, 2004,24(8): 675-678.

[39]高济. 基于知识的软件智能化技术. 杭州:浙江大学出版社,2000.

[40]Hayes-Roth F. Rule-based Systems. Communications of the ACM. 1985,28(9):921-932

[41]Eshelman L J. The CHC Adaptive Search Algorithm: How to Have Safe Search when Engaging in Nontraditional Genetic Recombination. In: Foundations of Genetic Algorithms, Morgan Kaufmann Publishers, 1991, 265-283.

附录 A 落箱算法部分程序代码

```
/*
    COMDLL 的调用过程
*/
    ALC= ::CoCreateInstance(
                                CLSID_AutoAllocDll,
                                NULL,
                                CLSCTX_INPROC_SERVER,          //以进程内组件 DLL 方式加载
                                IID_IUnknown,                  //想取得 IUnknown 接口指针
                                (LPVOID *) &pUnk
                                                    );
    if( FAILED(ALC) ) //throw( _T("没注册") );
        cout<<"没注册"<<endl;
    ALC = pUnk->QueryInterface(// 从 IUnknown 得到其它接口指针
                                IID_IAutoAllocDll,             //想取得 IFun 接口指针
                                (LPVOID *)&pAutoAlloc);
    if( FAILED(ALC) ) //throw( _T("没有接口？") );
    ALC =pAutoAlloc->addData_dll();
    ALC =pAutoAlloc->run_dll();                                //算法运行接口
/*
    落箱算法调用接口
*/
void SAutoAlloc::run()
{   //初始化，指定位置的排在前面；没有指定位置，按重量等级先后排列
    initial();
    for(int i=0;i<conAmt;i++)
    {   //依次对各个箱处理
        if(container[i].getPosition().getBlock()!="UNKNOWN_BLOCK")
        {   //如果箱已指定位置，运行处理指定位置的函数
            preAssigned(container[i]);
            addRsltPos(i,bestPos);
            setError(i);                //设置结果状态代码
        }
        else
        {   //箱没有指定位置，按照自动分配箱位算法处理
            preProcess();               //预处理,排除不可作业位置
            assignModeNum();            //匹配设备作业模式，给每个位置赋模式序号
            sortAccMode();              //分贝按模式序号从小到大排序
            select(container[i]);       //根据重量等级选择落箱备选位置
            chooseBest();               //选择最优位置
            addRsltPos(i,bestPos);      //把计算得到位置放入结果中
            setError(i);                //设置结果状态代码
        }
        update(container[i]);       //刷新所有数据结构，准备计算下一个箱
    }
}
```

附录 B 提箱算法部分程序代码

```
/*
    提箱算法调用接口
*/
vector<Position> PickUp::run()
{
    AStar_rehandleCon astar(block,*iter,conList,range,rehandWay,facility);  //初始化 A*算法对象
    int astarRe = astar.start();                                            //调用外层 A*算法运行
    if(astarRe!=0)
    {
        return astar.getTagPos();                                           //返回箱目标位置
    }
}
/*
    外层 A*算法
*/
int AStar_rehandleCon::start()
{
    Node *BestNode=NULL,*lastBestNode=NULL;
                        //BestNode 是 f 值最小的节点指针，lastBestNode 是记录上次的 bestNode
    bool judge=false;           //用来判断当前节点是否是目标节点 final node，destination
    Position* dPos=dContainer->getPosition();     //层行贝，以结构体的形式保存待提箱的位置信息
    changeP(dPos,dx,dy,dz);                 //将箱的 Postion 信息进行坐标变换
    int sLayer=1;                           //当前搜索树的层数从 1 开始
    count=0;                                //记录循环次数
    GenerateChild(BestNode,sLayer);         //根据第一个待搞的箱的信息，生成一后继节点
    listType::iterator headIterator;
    while (!Open.empty())
    {    //如果 open 表已经为空，则退出循环
        BestNode=&Open.front();             //从 Open 表中取得当前 f 值最小的节点地址-BestNode
        for (headIterator=Open.begin(); headIterator!=Open.end();++headIterator)
        {
            if(headIterator==Open.begin())
            {
                continue;
            }
            if(headIterator->data.f==BestNode->data.f)
            {
                if(headIterator->data.g<BestNode->data.g)       //选取 h 值最小的那个节点
                {
                    BestNode=&(*headIterator);
                }
                else
                {
                    continue;
                }
            }
```

```
            else
            {
                break;
            }
        }
        while(BestNode!=&(*headIterator))
        {
            headIterator--;
        }
        Closed.insert(Closed.end(),*BestNode);    //向 Closed 表中加入 BestNode
    BestNode=&Closed.back();
        Open.pop_front();                         //从 Open 表中删除刚扩展的节点 BestNode
        sLayer=BestNode->data.layer+1;            //层数增加 1，转到下一层
    if(isDestCont(BestNode,dLayer))              //是目标节点的话，则终止循环
        {
            judge=true;
            break;
        }
        else                                      //不是目标节点的话，继续扩展节点
        {//如果上次的 bestNode 节点正好是这次最优节点的父节点、则只需要更新 BestNode 节点对应
的要捣箱的移动情况
            if(lastBestNode==BestNode->parent)
            {//只需要更新一次，将原位置设为 NULL，新位置设为箱号就可以，1 表示只更新一次
                updateBlock(BestNode,true,0);
            }
            else
            { //在扩展 BestNode 节点前，先更新街场信息，记录 bestnode 从原位置移动到当前位置
            updateBlock(BestNode);
            }
            GenerateChild(BestNode,sLayer);       //生成子节点
        }
        count++;
        lastBestNode=BestNode;
    }
    if(judge)                                     //当找到待提箱时
    {
        trsfCost=Path_best(BestNode);             //求得路径 PATH;从 BestNode 开始，找寻父节点，
        return 1;                                 //1 表示找到了最优方案
    }
    else                                          //当 Open 表已经为空且没有找到待提箱
    {
        return 0;                                 //0 表示没找到
    }
}
```

攻读硕士学位期间发表学术论文情况

发表论文：

1、Peng Gao, Chun Jin, and Xuejie Wang. An Intelligent Simulation Method Based on Artificial Neural Network for Container Yard Operation. Advances in Neural Networks --ISNN2004. International Symposium on Neural Networks DaLian, China, August2004. 论文第一、三章.

2、Peng Gao, Chun Jin, and Xuejie Wang. Knowledge-based Federation Simulation Model for Operation Coordination of Multimodal Transportation System of Ports. The Proceeding of the Sixth International Symposium on Knowledge and Systems Sciences (KSS'2005), August 29-31, 2005, IIASA, Laxenburg, Austria. 论文第一、三章.

3、高鹏. 基于A*算法的集装箱堆场提箱作业优化. 大连理工大学研究生网络学刊.论文第一、三、四章.

参加项目及研究成果：

1、国家重点学科“管理科学与工程”交叉项目——《基于供应链的的物流及作业计划问题的研究》，2003年7月——2004年7月，主要参加人

2、国家重点学科“管理科学与工程”交叉项目——《基于知识的复杂系统仿真方法研究》，2004年7月——2005年6月，主要参加人

3、企业合作项目——集装箱堆场作业优化算法的软件系统开发，2005年5月——2005年9月，主要参加人

致　谢

本文是在导师金淳老师的悉心指导下完成的。金老师对本文的选题、整体构思，以及论文的章节安排给予了耐心的指导，付出了很多心血，提出了大量宝贵的意见，使我受益良多。在这里特别感谢您平日里对我的悉心指导和亲切的关怀。两年多来，金淳老师正直的为人、从容大度的处事风范、平易近人的作风、渊博的学识、严谨治学的态度和孜孜不倦的工作热情，给我留下了极其深刻的印象，不仅使我在专业学术方面受益匪浅，更让我在今后的学习研究、工作和生活中终身受益。在此谨向金淳老师表示由衷的感谢和最崇高的敬意！

其次，要感谢信息管理系所有老师为本文提出的宝贵意见和建议，正是由于他们的辛勤施教，使我学到了许多宝贵的知识，并拥有了不懈学习的动力，在此深表感激。感谢刘昕露、沈剑锋、王雪杰和于越等同门给予我的帮助和支持！在论文的写作期间，我们就有关问题进行了深入的有益的探讨和交流。感谢大连口岸物流科技有限公司的领导和员工给予的大力支持，使论文的资料调查和收集工作得以顺利进行。

同时感谢我的母校、感谢信息管理系为我提供了一个良好的研究和学习的环境，感谢和我朝夕相处的同学们，彼此间融洽的合作和自由的讨论使我受益匪浅；感谢所有关心过我、帮助过我的人们！

最后，深深地感谢我的父母，是他们给予我的爱和支持，使我能够克服各种困难，顺利完成学业。在近两年半的学习生活中，生活里的付出和收获，学习上的刻苦和拼搏，研究中的困难和喜悦，都将变成我心灵深处最美好的回忆，值得永久地珍藏和品味。我会永远记住这一段难忘的时光！在以后的学习、工作、生活中，不断求索，领悟人生的真谛。