汽车装配线生产计划与调度的集成优化方法

陈 琳, 严洪森, 刘 通, 刘霞玲
(东南大学 复杂工程系统测量与控制教育部重点实验室, 江苏 南京 210096)

摘 要: 为提高汽车装配线的生产效率, 优化资源配置, 研究了汽车装配线生产计划和调度的集成优化问题, 给出了该问题的混合整数规划模型。利用分枝定界算法和单纯型法求得问题的粗生产计划。通过将模拟退火算法和快速调度仿真相结合, 探讨了一种新的启发式算法。然后基于已求得的粗生产计划, 针对三种不同寻优组合论述了该算法的实现。将该算法应用于实际算例, 仿真结果表明该算法对求解此类问题有着很好的效果。

关键词: 模拟退火算法; 快速调度仿真; 生产计划与调度
中图分类号: TP311 文献标识码: A 文章编号: 1673- 629X(2009)01- 0134- 05

Approaches to Integrated Optimization of Production Planning and Scheduling on Automobile Assembly Lines

CHEN Lin, YAN Hong-sen, LIU Tong, LIU Xiao-ling
(Ministry of Education Key Laboratory of Measurement and Control of Complex Systems of Engineering, Southeast University, Nanjing 210096, China)

Abstract: In order to advance the production efficiency of automobile assembly lines, and optimize the allocation of resources, researched into the integrated optimization problem of production planning and scheduling on automobile assembly lines, and presented the mixed integer programming model of this problem. By using the branch- and- bound algorithm and simplex method got the rough production plan of this problem. A heuristic algorithm was inquired into by combining simulated annealing algorithm with quick schedule simulation. Then on the basis of the obtained rough production plan, the implementing of this algorithm was presented according to three different optimizing search combinations. Finally, the algorithm was applied to the practical examples. Simulations show that this algorithm can solve the problem effectively.

Key words: simulated annealing algorithm; quick schedule simulation; production planning and scheduling

0 引 言

在敏捷制造环境中, 要实现敏捷响应用户需求而不增加生产成本, 就需要生产成本与批量无关, 即要求汽车装配线能够装配任意混批甚至单辆汽车。由于汽车的品种、批量、混批经常会发生变动, 故如何编制和优化汽车装配线的生产计划与调度, 控制整个装配工位的装配节奏, 使其负荷均衡并保持与大规模生产线一样的资源利用率, 就成了能否实现汽车装配线敏捷制造的关键[1]。

为解决此整体优化问题, 提出一种解决方法: 将汽车装配线简化为一个 Flow shop 问题, 并建立其混合整数规划模型, 由此求得使各装配工位的资源利用率和准备成本达到整体优化并尽可能满足需求的粗生产计划。然后考虑装配线的细节, 建立求解生产计划与调度整体优化问题的数学模型, 利用模拟退火(Simulated Annealing)算法与快速调度仿真相结合的方法使生产计划与调度达到整体优化。

模拟退火算法[2](简称 SA 算法)是源于对固体退火过程的模拟, 采用 Metropolis 接受准则, 用一组称为冷却进度表的参数控制算法进程的一种解大规模组合优化问题的有效近似算法。特点是能够跳出问题的局部最优解, 以较高的精度收敛于整体最优解。

由于 SA 算法的搜索过程是随机的, 且当退火的控制参数值较大时可以接收部分恶化解, 故考虑在算法中增加一个记忆器[3,4], 以记住搜索过程中遇到过的最好结果。退火过程结束时, 将所得的最终解与记忆器中的解比较并取较优者作为最后结果。由此, 对

收稿日期: 2008- 05- 17
基金项目: 国家 863 计划资助项目(2007AA04Z112); 国家自然科学基金资助项目(60574062)
作者简介: 陈 琳(1983-), 女, 江苏镇江人, 硕士研究生, 主要研究领域为计划与调度、计算机集成制造等; 严洪森, 教授, 博士生导师, 主要研究领域为生产计划与调度、知识化制造、并行工程等。

算法改进得到带有记忆的 SA 算法。同时, 在当前解搜索的邻域中, 引进 Tabu 搜索算法机制[5], 即通过嵌入 Tabu 搜索算法的记忆过程将搜索过的解保存在记忆近期操作的 Tabu 表中, 找到一个较优的解, 然后在新解的邻域中, 继续搜索较好的解。这样设计可以避免很多无效的搜索, 达到快速寻优的目的。

1　粗生产计划

建立求解最优粗生产计划的混合整数规划模型[1,6]如下:

$$\min J = \sum_{i=1}^{N}[a_i^+ (x_i - d_i)^+ + a_i^- (d_i - x_i)^+] + \sum_{j=1}^{M}\sum_{i=1}^{N} b_{ij}\mathrm{sgn}(x_i) + \sum_{j=1}^{M} c_j\tau_j \quad (1)$$

$$\text{s.t.} \quad \sum_{i=1}^{N} t_{ij}x_i + \sum_{i=1}^{N}\Delta t_{ij}\mathrm{sgn}(x_i) + \tau_j = \beta_j \quad (2)$$

$(x_i \geqslant 0$ 且为整数, $\tau_j \geqslant 0$, $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, M)$

其中, N 为计划任务所要装配的汽车种类数; M 为汽车装配线上的工位数; x_i 为计划区间内装配第 i 种汽车的产量; d_i 为计划区间内对第 i 种汽车的需求; τ_j、β_j 分别为计划区间内第 j 个装配工位的空闲时间和可用时间; a_i^+ 为第 i 种汽车超产的存储及占用流动资金的单位成本; a_i^- 为第 i 种汽车欠产而违约受罚的单位成本; c_j 为与资源闲置有关的成本系数; t_{ij} 为第 j 个装配工位装配第 i 种汽车所需要的时间; $(c)^+$ 为 $\max(0, c)$, 即取正数; $\mathrm{sgn}(x_i)$ 为符号函数, 当 $x_i > 0$ 时, $\mathrm{sgn}(x_i)$ 取 1, 否则取 0; b_{ij}、Δt_{ij} 分别为第 i 种汽车在第 j 个装配工位上的准备成本和准备时间。

式(1)为混合整数非线性规划模型, 因为该模型对某些点的导数不存在, 不便用一般的非线性规划方法求解, 故引进一些变量和约束将其转化为混合整数线性规划模型进行求解。

改进模型如下:

$$\min J = \sum_{i=1}^{N}[a_i^+ \Delta^+ x_i + a_i^- \Delta^- x_i] + \sum_{j=1}^{M}\sum_{i=1}^{N} b_{ij}y_i + \sum_{j=1}^{M} c_j\tau_j \quad (3)$$

$$\text{s.t.} \quad \sum_{i=1}^{N} t_{ij}x_i + \sum_{i=1}^{N}\Delta t_{ij}y_i + \tau_j = \beta_j \quad (4)$$

$$By_i \geqslant x_i \quad (5)$$

$$x_i - \Delta^+ x_i + \Delta^- x_i = d_i \quad (6)$$

$(x_i \geqslant 0$ 且为整数, $\tau_j \geqslant 0$, $\Delta^+ x_i \geqslant 0$, $\Delta^- x_i \geqslant 0$, $y_i \in (0, 1)$, $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, M)$

其中, y_i 为 0～1 的整数变量, 当 $x_i > 0$ 时为 1, 否则为 0; B 为一个大的正数。

式(3)～(6)的混合整数线性规划模型采用分枝定界法或单纯形法求解[4], 得到使各装配工位的资源利用率和准备成本达到整体优化并尽可能满足产品需求的粗生产计划。

粗生产计划求解方法介绍如下(程序框图如图 1 所示):

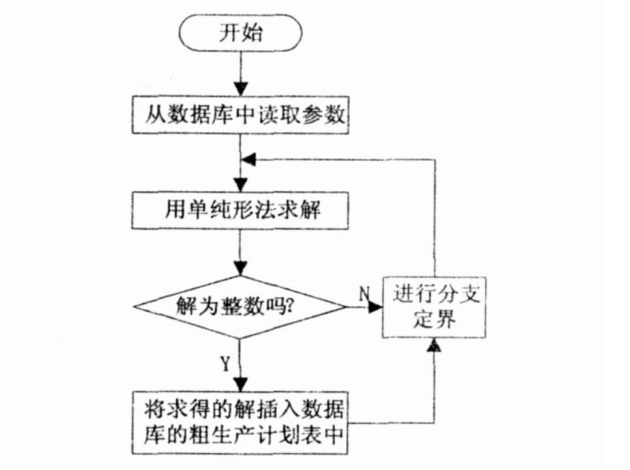

图 1　粗生产计划求解程序框图

在这里直接使用 ODBC API 编制应用程序, 实现数据的读取、插入以及修改功能。根据平滑定单给出的需求计划 $D[N]$(需求的汽车种类为 N, 需求分别为 $D[0], D[1], \cdots, D[N-1]$), 通过单纯形法求得一个最优值, 但这个值不一定就是最优的整数解, 所以需要对获得的最优解判断其是否为整数解。若为整数解, 则将其作为粗生产计划; 否则对其进行分枝定界:

第一步: 分枝, 假设原模型问题为问题 A, 任选其中的一个非整数解 $X[i]$, 其值为 $D[i]$, 如 $D[i] = 17.5$, 构造两个约束条件:

①$D[i] \leqslant 17$; ②$D[i] \geqslant 18$

将这两个约束分别加入到原问题 A 上, 则分解为两个子问题 A_1 和 A_2(即两支, 左右分枝), 左分枝增加约束条件 $D[i] \leqslant 17$ 得到问题 A_1, 右分枝增加约束条件 $D[i] \geqslant 18$ 得到问题 A_2, 求此两个后继子问题的解。

定界, 以每个后继问题为一分枝表明求解的结果, 与其它问题的解的结果相比较, 找出性能指标最好的作为新的上界。

第二步: 比较与剪枝, 各分枝中的性能指标值若有大于上界的, 则剪掉此枝, 否则重复第一步骤。一直到求得整数解为止。

分枝定界法求得的粗生产计划是最优的整数解, 但随着问题规模的增大, 分枝会越来越复杂, 其个数以 2 的指数增加, 最多的分枝节点个数为 2^N。这样问题的复杂度也随之增大, 求解的速度明显下降, 故当问题

复杂度大时(在装配线上指当需求的汽车种类超过 10 时),直接采用单纯形法求解,然后对求得的解取整。

2 生产计划与调度集成优化的实现

2.1 数学模型

式(3)~(6)的模型中忽略了装配线的细节,由其获得的粗生产计划可作为后续生产计划与调度整体优化问题迭代求解的初始计划,以加快问题的求解速度。另外,考虑细节的装配线调度往往是一个非结构化问题,很难用解析的方法求解,较为可行的办法是使用基于可变时间流的快速调度仿真[6]。最后,利用 SA 算法搜索来解决最优计划与调度的选择问题。

为节省汽车装配准备时间,同种汽车集中装配,不分割成多个装配任务,在调度中仅占用一个排序位置。故求解汽车装配车间生产计划与调度整体优化问题的数学模型[1,6]可描述如下:

$$\min_{x,s} G(x,s) = \min_{x,s}\Bigg\{\sum_{i=1}^{N}[a^{+}_{\mu(s,i)}(x_{\mu(s,i)} - d_{\mu(s,i)})^{+} + a^{-}_{\mu(s,i)}(d_{\mu(s,i)} - x_{\mu(s,i)})^{+}] + \sum_{j=1}^{M}\sum_{i=1}^{N} b_{\mu(s,i)j}\operatorname{sgn}(x_{\mu(s,i)}) + \sum_{j=1}^{M} c_j\tau_j + rf(x,s) + q\sum_{j=1}^{M}[(\beta_j - \tau_j) - (1/M)\sum_{j=1}^{M}(\beta_j - \tau_j)]^2\Bigg\} \tag{7}$$

$$\text{s.t.}\quad \sum_{i=1}^{N} t_{\mu(s,i)j}x_{\mu(s,i)} + \sum_{i=1}^{N}\Delta t_{\mu(s,i)j}\operatorname{sgn}(x_{\mu(s,i)}) + \tau_j = \beta_j \tag{8}$$

($x_{\mu(s,i)} \geqslant 0$ 且为整数, $\tau_j \geqslant 0$, $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, M$)

其中,$\mu(s,i)$ 为 N 种汽车或其中部分通过装配线的顺序(调度) s 中的第 i 个位置所对应的汽车种类;$\boldsymbol{x} = (x_{\mu(s,1)}, x_{\mu(s,2)}, \cdots, x_{\mu(s,N)})$ 为生产计划向量;$f(x,s)$ 为汽车装配任务全部完成的时间;r 为任务完成时间权系数;q 为各装配工位负荷均衡权系数。

2.2 算法实现

文中提出了基于生产计划和调度不同寻优组合的三种 SA 算法。具体设计思路如下:

(1)嵌入式 SA 算法(ESAA)。

基本思想是:从初始生产计划出发,在计划层用 SA 算法进行搜索寻找最好的计划,同时对计划层 Markov 链中随机产生的相邻计划用另一个 SA 算法搜索经过快速调度仿真计算具有最好性能指标的调度,直至生产计划与调度同时达到优化。

(2)交替式 SA 算法(ASAA)。

基本思想是:①从初始生产计划出发寻找一个可行计划与调度;②给定调度,用 SA 算法寻找最好的计划;③反过来给定计划,再用另一个 SA 算法搜索最好的调度;④交替使用②、③两步直至找到最好的计划与调度。由于分别对计划与调度进行 SA 算法搜索,在此称这种方法为交替式 SA 算法。

(3)串行式 SA 算法(SSAA)。

基本思想是:①从初始生产计划出发寻找一个可行的计划;②从可行计划开始,使用 SA 算法寻找最好的计划;③对此最好的计划,使用另一 SA 算法寻找最好的调度。由于对计划和调度依次使用 SA 算法,故称为串行式 SA 算法。

下面以嵌入式 SA 算法为例对算法流程进行详细分析:

算法 1:生产计划与调度集成优化问题嵌入式 SA 算法。

Step1:初始化

(1)读取相关参数(包括平滑定单,粗生产计划,工位装配时间、准备时间及准备成本,工位故障等)。

(2)读取冷却进度表参数,包括生产计划与调度 SA 算法控制参数 P_c_0, S_c_0;生产计划及调度 SA 算法控制参数衰减率 P_α, S_α;生产计划与调度 SA 算法的 Markov 链的最大长度 PM_max, SM_max;计划表长 PT_size,调度表长 ST_size。

(3)令 PN_len = 1, PT_list = $\{\Phi\}$, $P\_best = \{\Phi\}$, $S\_best = \{\Phi\}$, $P\_c_1 = P\_c_0$, G_best = M_big。

Step2:搜索初始可行计划与调度

(1)设置初始生产计划 $p_0 = x$。

(2)当前生产计划 $p^* = p_0$,调用算法 2(flag)。

(3)若 flag = 1,则 $P\_best \leftarrow p_0$, $S\_best \leftarrow s^{**}$, $G\_best \leftarrow G(p^*, s^{**})$,转到 Step3。

(4)若 flag = 0,则取 p_0 的某一相邻生产计划 p,并令 $p_0 \leftarrow p$,转到 Step2(2)。

Step3:搜索最好计划与调度

(1)PM_len = 0,清空计划表 PT_list。

(2)随机取 p^* 相邻计划中的一个可行计划 p,若 p 不在 PT_list 中,则调用算法 2(flag)。若 flag = 0,则转到 Step3(2)。否则计算 $\Delta G = G(p, s^{**}) - G(p^*, s^{**})$, PM_len ← PM_len + 1,若 $\exp(-\Delta G/P\_c_{PN\_len}) \geqslant \delta$ 或 $\Delta G < 0$,其中 $\delta = \text{random}[0,1]$,则生产计划由 p^* 转移到 p,即 $p^* \leftarrow p$, $G(p^*, s^{**}) \leftarrow G(p, s^{**})$。否则转到 Step3(2)。

(3)若 $G\_best > G(p^*, s^{**})$,则 $P\_best \leftarrow p^*$, $S\_best \leftarrow s^{**}$, $G\_best \leftarrow G(p^*, s^{**})$。

(4)若计划表已满,则删除最老的计划,p 加到

PT_ list 顶部。

(5) 若 PM_ len < PM_ max, 则转到 Step3(2)。

(6) 若控制参数 $P\_c_{PN\_len} < P\_c_f$, 则转到 Step4。否则 PN_ len ← PN_ len + 1, $P\_c_{PN\_len} \leftarrow P\_\alpha \cdot P\_c_{PN\_len-1}$, 转到 Step3(1)。

Step4: 输出结果

输出 P_ best, S_ best, G_ best, 各装配工位的利用率和装配完成时间。

其中, M_ big 为一个大数; p_0 为初始生产计划; x 为粗生产计划; p^* 为当前 Markov 链迭代后所接受的解; p 为当前搜索的新解; s^{**} 为相应计划的最好调度; PM_ len 为当前计划 SA 算法的 Markov 链中的变换个数; PN_ len 为当前计划 SA 算法的 Markov 链的迭代次数; $G(a, b)$ 为对应计划 a 与调度 b 的性能指标; P_ best 为经过 SA 算法搜索到目前为止最好的计划; S_ best 为经过 SA 算法搜索到目前为止最好的调度; G_ best 为到目前为止最好的生产计划与调度对应的最好性能指标; flag 为生产计划可行与否的标识, 可行则为 1, 否则为 0。

算法 2: 基于 SA 算法的调度优化。

Step1: 初始化

令 SN_ len = 1, 调度表 ST_ list = { Φ}。

Step2: 寻找初始调度

(1) 对给定的当前生产计划 p, 按最早交付期和最少批量优先规则, 确定初始调度 s_0。

(2) 对当前的初始调度 s_0, 经过快速调度仿真, 若不可行, 则 flag = 0, 且取其相邻调度 s 作为初始调度, 即 $s_0 \leftarrow s$, 转到 Step2(2)。否则 flag = 1, $s^* \leftarrow s_0$, $s^{**} \leftarrow s_0$, $G(p, s^*) \leftarrow G(p, s_0)$, $G(p, s^{**}) \leftarrow G(p, s_0)$ 转到 Step3。若对当前计划 p 的初始调度的所有相邻调度搜索完毕, 则转到 Step4。

Step3: 寻找最好的调度

(1) SM_ len = 0, 清空调度表 ST_ list。

(2) 随机取 s^* 相邻调度中的某一可行调度 s, 若 s 在 ST_ list 中, 则回到 Step3(2)。否则调用快速调度仿真, 计算 $\Delta G = G(p, s) - G(p, s^*)$, SM_ len ← SM_ len + 1, 若 $\Delta G < 0$ 或 $\exp(-\Delta G / P\_c_{PN\_len}) \geqslant \delta$, 则调度由 s^* 转移到 s, 即 $s^* \leftarrow s$, $G(p, s^*) \leftarrow G(p, s)$。若调度表已满, 则删除最老的调度, s 加到 ST_ list 顶部。

(3) 计算 $\Delta G = G(p, s^{**}) - G(p, s^*)$, 如果 $\Delta G < 0$, 则 $s^{**} \leftarrow s^*$。

(4) 若 SM_ len < SM_ max, 则转到 Step3(2)。

(5) 若控制参数 $S\_c_{SN\_len} < S\_c_f$, 则转到 Step4。否则 SN_ len ← SN_ len + 1, $S\_c_{SN\_len} \leftarrow S\_\alpha \cdot S\_c_{SN\_len-1}$, 转到 Step3(1)。

Step4: 返回

其中, s^* 为当前调度搜索 Markov 链迭代后接受的解(旧解); s 为当前调度搜索的新解; s^{**} 是针对当前计划 p 搜索到的最好调度; SM_ len 为当前调度的 SA 算法的 Markov 链中的变换个数; SN_ len 为当前调度的 SA 算法 Markov 链的迭代次数。

3　算例研究

文中提出的嵌入式模拟退火算法(ESAA), 交替式模拟退火算法(ASAA)和串行式模拟退火算法(SSAA)均用 VC+ + 6.0 编成软件。算法中的参数读取、修改等通过 MFC 访问 ODBC 实现[7]。在下面的例子中, 所有这三种算法软件都在内存 1G, Pentium (R) 4 2.80GHz, Windows XP 环境中运行。

算例 1: 5 种车型汽车装配线有缓冲区有准备成本确定情况(由于数据过多, 在这里仅列出主要参数)。

假定算例中汽车装配线同南京某汽车总装厂的一条装配线一样都有 33 个工位, 并且装配线的每个工位的缓冲区都有 3 个存放位置(一个存放位置只能放一辆汽车)。

算例仿真参数如下:

(1) N = 5(即 5 种车型), 需求为{4, 5, 14, 20, 5}, 粗生产计划为{4, 5, 36, 20, 5};

(2) 设第 i 种车在第 j 个工位上的装配时间为 t_{ij}, 准备时间为 dt_{ij}, 准备成本为 c_{ij};

(3) 产品成本参数如表 1 所示;

(4) 冷却进度表参数:

P_c_0 = 8500, S_c_0 = 99, P_c_f = 100, S_c_f = 10, P_α = 0.9, S_α = 0.8, PM_ max = 30, SM_ max = 4, PT_ size = 5, ST_ size = 2。

表 1　成本参数表

主产品号	超产惩罚系数	欠产惩罚系数
1	1200	5000
2	1000	4000
3	1000	4000
4	1450	5600
5	1456	6000

根据以上数据在 PC 机上分别运行 SSAA、ASAA、ESAA。仿真结果如表 2 所示。

算例 2: 5 种、10 种、30 种汽车装配线有缓冲区有准备成本确定情况。

5 种车型参数设置同算例 1, 10 种、30 种车型参数过多, 这里省略不表。为简便起见, 仅以 ASAA 为例, 观察问题规模变化时的仿真结果, 如表 3 所示。

表 2 算例 1 仿真结果

	SSAA	ASAA	ESAA
P_ best	{0, 0, 31, 15, 1}	{1, 2, 33, 14, 1}	{2, 5, 26, 14, 5}
S_ best	{1, 2, 5, 4, 3}	{1, 5, 3, 2, 4}	{1, 2, 5, 3, 4}
G_ best	216605	208926	180540
性能指标提高率	2.81%	20.27%	31.10%
仿真时间	6.203s	28.344s	431.39s

表 3 算例 2 仿真结果

ASAA	5 种车	10 种车	30 种车
P_ best	{1, 2, 33, 14, 1}	{5, 9, 6, 8, 4, 10, 2, 4, 1, 8}	{3, 3, 2, 6, 2, 2, 4, 3, 2, 2, 2, 2, 4, 4, 2, 3, 2, 3, 2, 2, 1, 2, 5, 2, 3, 3, 2, 5, 3, 2}
S_ best	{1, 5, 3, 2, 4}	{9, 7, 5, 8, 1, 3, 4, 10, 6, 2}	{5, 20, 5, 19, 9, 13, 18, 12, 15, 17, 28, 14, 22, 24, 27, 30, 11, 2, 8, 16, 21, 25, 26, 29, 7, 1, 3, 23, 10, 4}
G_ best	208926	280650	176160
性能指标提高率	20.27%	2.54%	36.69%
仿真时间	28.344s	122.531s	2478.22s

比较算例 1 与算例 2 的仿真结果, 得出结论:

(1) 与传统方法相比, 文中方法的优点是将解析方法、模拟退火方法和快速调度仿真有机地结合在一起, 有效地解决了汽车装配线生产计划与调度的集成优化问题, 并保证至少有一个可行解。

(2) ESAA 的问题求解速度最慢, 但获得的性能指标往往最好。SSAA 的问题求解速度最快, 获得的性能指标却往往最差。ASAA 介于两者之间。

(3) 随着问题规模的增大, 这三种算法求解问题所需要的时间也随之增长, 在可以接受的时间内, ESAA 比较适合于求解小规模问题, ASAA 较适合于求解中规模问题, 而 SSAA 则适合于求解大规模问题。

4 结束语

研究了汽车装配线上生产计划与调度的寻优方法, 成功设计并实现了基于模拟退火和快速调度仿真相结合的算法。通过实际算例仿真, 证明该算法可以极大地提高生产效率, 优化资源配置, 并能应对不断变化的实际生产情况, 在较短的时间内得到比较好的调度方案。另外, 使用者针对问题规模的大小可以选择基于不同寻优组合的算法, 这样的设计也大大增强了该方法的实用性和有效性。

参考文献:

[1] Yan Hongsen, Xia Qifeng, Zhu Minru, et al. Integrated production planning and scheduling on automobile assembly lines [J]. IIE Transactions, 2003, 35: 711- 725.

[2] Michalewicz Z, Fogel D B. 如何求解问题——现代启发式方法[M]. 曹宏庆等译. 北京: 中国水利水电出版社, 2003.

[3] 潘全科, 段俊华, 赵清理, 等. 解决车间调度问题的改进模拟退火算法[J]. 机械科学与技术, 2007, 26(1): 112- 114.

[4] 赵良辉, 邓飞其. 解决 Job Shop 调度问题的模拟退火算法改进[J]. 计算机工程, 2006, 32(21): 38- 40.

[5] 刘霞玲. 汽车装配件制造执行系统的研究与开发[D]. 南京: 东南大学, 2001.

[6] 严洪森, 夏琦峰, 朱昊如, 等. 汽车装配车间生产计划与调度的同时优化方法[J]. 自动化学报, 2002, 28(6): 911- 919.

[7] 任 哲. MFC Windows 应用程序设计[M]. 第 2 版. 北京: 清华大学出版社, 2007.

(上接第 133 页)

远小于 $m \times R$。当 $R' << R$ 时(设 $\sum P$ 是包含 P 中字符的字符表, $R' = |\sum P|$), 这一现象将更为突出。

从如上(1)和(2)可以看出, 快速串匹配算法在时间复杂度上与 K. M. P. 算法相当, 但实际的比较次数要小于后者。在空间复杂度上, 前者要大于后者。而且这些差别取决于具体的 T, P。事实上, 快速串匹配算法采用二维状态转换表来代替 K. M. P. 算法中的一维失败链数组的做法就是通过牺牲部分的空间复杂度来换取更高的执行速度。

4 结束语

文中基于有限自动机理论, 提出了一种快速串匹配算法。理论分析与实验结果均表明, 在正文串比较长, 模式串中局部匹配失败时的失败链反馈较多的情况下, 该算法在速度上明显优于 K. M. P. 算法。算法缺点是对模式预处理后得到的自动机需要较多的存储空间。尽管如此, 该算法在速度上的优势表明它在理论和应用上都有一定的价值。

参考文献:

[1] 徐孝凯. 数据结构实用教程[M]. 北京: 清华大学出版社, 2006.

[2] 王建国, 郑家恒. BM 串匹配算法的一个改进算法[J]. 计算机工程与科学, 2007, 29(5): 94- 95.

[3] 蔡晓妍, 戴冠中, 杨黎斌. 改进的多模式字符串匹配算法[J]. 计算机应用, 2007, 27(6): 1415- 1417.

[4] Knuth D E, Pratt V R, Morris J H. Fast pattern matching in strings[J]. SIAM J. Comput., 1977, 6(1): 323- 350.

[5] 李 钢, 吴燎原, 张仁斌, 等. 基于有限自动机的模式匹配算法及其应用研究[J]. 系统仿真学报, 2007, 19(12): 2772- 2775.

[6] Crochemore M, Rytter W. Text algorithms[M]. Oxfordshire: Oxford University Press, 1994.