# 桥吊迁移的港口连续泊位分配问题在线算法①

仇 明[1], 杨智应[1,2]

[1](上海海事大学 信息工程学院, 上海 201306)
[2](复旦大学 计算机科学技术学院, 上海 200433)

**摘 要**: 集装箱码头资源的高效利用已被研究多年, 而多数是在预知所有船舶作业的相关信息(到港时间、船舶尺寸等)的离线情况下建模与计算. 现实中, 却因一些突发因素(如恶劣天气、设备故障等)使预知信息不可靠, 以至原调度方案不可行, 从而降低港口作业效率及资源浪费. 故在桥吊可迁移的连续泊位分配模式下, 首次结合在线算法思想, 提出泊位与桥吊调度的模型, 并设计相应的在线调度算法. 利用平滑分析方法给出算法的平滑竞争比, 实验证实算法可行性.

**关键词**: 集装箱码头; 泊位分配; 桥吊迁移; 在线算法; 平滑分析

## Online Scheduling Algorithm of Continuous Berth and Quay Crane of Container Terminal With QCs Transfer

QIU Ming[1], YANG Zhi-Ying[1,2]

[1](College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China)
[2](College of computer science and technology, Fudan University, Shanghai 200433, China)

**Abstract**: Container terminal resources efficiency have been studied for many years, however, most researches focused on offline algorithms which are given the whole problem data from the beginning. Due to some unforeseen factors(such as bad weather, equipment failure), the predict information is not reliable, these make the planned difficult to implement, and reduce the efficiency of port operations. So we are based on the idea of online algorithms and propose a optimization model of berth container terminal and bridge crane resource, and give a online scheduling algorithm for solving. We also analyzed the smoothed competitive ratio by the smooth analysis. Through the experiments confirmed the algorithm feasibility.

**Key words**: container terminal; berth allocation; crane transfer; online algorithm; smoothed analysis

## 1 引言

随着国际贸易的步伐加速, 自由贸易区的落沪, 对国际物流提出了更高的新要求. 一切迫使集装箱码头资源的调度能应对实时港口作业变化现状以利于更加合理、高效的运作. 集装箱码头岸边资源主要有: 泊位和桥吊. 对这些重要资源的分配问题可分为: 泊位分配问题(BAP)、桥吊分配问题(QCAP)[1]. 这两大问题的解决方案需要考虑泊位布局和工艺, 它影响着堆场操作和人力资源分配, 在提高港口作业效率方面具有重要作用.

人们很早就开始对集装箱港口资源分配问题进行研究, 并取得了大量的研究成果. 但绝大多数研究是在假设预知所有集装箱船舶装卸作业的相关信息(到港时间、箱量、船舶尺寸、吃水深度等)的离线情况下进行建模和计算, 且传统的集装箱港口资源分配问题研究中, 泊位分配与桥吊分配问题多数是分开考虑的. 首先, 泊位分配问题(BAP)是指对港口岸线的码头空间和船舶装卸货物服务时间的分配, 已被证实是一个NP难问题. 因而研究者主要通过研究和改善近似算法或智能算法来优化目标函数值. Lai 等[2]以先到先服务(FCFS)策略为基础, 针对固定泊位大小的离散型泊位调度问题给出了一种的启发式算法, 并针对“船舶平均等待时间最小、平均停靠时间最小和平均泊位利用率最大”等不同目标函数下的泊位调度方案做了评估.

① 收稿时间:2015-03-10;收到修改稿时间:2015-04-26

蔡芸等[3]建立了最小化总体船舶在港时间的仿真优化模型, 采用遗传算法求解仿真优化模型, 并对该泊位分配方案进行了评价, 该文通过仿真模型获得了满足靠泊约束和岸桥调度策略的可行解, 最终求得了包括船舶的靠泊时间、靠泊位置和为其服务的岸桥数目等要素的优化解. 为提高岸线利用率, 不少专业化码头开始广泛采用柔性靠泊方式, 即指船舶直接按照岸线上表示的物理位置进行停靠. 可以跨越多个泊位的连续型泊位分配. Park 等[4]假设船舶有最佳泊位和最佳离港时间, 当船舶未分配到最佳泊位时, 便会形成额外成本, 显然目标函数是最小化总额外成本, 文中依据现实约束建立相应整数规划模型, 并分两阶段求解. 张佳运等[5]以船舶总在港时间最短为目标函数, 基于 Q 学习方法给出优化算法, 并用集装箱码头的实际数据, 验证了模型及算法的有效性. 张海滨等[6]将集装箱码头泊位调度问题转化为带有约束条件的二维装箱问题, 建立非线性规划的连续泊位调度模型, 并给出了一种求解的启发式算法.

泊位调度的下一阶段便是桥吊调度. 此时, 各到港船舶的停靠泊位、停靠时间和离港时间均已知. 桥吊调度问题即是为各船舶作业分配桥吊, 使得各船舶的装卸作业能在合理的离港时间前按时完成. 桥吊调度问题一般都以满足泊位调度计划为目标. 魏晓东[7]针对连续泊位分配与相邻船间桥吊迁移相结合的问题进行研究, 以最小化所有船舶的在港时间之和为目标, 提出一种新颖的桥吊迁移策略.

总体来看, 泊位调度问题和桥吊调度问题分开考虑的求解速度较快, 但存在局限性, 即当船舶作业时间预算偏长时, 会存在因作业时间充足而闲置的桥吊使资源浪费; 相反, 当预算偏短时, 桥吊会为按时完成作业而被迫加快装卸速度或频繁调度, 造成作业冲突. 寻找能同时求解泊位调度问题和桥吊调度问题的算法是很有必要的. 韩晓龙[8]等将岸线视为离散化的连续空间, 时间视为离散化的连续时间, 研究船舶动态到达条件下的泊位分配和桥吊调度问题. 作者首先在考虑到了每条船的有偏好位置这一因素下, 建立一个以所有船舶总在港时间最短为目标的泊位分配模型, 对到港的船舶进行泊位的指派和桥底数量的确定; 然后建立了一个在满足各船舶对桥吊需求计划下的最小桥吊调动次数为目标的桥吊调度的模型, 为船舶分派具体的桥吊.

以上是些对集装箱码头泊位-桥吊调度问题的传统研究情况, 其有个共性: 针对事先知道所有船舶作业相关信息的离线情况进行建模和设计算法. 然而, 在港口的日常生产中, 这样做过于理想化. 由于存在一些无法预料的因素, 如恶劣天气、设备故障等使船舶到达时间延迟; 或者因种种原因导致船舶作业量与计划作业量不符等不可靠的作业信息. 这使得原来的资源调度方案难以实施, 甚至于使港口作业的效率降低和资源浪费. 将在线算法应用于岸边资源调度可以有效应地解决这一问题.

早在 1998 年, Allan Borodin 将在线算法定义为: “一个计算机在线算法, 是在对未来的输入状况一无所知的情况下, 决定如何处理当前到来的输入项”[9]. 随着人们对该课题的深入研究, 在线调度算法的优化目标函数大体可分两类: MAXTHROUGHPUT 和 MINBUSY, 前者目标是在给定的预计处理机工作时间 T 的条件下, 使处理机处理的作业量最大化; 后者是最小化所有处理机工作的总时间[10]. Albers[11]给出在线算法的性能的一个衡量标准——算法的竞争比(competitive analysis), 即对于一个实例, 一个在线算法的调度结果成本跨度与离线算法的调度结果的最佳成本跨度的比值, 记作 c-competitive. 而被提出在线算法有最短剩余处理时间(SRPT)、先进先出(FIFO)、最短作业优先(SJF)、最高密度比优先(HDF)、多级反馈算法(MLF)等[12], 但这些算法都是针对处理机-进程作业问题提出的.

本文的主要工作是: 基于在线算法思想, 在桥吊可迁移的连续泊位分配模式下, 提出集装箱码头的泊位与桥吊资源的的优化模型, 并设计以当前资源调度最优化的在线调度算法. 利用算法复杂性平滑分析方法给出算法的平滑竞争比. 通过实验证实了所提出的在线调度算法可以有效地应用于船舶信息实时变化的集装箱码头生产实际中以提高码头作业效率.

## 2 问题建模

### 2.1 前提假设

本文中作以下假设:

A) 泊位水深能满足所有到港船舶吃水要求;

B) 船与船之间的安全空隙计算在船长内;

C) 每条船的装卸作业过程不允许中断;

D) 靠泊的船, 只有装卸作业完成后才离开;

E) 每条船的可靠信息(如, 船舶的到港时间, 总工作量, 最大桥吊数, 船长等信息)在到港时可知;

F) 每艘靠泊船有最大桥吊数, 其最小桥吊数为 1, 若分配到桥吊数为 0, 则推迟停泊;

G) 桥吊在统一轨道上, 不可跨越交叉迁移;

H) 所有桥吊的工作效率相同;

I) 桥吊迁移的时间忽略不计;

J) 只有当所有的桥吊都不工作了, 表示一个问题实例的结束.

### 2.2 模型表示

目标函数:

$$\min \sum_{j\in J}\left(C_j^A - R_j^A\right) \tag{1}$$

约束条件:

$$z_{ij'} + y_{tj'} + z_{ij''} + y_{tj''} \le 3 \tag{2}$$

$$\sum_{i=1}^{L} z_{ij} = L_{V_j} \tag{3}$$

$$H_j = \sum_{t\ge R_j}^{T} y_{tj} \tag{4}$$

$$\sum_{t=1}^{T}\left(y_{tj} * C_{Vj}(t)\right) = w_j \tag{5}$$

$$C_{Vj}(t) \le C_{Vj} \tag{6}$$

$$\sum_{j=1}^{V} C_{Vj}(t) \le C \tag{7}$$

$$\sum_{i=1}^{L-L_{Vj}+1} u_{ij} = 1 \tag{8}$$

$$\sum_{t=1}^{T} v_{tj} = 1 \tag{9}$$

$$R_j - t \le Q\left(1 - v_{tj}\right) \tag{10}$$

$$1 - z_{i''j} \le Q\left(1 - u_{i'j}\right) \tag{11}$$

$$1 - y_{t''j} \le Q\left(1 - v_{t'j}\right) \tag{12}$$

$$L_{b_{s_i,e_i}} > 0 \tag{13}$$

$$L_{b_{s_i,e_i}} - L_{Vj} \ge 0 \tag{14}$$

$$y_{t'''j} = 0 \tag{15}$$

$$v_{t'''j} = 0 \tag{16}$$

$$u_{i'''j} = 0 \tag{17}$$

$$z_{ij}, u_{ij}, y_{tj}, v_{ij} \in \{0,1\} \tag{18}$$

$j=1,2,...,V;\quad t=1,2,...,T;\quad i=1,2,...,L;\quad i'=1,...,L-L_{Vj}+1;$

$i''=i',...,i'+L_{Vj}-1;\quad t'=1,...,T-H_j+1;\quad t''=t',...,t'+H_{Vj}-1;$

$i'''=L-L_{Vj}+2,...,L;\quad t'''=1,...,R_j-1;$

$L$ 为岸线长度; $T$ 为计划期长度; $R_j$ 为船 $j$ 到港时间; $Q$ 表示无穷大的数; $C$ 为港口总桥吊数; $w_j$ 为船舶 $j$ 的总工作量; $H_j$ 为船 $j$ 装卸作业时间; $C_j^A$ 表示船 $j$ 实际离港时间; $L_{b_{si,ei}}$ 表示空闲泊位 $b_{si,ei}$ 的长度; $C_{Vj}$ 表示船 $j$ 最多可承载桥吊数; $J=\{1,2,...,V\}$ 表示到达的船舶集合; $L_{Vj}$ 为船 $j$ 的长度, 包含安全预留长度; $y_{tj}$ 表示若 $t$ 时刻船 $j$ 在泊则取 1, 否则取 0; $z_{ij}$ 表示若岸线 $i$ 处被船 $j$ 占用则取 1 否则取 0; $v_{tj}$ 表示若船 $j$ 开始泊位则取 1 否则取 0; $u_{tj}$ 表示若船 $j$ 从岸线 $i$ 开始泊位则取 1 否则取 0.

式(1)表示式所有船舶总在港时间最小化; 式(2)表示船舶之间不可重叠; 式(3)表示船长度的限制; 式(4)表示任意船装卸作业时间; 式(5)表示任意船总的工作量为每时刻占有的桥吊的总和; 式(6)表示任意船每时刻所占用的桥吊数约束; 式(7)表示对港口总桥吊的约束; 式(8)和(9)表示任意船只泊位一次; 式(10)表示船只有到港后才能泊位; 式(11)和(12)表示船在时间上是连续的; 式(13) 表示空闲泊位的泊位长不为 0; 式(14)表示可为在港船舶分配的泊位长不小于船长; 式(15)、(16)和(17)表示船只有泊位后才可作业; 式(18)表示对变量的约束.

## 3 算法的提出

### 3.1 桥吊可迁移的港口泊位分配问题

假设某集装箱码头在一个计划期 T 内, 到达的作业船舶集合 J={1,2,…,V} 和码头空闲泊位集 $B=\{b_{s_1,e_1},b_{s_2,e_2},...,b_{s_i,e_i},...,b_{s_m,e_m}\}$, 其中 $s_i$ 表示空泊位的起始位置, $e_i$ 表示空泊位的结束位置. 对于每个作业 j(即作业船舶, 以下简称作业)有属性: 工作量 $w_j$; 到达时间 $R_j$; 所需最大桥吊数 $C_{Vj}$; 船长 $L_{Vj}$; 最佳处理时间 $p_j^{OPT}$ 表示作业的工作量与最大桥吊数的比值; 桥吊-船长密度定义为 $\beta_j = C_{Vj}/L_{Vj}$; $C_{Vj}(t)$ 表示 t 时刻船舶 j 拥有的桥吊数. 泊位 $b_{s_i,e_i}$ 的属性有: 空闲泊位中可用桥吊数 $C_{b_{s_i,e_i}}$; 空闲泊位长度 $L_{b_{s_i,e_i}}$; 将它们的比值称为桥吊-泊位密度, 记为 $\alpha_{b_{s_i,e_i}} = C_{b_{s_i,e_i}}/L_{b_{s_i,e_i}}$.

本章要做的就是基于在线算法思想为到来的船舶 j 分配合适的空闲泊位 $b_{s_1,e_1}$ 和桥吊, 使得当前已泊位的船舶总在刚时间最小, 即一个作业集经在线算法 A 调度的总在港时间 $\min F^A = \min\sum_{j\in J} C_j^A - R_j^A$.

### 3.2 泊位分配及桥吊迁移决策

定理 1. 对于作业 j, 若有 s 个可选的合适泊位 $B=\{b_{s_1,e_1},b_{s_2,e_2},...,b_{s_i,e_i},...,b_{s_m,e_m}\}$, 则选择与 $\beta_j = C_{Vj}/L_{Vj}$ 最接近的 $\alpha_{b_{s_i,e_i}} = C_{b_{s_i,e_i}}/L_{b_{s_i,e_i}}$ 泊位, 可使泊位 $b_{s_i,e_i}$ 的岸线利用率和桥吊利用率提高.

证明: 首先, 文献[21]中基于柔性靠泊系统的集装箱码头岸线规划提出的岸线利用率 $=\frac{\sum 在泊船长\times 在泊时间}{岸线总长度\times 日历时间}\times 100\%=\frac{\sum L_{vj}*H_j}{L*TT}\times 100\%$, 其中 TT 表示日历时间. 且文献中指出当到港船型组合一定时, 岸线越长, 岸线利用率越高; 岸线规模一定时, 到港船型越大, 岸线利用率越高. 文献[22]中有描述了桥吊利用率$=\frac{\sum 单个桥吊运行时间}{总桥吊数\times 日历时间}\times 100\%=\frac{\sum H_j}{C*TT}\times 100\%$. 可发现, 在日历时间内, 对于泊位 $b_{s_i,e_i}$, 若增加被停泊岸线长度和运行桥吊数, 即增加 $L_{Vj}$、$C_{Vj}$, 可使得岸线利用率和桥吊利用率提高. 然而, 对于被停泊岸线长度过大, 运行桥吊数过小情况(即停靠 $\alpha_{b_{si,ei}}<\beta_j$ 的船舶), 会使得泊位 $b_{s_i,e_i}$ 的桥吊运用率不高; 对于运行被停泊岸线长度过小, 桥吊数过大情况(即停靠 $\alpha_{b_{si,ei}}>\beta_j$ 的船舶), 会使泊位 $b_{s_i,e_i}$ 的岸线运用率不高. 显然, 是一个凸函数, 故选择与 $\beta_j$ 最接近的 $\alpha_i$ 空闲泊位 $b_{s_i,e_i}$, 可使岸线利用率和桥吊利用率都提高.

定理 2.　对于某作业因完成而释放出的空闲桥吊, 若迁移给未达最佳桥吊数的左、右邻居, 可使总处理时间减小.

证明:　假设 t 时刻, 作业 j 有工作量 $W_j(t)$, 桥吊数 $C_{Vj}(t)$, 则完成作业需要时间 $p1_j=\frac{W_j(t)}{C_{Vj}(t)}$; 而此时得到完成作业的邻居迁移桥吊 x($x\neq 0$), 则完成作业需要的时间 $p2_j=\frac{W_j(t)}{C_{Vj}(t)+x}$, 很明显 $p1_j<p2_j$.

定理 3.　对于任意船舶 j, 其所需要的最大桥吊数 $C_{Vj}$ 与其需要的处理时间 $p_j$ 成正比, 且 $C_{Vj}$ 越小船舶可停泊泊位越多.

证明: 对于船舶 j, 有 $L_{Vj}=\eta C_{Vj}+\theta_j$ 及 $W_j=\eta C_{Vj}G_j$, 则可知船舶作业 j 需要的处理时间 $p_j=\frac{\eta C_{Vj}G_j}{C_{Vj}(t)}$ 与 $C_{Vj}$ 成正比, 而船舶长度 $L_{Vj}$ 与 $C_{Vj}$ 也是成正比, 故 $C_{Vj}$ 越小的船舶 j, 其要处理的时间 $p_j$ 越小, 且船舶长度 $L_{Vj}$ 越短, 故可停泊的泊位越多. 其中 $\eta$ 是桥吊之间的最佳距离, $\theta_j$ 是船舶 j 的非桥吊占用长度, $G_j$ 是船舶 j 的单个桥吊在最佳工作范围内能处理到的工作量密度, 通常船舶越大, 其越大.

### 3.3 求解算法

针对上面叙述问题, 给出一个泊位分配的在线算法 I 和一个桥吊迁移算法 II. 算法 I 的主要思想是: 基于先来先服务的原则对桥吊需求最小的作业优先选择泊位, 而选泊位的原则是选择泊位集 B 中泊位的 $\alpha_{b_{si,ei}}$, 同船舶的 $\beta_j$ 最接近. 这样既可提高岸线与桥吊的使用率, 也达到当前船舶在港时间最小化. 桥吊迁移算法 II 的思想是: 在有船舶完成作业离港后, 若其待收回泊位左、右邻居泊位有未分配满桥吊的船舶, 则进行桥吊迁移; 否则, 待收回泊位与左、右空闲邻居泊位合并或直接加入空闲泊位集 B, 这样可以提高未分配到最佳桥吊数的船舶工作效率, 从而降低在港时间. 且算法 I 和 II 使得每一时刻的岸线泊位和桥吊都是最大化利用, 对于任何问题实例, 全部船舶作业总会被完成, 泊位将完全空闲, 此时算法将终止, 可见算法是正确可行的.　针对上面叙述问题, 给出一个泊位分配的在线算法 I 和一个桥吊迁移算法 II. 算法 I 的主要思想是: 基于先来先服务的原则对桥吊需求最小的作业优先选择泊位, 而选泊位的原则是选择泊位集 B 中泊位的, 同船舶的最接近. 这样既可提高岸线与桥吊的使用率, 也达到当前船舶在港时间最小化. 桥吊迁移算法 II 的思想是: 在有船舶完成作业离港后, 若其待收回泊位左、右邻居泊位有未分配满桥吊的船舶, 则进行桥吊迁移; 否则, 待收回泊位与左、右空闲邻居泊位合并或直接加入空闲泊位集 B, 这样可以提高未分配到最佳桥吊数的船舶工作效率, 从而降低在港时间. 且算法 I 和 II 使得每一时刻的岸线泊位和桥吊都是最大化利用, 对于任何问题实例, 全部船舶作业总会被完成, 泊位将完全空闲, 此时算法将终止, 可见算法是正确可行的.

## 4　算法性能平滑分析

算法性能平滑分析是由 Spielman 和 Teng 提出[13], 通过混合分析算法的平均情况复杂性和最坏情况复杂性, 用来解释一些算法的最坏情况复杂性很坏, 但在实际应用却很有效的原因. 其基本思想是: 将初始输入实例作随机扰动后所得输入实例作为算法 A 的输入, 对算法 A 的性能进行分析.

在线算法的平滑分析主要分析其平滑竞争比. 平滑竞争比即是, 对于经概率分布函数 $f$ 平滑处理过的全部输入实例的情况下, 在线算法 A 最小化目标函数所花的成本与最佳(OPT)成本之间最大比值 $\square=\left(\sup_{\bar{I}} E_{I\in_f N(\bar{I})}\left[\frac{A_I}{OPT_I}\right]\right)$, 其中 $\bar{I}$ 是由概率分布函数 $f$ 对 $N(\bar{I})$ 邻域平滑过的全部输入实例, 本文将用 A 表示

本文提出的算法的调度结果, 而用 OPT 表示离线型算法调度结果的最佳成本.

Input：港口初始泊位集 B={ $b_{s_1,e_1}$ }, 到来的船舶集 J={1,2,…,j}.
Output：船舶分配的泊位及桥吊数调度序列, 所有船舶的总靠港时间.
1) 把第一个到来船舶分配给泊位 $b_{s_1,e_1}$, 并在修改泊位集 B 中空闲泊位属性;
2) FORALL 未靠泊的到港船舶 DO
IF 有空闲泊位 THEN
IF 只有一条船
THEN 把它分配给 $|\beta_j-\alpha_{b_{s_1,e_1}}|$ 最小的能停靠的空闲泊位, 并分配给最佳桥吊数;
IF 有多条船
THEN 靠港时间最长且 $C_{Vj}$ 最小的船舶优先选择 $|\beta_j-\alpha_{b_{s_1,e_1}}|$ 最小的能停靠的空闲泊位, 并优先满足桥吊数;
ELSE 将未泊位的船推迟下一时间段泊位
3) END
4) 若港口无未处理完船舶, 则结束本案例;

**算法 I**

Input：港口空闲泊位集 B, 待收回泊位 b.
Output：泊位集.
1) FORALL 有船舶 j 离开港口 DO
IF 船舶 j 前或后是否有紧挨着的未完成作业的船舶
THEN 将桥吊按剩余处理时间最短优先满足原则迁移给邻居船舶, 修改空出泊位 b 桥吊数
ELSE 将空出泊位 b 与其前后空闲泊位合并, 修改泊位集 B
IF 泊位 b 桥吊数为 0
THEN 将泊位 b 直接并到迁移桥吊数最多的邻居停船泊位
ELSE 将泊位 b 与前后空闲泊位合并, 修改泊位集 B
2) END

**算法 II**

与平滑竞争比相关性较高的则是对输入实例进行平滑的模型, 本文对输入实例的预处理时间 $\overline{p}_j$ 进行平滑处理的模型则是局部位随机模型(Partial Bit Randomization Model)[29]. 定义为对输入实例的预处理时间的低 k 位用[1,$2^k$]上具有分布 f 的随机数来替代. 即 $p_j=2^k\left\lfloor\frac{\overline{p}_j-1}{2^k}\right\rfloor+\varepsilon_j$, 其中 $\varepsilon_j\xleftarrow{f}[1,2^k]$.

为了方便描述, 我们定义 $\phi_j=2^k\left\lfloor\frac{\overline{p}_j-1}{2^k}\right\rfloor$, 于是 $p_j=\phi_j+\varepsilon_j$.

下面我们还要知道几个事实:

事实 1. 对于 $\overline{p}_j\in[1,2^k]$, 则 $\phi_j=0$ 及 $p_j\in[1,2^k]$, 且 $P[p_j\le x]=P[\varepsilon_j\le x]$, 其中 $x\in[1,2^k]$.

事实 2. 对于 $\overline{p}_j\in(2^{i-1},2^i]$, 其中 $k<i\le K$, 则 $2^{i-1}\le\phi_j\le 2^i-2^k$ 及 $p_j\in(2^{i-1},2^i]$.

事实 3. $F^A=\sum_{j\in J}F_j^A=\int_t\delta^A(t)\,dt$.

事实 4. $F^A\ge\sum_{j\in J}p_j$.

事实 5. 对于任意时间 t, 港口可以停泊的船舶数是有限的, 即同时可处理的作业数是有上限的, 我们这里可取上限为 $\omega$(这是随实际港口情况而变的).

分析前, 我们还要介绍一些概念. 通过作业的处理时间, 对其分类: 若 $p_j\in(2^{i-1},2^i]$, 则说作业 j 是第 i 类(很明显, 作业 j 要在 $2^i$ 时间内完成), 并可记为 $Cl_j=i$, 当然, 取 K 为最大的类. 用 $\delta^A(t)$ 表示对于调度算法 A, 在 t 时刻处于激活态的作业数. 若一个作业的预处理时间 $p_j\ge(1+\gamma_k)2^{i-1}$, 则说这个作业是幸运的作业, 其中, $\gamma_k=\min[\frac{1}{\sqrt{2}}(\frac{\sigma}{2^{k-1}}),2^{k-K}]$. 设变量 $X_j^l=1$ 表示作业 j 是幸运作业, 否则为 0. 而用 $\delta^l(t)$ 表示对于算法 A, 在 t 时刻处于激活态的幸运作业数. 对于 i 类中, 正在处理态的作业称为顶端作业, 而用 h(t)表示 t 时刻的所有顶端作业数. $V_{=i}^A(t)$ 表示对于算法 A, 在 t 时刻处于激活态的作业中, 第 i 类内未处理作业的时间总和.

### 3.1 平均情况复杂性

引理 1. 在 t 时刻的处于激活态的幸运作业, 则有 $\delta^l(t)\le h(t)+4\frac{\delta^{OPT}(t)}{1+\gamma_k}$.

证明: 我们设 $k_1$, $k_2$ 分别表示 t 时刻处于激活态的作业对应类的最小类和最大类. 就 t 时刻, 处于激活态的幸运作业来说, 要么是在处理的顶端作业, 要么是等待处理的作业, 于是, 很容易有 $\delta^l(t)\le h(t)+\sum_{i=k_1}^{k_2}\frac{V_{=i}^A(t)}{(1+\gamma_k)2^{i-1}}$.

$$\sum_{i=k_1}^{k_2}\frac{V_{=i}^{A}(t)}{2^{i-1}}=\sum_{i=k_1}^{k_2}\left[\frac{V_{=i}^{OPT}(t)+\Delta V_{=i}(t)}{2^{i-1}}\right]\le 2\delta_{\ge k_1,\le k_2}^{OPT}(t)$$

$$+\sum_{i=k_1}^{k_2}\frac{\Delta V_{=i}(t)}{2^{i-1}}\le 2\delta_{\ge k_1,\le k_2}^{OPT}(t)+\sum_{i=k_1}^{k_2}\frac{\max\Delta V_{=i}(t)}{2^{i-1}}\le$$

$$2\delta_{\ge k_1,\le k_2}^{OPT}(t)+2\delta_{\ge k_1,\le k_2}^{OPT}(t)\le 4\delta_{\ge k_1,\le k_2}^{OPT}(t)+4\delta_{\le k_1-1}^{OPT}(t)\ =4\delta_{\le k_2}^{OPT}(t)$$

上式中第 2 个不等式是因为对于类 i 中最大预处理时间为 $2^i$. 对于 i 类的一个作业, 由 A 算法决策的在 t 时刻未处理时间与最佳方案(OPT)的未处理时间之间的差值只会在$[0,2^i]$, 即两种方案中作业都处理完或为处理及前者未处理而后者处理完了. 第 5 个不等式是因为 $\Delta V_{\le k_1-1}(t)=0$. 所以在 t 时刻的处于激活态的幸运作业, 有 $\delta^{I}(t)\le h(t)+4\frac{\delta^{OPT}(t)}{1+\gamma_k}$ 成立. 得证.

引理 2[14]. 对于 $\overline{p}_j\in(2^{i-1},2^i]$ 的作业 j, 其中 $k<i\le K$, 则 $P[X_j^I=1]\ge\frac{1}{2}$.

引理 3[14]. 若 $\varepsilon$ 取之于对称分布区间$[1,2^k]$, 且有 $\mu=2^{k-1}+1/2$, 则对于任意 $\lambda$, $0\le\lambda\le 2^k-\mu$, 则 $P[|\varepsilon-\mu|\ge\lambda]\ge\left(\frac{\sigma}{2^{k-1}}\right)^2-\left(\frac{\lambda}{2^{k-1}}\right)^2$.

引理 4[14]. 对于 $\overline{p}_j\in[1,2^k]$ 的作业 j, 其中 $0\le i\le k$, 则 $P[X_j^I=1\mid CI_j=i]\ge\left(\frac{\sigma}{2^k}\right)^2$.

引理 5. 对于 $j\in S$, 若 $P[X_j^I=1\mid S(t)]\ge\varphi$, 则有 $P[\delta^I(t)<\frac{1}{2}\varphi\delta(t)\mid S(t)]\le e^{-\frac{\varphi S}{8}}$. 其中, $S\subseteq J$ 表示 t 时刻处于激活态的作业数, 而 S(t)表示 t 时刻处于激活态作业为 S 的事件及令 $\varphi=\left(\frac{\sigma}{2^k}\right)^2$.

证明: 有 $E[\delta^I(t)\mid S(t)]\ge P[X_j^I=1\mid S(t)]*S=\varphi S$ 根据标准切尔诺夫界 (Chernoff bound), 有 $P[\delta^I(t)<\frac{1}{2}\varphi\delta(t)\mid S(t)]=P[\delta^I(t)<\frac{1}{2}\varphi S\mid S(t)]\le P[\delta^I(t)<\frac{1}{2}\varphi E[\delta^I(t)\mid S(t)]\mid S(t)]\le e^{-\frac{\varphi S}{8}}$.

引理 6[14]. $P[\sum_j p_j<\sum_j\phi_j+\frac{n(2^k+1)}{8}]<e^{-\frac{n}{16}}$.

定理 4. 对于任意输入实例 $\overline{I}$ 及合适的概率分布函数 $f$, 有 $E_{I\in_f N(\overline{I})}\left[\frac{F^A}{F^{OPT}}\right]=O\left(\left(\frac{2^k}{\sigma}\right)^3+\left(\frac{2^k}{\sigma}\right)^2\left(8\omega+32+2^{K-k}\right)\right)$.

证明: 首先定义事件 $\xi=\left(\sum_j p_j\ge\sum_j\phi_j+\frac{n(2^k+1)}{8}\right)$, 对应的逆事件是 $\overline{\xi}$. 对于实例 I, 有 $D_I=\{t:\delta_I^A(t)\le\frac{2}{\varphi}\delta_I^I(t)\}$ 和 $\overline{D}_I=\{t:\delta_I^A(t)>\frac{2}{\varphi}\delta_I^I(t)\}$. 为了描述的方便我们将省去写 $I\in_f N(\overline{I})$. 由事实可知 $F^{OPT}=\int\delta^{OPT}(t)\,dt\ge\sum_j p_j\ge\sum_j\phi_j+\frac{n(2^k+1)}{8}$, 而由事实 5 可知 t 时刻的顶端作业总处理时间是不会超过 $\sum_j p_j$, 而 $\sum_j p_j=\sum_j\phi_j+\frac{n(2^k+1)}{2}$. 对于事件 $\overline{\xi}$ 情况时, 由引理 6 可知.

$$E\left[\frac{\int_{t\in D}\delta^A(t)\,dt}{F^{OPT}}\mid\xi\right]P[\xi]\le E\left[\frac{\int_{t\in D}\frac{2}{\varphi}\delta^I(t)\,dt}{F^{OPT}}\mid\xi\right]P[\xi]\le$$

$$E\left[\frac{\int_{t\in D}\frac{2}{\varphi}\left(h(t)+4\frac{\delta^{OPT}(t)}{1+\gamma_k}\right)dt}{F^{OPT}}\mid\xi\right]P[\xi]\le E\left[\frac{\int_{t\in D}\frac{2}{\varphi}h(t)dt}{F^{OPT}}\mid\xi\right]$$

$$P[\xi]+\frac{8}{(1+\gamma_k)\varphi}\le E\left[\frac{\int_{t\in D}\frac{2}{\varphi}\omega\,dt}{F^{OPT}}\mid\xi\right]P[\xi]+\frac{8}{(1+\gamma_k)\varphi}\le$$

$$\frac{\frac{2}{\varphi}\omega E\left[\sum_j p_j\right]}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}+\frac{8}{(1+\gamma_k)\varphi}\le\frac{8}{\varphi}\omega+\frac{8}{(1+\gamma_k)\varphi}$$

$$E\left[\frac{\int_{t\in\overline{D}}\delta^A(t)\,dt}{F^{OPT}}\mid\xi\right]P[\xi]\le\frac{E\left[\int_{t\in\overline{D}}\delta^A(t)dt\mid\xi\right]P[\xi]}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}\le$$

$$\frac{E\left[\int_{t\in\overline{D}}\delta^A(t)dt\right]}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}=\frac{\int_{t\ge0}E\left[\delta^A(t)\mid t\in\overline{D}\right]P[t\in\overline{D}]\,dt}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}$$

$$=\frac{\int_{t\ge0}\sum_{s=1}^{n}sP[\delta^A(t)=s\mid t\in\overline{D}]P[t\in\overline{D}]\,dt}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}=$$

$$\frac{\int_{t\ge0}\sum_{s=1}^{n}sP[t\in\overline{D}\mid\delta^A(t)=s]P[\delta^A(t)=s]\,dt}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}\le$$

$$\frac{\int_{t\ge0}\sum_{s=1}^{n}se^{-\frac{\varphi s}{8}}P[\delta^A(t)=s]\,dt}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}\le\frac{8}{\varphi}\frac{\int_{t\ge0}\sum_{s=1}^{n}P[\delta^A(t)=s]\,dt}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}$$

$$=\frac{\frac{8}{\varphi}E\left[\sum_j p_j\right]}{\sum_j\phi_j+\frac{n(2^k+1)}{8}}\le\frac{32}{\varphi}$$

$$E\left[\frac{F^A}{F^{OPT}}\right]=E\left[\frac{F^A}{F^{OPT}}\mid\xi\right]P[\xi]+E\left[\frac{F^A}{F^{OPT}}\mid\overline{\xi}\right]P[\overline{\xi}]=$$

$$E\left[\frac{\int_{t\in D}\delta^A(t)dt}{F^{OPT}}\mid\xi\right]P[\xi]+E\left[\frac{\int_{t\in\overline{D}}\delta^A(t)dt}{F^{OPT}}\mid\xi\right]P[\xi]+$$

$$E\left[\frac{F^A}{F^{OPT}}\mid\overline{\xi}\right]P[\overline{\xi}]\le\frac{8}{\varphi}\omega+\frac{8}{(1+\gamma_k)\varphi}+\frac{32}{\varphi}+ne^{-\frac{n}{16}}$$

$$\le O\left(\left(\frac{2^k}{\sigma}\right)^3+\left(\frac{2^k}{\sigma}\right)^2\left(8\omega+32+2^{K-k}\right)\right)$$

### 4.2 最坏情况复杂性

这里我们会借用文献[15]中对随机化在线算法分析的方法, 同样定义不同类的竞争对手来与本文算法来比较. 首先, 对于输入实例队列完全由算法实现和平滑模型的分布函数决定的对手, 称为随意型对手, 这种对手而对随机化算法的运行结果却不知道, 故事

比较弱的对手; 对于输入实例队列的抉择取决于 t 时刻算法的状态的对手, 称为自适应型算法, 这种对手知道随机化算法的随机过程等一切信息, 故是比较强劲的对手. 这两种对手都适用于使得输入实例成本最佳的离线算法.

引理 7[14]. 对于任何确定性算法 A, 在 $k \le K/3$ 时, 以 $\Omega\left(2^{\frac{K}{6}\frac{k}{2}}\right)$ 的平滑竞争比击败随意型对手, 其中平滑实例模型是局部位随机模型.

定理 5. 对于任何确定性算法 A, 以 $\Omega\left(2^{K-k}\right)$ 的平滑竞争比击败自适应型对手, 其中平滑实例模型是局部位随机模型.

正如文献[15]中说明的那样自适应型的对手要比任意型的对手要优秀些. 在文献[16]中, Motwani 已经给出非可预见性算法的下界是 $\Omega(2^K)$, 本文也用类似与其的思想得出了对于任何确定性算法 A, 以 $\Omega\left(2^{K-k}\right)$ 的平滑竞争比击败自适应型对手, 其中平滑实例模型是局部位随机模型.

## 5 算例

我们结合实际情况, 提取一组用于本文实验的案例: 某港口的岸线长度为 100, 可用桥吊数为 17. 依次到港的船舶信息如表 1 所示.

表 1 船舶信息

| 编号 | 到达时间 | 船长 | 最大桥吊数 | 最佳处理时间 | 工作量 |
|---|---|---|---|---|---|
| 1 | 1.0 | 22 | 4 | 5.5 | 22.0 |
| 2 | 2.5 | 16 | 3 | 4.4 | 13.2 |
| 3 | 4.0 | 22 | 4 | 5.5 | 22.0 |
| 4 | 5.5 | 10 | 2 | 3.6 | 7.2 |
| 5 | 7.0 | 16 | 3 | 4.4 | 13.2 |
| 6 | 8.5 | 28 | 5 | 6.7 | 33.5 |
| 7 | 9.5 | 10 | 2 | 3.6 | 7.2 |
| 8 | 11.0 | 16 | 3 | 4.4 | 13.2 |
| 9 | 11.0 | 28 | 5 | 6.7 | 33.5 |
| 10 | 12.5 | 16 | 3 | 4.4 | 13.2 |
| 11 | 14.0 | 10 | 2 | 3.6 | 7.2 |
| 12 | 15.5 | 22 | 4 | 5.5 | 22.0 |
| 13 | 15.5 | 28 | 5 | 6.7 | 33.5 |
| 14 | 17.0 | 22 | 4 | 5.5 | 22.0 |
| 15 | 18.0 | 10 | 2 | 3.6 | 7.2 |

表 2 本文算法调度结果

| 时段 | 编号 | 泊位 | 船拥有桥吊数 | 空闲泊位/桥吊 | 预计结束时间 |
|---|---|---|---|---|---|
| 1.0 | 1 | 0~22 | 4 | 22~100/13 | 6.5 |
| 2.5 | 1 | 0~22 | 4 | 38~100/10 | 6.5 |
| | 2 | 22~38 | 3 | | 6.9 |
| 4.0 | 1 | 0~22 | 4 | 60~100/6 | 6.5 |
| | 2 | 22~38 | 3 | | 6.9 |
| | 3 | 38~60 | 4 | | 9.5 |
| 5.5 | 1 | 0~22 | 4 | 70~100/4 | 6.5 |
| | 2 | 22~38 | 3 | | 6.9 |
| | 3 | 38~60 | 4 | | 9.5 |
| | 4 | 60~70 | 2 | | 9.1 |
| 6.5 | 2 | 22~38 | 3 | 0~22/4 | 6.9 |
| | 3 | 38~60 | 4 | 70~100/4 | 9.5 |
| | 4 | 60~70 | 2 | | 9.1 |
| 6.9 | 3 | 38~60 | 4 | 0~38/7 | 9.5 |
| | 4 | 60~70 | 2 | 70~100/4 | 9.1 |
| 7.0 | 3 | 38~60 | 4 | 16~38/4 | 9.5 |
| | 4 | 60~70 | 2 | 70~100/4 | 9.1 |
| | 5 | 0~16 | 3 | | 11.4 |
| 8.5 | 3 | 38~60 | 4 | 16~38/4 | 9.5 |
| | 4 | 60~70 | 2 | 98~100/0 | 9.1 |
| | 5 | 0~16 | 3 | | 11.4 |
| | 6 | 70~98 | 4 | | 16.875 |
| 9.1 | 3 | 38~60 | 4 | 16~38/4 | 9.5 |
| | 5 | 0~16 | 3 | 60~70/1 | 11.4 |
| | 6 | 70~98 | 5 | 98~100/0 | 15.32 |
| 9.5 | 5 | 0~16 | 3 | 26~70/7 | 11.4 |
| | 6 | 70~98 | 5 | 98~100/0 | 15.32 |
| | 7 | 16~26 | 2 | | 13.1 |
| 11.0 | 5 | 0~16 | 3 | 98~100/0 | 11.4 |
| | 6 | 70~98 | 5 | | 15.32 |
| | 7 | 16~26 | 2 | | 13.1 |
| | 8 | 26~42 | 3 | | 15.4 |
| | 9 | 42~70 | 4 | | 19.375 |
| 11.4 | 6 | 70~98 | 5 | 0~16/3 | 15.32 |
| | 7 | 16~26 | 2 | 98~100/2 | 13.1 |
| | 8 | 26~42 | 3 | | 15.4 |
| | 9 | 42~70 | 4 | | 19.375 |
| 12.5 | 6 | 70~98 | 5 | 98~100/0 | 15.32 |
| | 7 | 16~26 | 2 | | 13.1 |
| | 8 | 26~42 | 3 | | 15.4 |

| | | | | | |
|---|---|---|---|---|---|
| | 9 | 42~70 | 4 | | 19.375 |
| | 10 | 0~16 | 3 | | 16.9 |
| 13.1 | 6 | 70~98 | 5 | 16~26 | 15.32 |
| | 8 | 26~42 | 3 | 98~100/0 | 15.4 |
| | 9 | 42~70 | 4 | | 19.375 |
| | 10 | 0~16 | 3 | | 16.9 |
| 14.0 | 6 | 70~98 | 5 | 98~100/0 | 15.32 |
| | 8 | 26~42 | 3 | | 15.4 |
| | 9 | 42~70 | 4 | | 19.375 |
| | 10 | 0~16 | 3 | | 16.9 |
| | 11 | 16~26 | 2 | | 17.6 |
| 15.32 | 8 | 26~42 | 3 | 70~100/4 | 15.4 |
| | 9 | 42~70 | 5 | | 18.564 |
| | 10 | 0~16 | 3 | | 16.9 |
| | 11 | 16~26 | 2 | | 17.6 |
| 15.4 | 9 | 42~70 | 5 | 26~42/3 | 18.564 |
| | 10 | 0~16 | 3 | 70~100/4 | 16.9 |
| | 11 | 16~26 | 2 | | 17.6 |
| 15.5 | 9 | 42~70 | 5 | 26~42/3 | 18.564 |
| | 10 | 0~16 | 3 | 92~100/0 | 16.9 |
| | 11 | 16~26 | 2 | | 17.6 |
| | 12 | 70~92 | 4 | | 21.0 |
| 16.9 | 9 | 42~70 | 5 | 0~16/3 | 18.564 |
| | 11 | 16~26 | 2 | 26~42/3 | 17.6 |
| | 12 | 70~92 | 4 | 92~100/0 | 21.0 |
| 17.6 | 9 | 42~70 | 5 | 28~42/3 | 18.564 |
| | 12 | 70~92 | 4 | 92~100/0 | 21.0 |
| | 13 | 0~28 | 5 | | 24.3 |
| 18.0 | 9 | 42~70 | 5 | 38~42/1 | 18.564 |
| | 12 | 70~92 | 4 | 92~100/0 | 21.0 |
| | 13 | 0~28 | 5 | | 24.3 |
| | 15 | 28~38 | 2 | | 21.6 |
| 18.564 | 12 | 70~92 | 4 | 60~70/2 | 21.0 |
| | 13 | 0~28 | 5 | 92~100/0 | 24.3 |
| | 14 | 38~60 | 4 | | 24.064 |
| | 15 | 28~38 | 2 | | 21.6 |
| 21.0 | 13 | 0~28 | 5 | 60~100/6 | 24.3 |
| | 14 | 38~60 | 4 | | 24.064 |
| | 15 | 28~38 | 2 | | 21.6 |
| 21.6 | 13 | 0~28 | 5 | 28~38/2 | 24.3 |
| | 14 | 38~60 | 4 | 60~100/6 | 24.064 |
| 24.064 | 13 | 0~28 | 5 | 28~100/12 | 24.3 |
| 24.3 | NULL | NULL | NULL | 0~100/17 | NULL |

表 2 是用 java 代码实现的本文提出的在线算法在 myeclipse 内运行结果. 表中标灰色底纹的编号对应的船舶, 是由算法 I 中$\left|\beta_j-\alpha_{b_{s_i,e_i}}\right|$最小决策决定船舶选着的泊位; 而加下滑线的编号对应的船舶, 则是第一阶段分配的桥吊数为当前最多可得到数, 后续加波浪线的则是在算法 II 处理后, 收回泊位的空闲桥吊迁移后的结果. 图 1 是本算法运行后船舶的泊位位置分布图, 其中没有考虑模型的第四条假设.

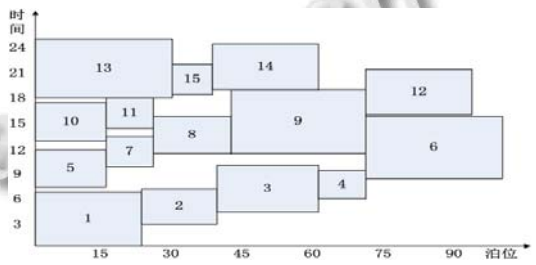


图 1　算法调度后的泊位分布图

很明显, 本算例的所有船舶在离线情况下最佳总在港时间不会低于 74.1, 而本文提供在线算法得到的总在港时间 $\sum_{j\in J}\left(C_j-R_j\right)$= 5.5+4.4+5.5+6.6+4.4+6.82+3.6+4.4+7.564+4.4+3.6+5.5+8.8+7.064+3.6=78.748, 可见是很接近离线情况下的最佳总在港时间, 说明了算法的有效性.

## 6　结语

本文结合实际情况, 对在线算法在港口资源分配上应用问题进行深入研究, 给出合适的在线算法对港口的泊位和桥吊资源进行在线合理分配, 并运用平滑分析思想对算法的平滑复杂度进行分析, 得到较理想结果. 当然, 本文是首个将在线算法运用在在港口资源分配问题的文章, 故还有许多待完善的地方. 如: 港口同时在线的船舶数与港口的长及桥吊数之间的关系等. 希望有感兴趣的读者可以一起探讨研究.

## 参考文献

1 Bierwirth C, Meisel F. A survey of berth allocation and quay crane scheduling problems in container terminals. European Journal of Operational Research, 2010, 202(3): 615–627.

2 Lai K K, Shih K. A Study of container berth allocation. Journal of Advanced Transportation, 1992, 26(1): 45– 60.

3 蔡芸,张艳伟.集装箱码头泊位分配的仿真优化方法.中国工

程机械学报,2006,4(2) :228–232.
4 Park YM, Kim KH. A scheduling method for berth and quay cranes. OR Spectrum, 2003, 25(1): 1–23.
5 张佳运,卢刚.集装箱码头连续泊位动态分配优化模型及算法.北方交通,2011,4:127–128.
6 张海滨,张纪会,宣金钊.集装箱码头泊位调度问题的启发式算法研究.青岛大学学报,2010,25(4):57–60.
7 魏晓东,杨智应.基于桥吊迁移的集装箱码头连续泊位分配算法研究.计算机应用研究,2013,30(10):2972–2976.
8 赵坤强,韩晓龙,梁承姬.连续泊位下集装箱港口泊位与桥吊协同调度优化研究.武汉理工大学学报,2012,33(11):60–65.
9 Borodin A, El-Yaniv R. Online computation and competitive analysis. Cambridge University Press, 1998.
10 Shalom M, Voloshin A, Wong PWH, et al. Online optimization of busy time on parallel machines. Theory and Applications of Models of Computation. Springer Berlin Heidelberg, 2012: 448–460.
11 Albers S. Better bounds for online scheduling. SIAM Journal on Computing, 1999, 29(2): 459–473.
12 Pruhs K, Sgall J, Torng E. Online scheduling. Handbook of Scheduling: Algorithms, Models, and Performance Analysis, 2004: 15–19.
13 Spielman D, Teng SH. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. Proc. of the Thirty-Third Annual ACM Symposium on Theory of Computing. 2001. 296–305.
14 Becchetti L, Leonardi S, Marchetti-Spaccamela A, et al. Average-case and smoothed competitive analysis of the multilevel feedback algorithm. Mathematics of Operations Research, 2006, 31(1): 85–108.
15 Borodin A, El-Yaniv R. Online Computation and Competitive Analysis. Cambridge University Press, 1998.
16 Motwani R, Phillips S, Torng E. Non-clairvoyant scheduling. Theoretical Computer Science, 1994, 130: 17–47.