第33卷 第2期 北京交通大学学报 Vol.33 No.2
2009年4月 JOURNAL OF BEIJING JIAOTONG UNIVERSITY Apr. 2009

文章编号: 1673-0291(2009)02-0039-05

二维矩形块布局的交叉熵方法实现

吕盛坪, 陆一平, 查建中, 吕胜祥
(北京交通大学 机械与电子控制工程学院, 北京 100044)

摘 要: 给定一个小矩形块集和一个固定宽度而高度变化的大矩形的二维布局问题, 就是将这矩形集中的所有矩形正交布置于这个大矩形中, 并且保证矩形块之间不发生重叠, 目的就是使得布局后这个大矩形块的高度最小. 本文提出通过DROP或DROPF (DROP FILL) 的启发式解码策略与交叉熵算法相结合求解该类问题. 试验结果显示, 算法稳定有效, 较经典元启发式算法在提高空间利用率上有较大提高.

关键词: 二维矩形块布局问题; 交叉熵; DROP; DROPF
中图分类号: TP312 文献标志码: A

A Cross-Entropy Method for Two-Dimensional Packing Problem with Rectangle Pieces

LU Shengping, LU Yiping, ZHA Jianzhong, LU Shengxiang
(School of Mechanical and Electronic Control Engineering, Beijing Jiaotong University, Beijing 100044, China)

Abstract: Given a set of rectangular pieces and an object of fixed width and variable height, the two-dimensional packing problem consists of orthogonally placing all the pieces within the object, without overlapping, so that the overall height of the layout is minimized. This paper proposes a new algorithm by combining the DROP or DROPF (DROP FILL) heuristic decoder policy into the Cross-Entropy to solve the problem. The experiment results demonstrated that the algorithm is stable and fairly efficient. And it improves the occupation rate comparing with the classical meta-heuristic algorithm.

Key words: 2D-R-ODP; cross-entropy(CE); DROP; DROPF

给定一个小矩形块集和一个固定宽度而高度变化的大矩形块的二维布局问题, 就是将这矩形集合中的所有矩形正交且不重叠地布置于这个大矩形空间中, 使得布局后这个大矩形块的高度最小化. 这类问题在玻璃、木材和纸张工业, 存储和分配等实际生产中广泛存在, 所以近年来运筹学和工业领域里的研究者对其进行了大量的研究. 按照Dyckhoff关于布局问题的分类算法[1]编码为2/V/O/M. 按照改进的分类算法[2], 其属于开维问题(Open Dimension Problem, ODP)的中间类型, 现一般称其为二维矩形块布局问题(Two-Dimensional Rectangular, 2D-R-ODP).

2D-R-ODP是NP-完全问题, 一般采用随机搜索技术来求解. 例如基因遗传(GA)算法, 模拟退火算法(SA), 禁忌搜索(TS)算法, 原始演化(NE)算法和其他复合元启发式算法. 采用这些算法时一般结合其他的解码启发式算法. 下面给出求解这类问题的一些相关文献.

Jakobs[3]在1996年提出了采用BL(Bottom Left)解码算法与GA算法结合来求解2D-R-ODP. Liu and Teng[4]在1999年对BL解码算法进行了改进, 其结果显示改进的BL解码算法比Jakobs采

收稿日期: 2007-07-17
基金项目: 河北省自然科学基金资助项目(F20060003040); 北京交通大学科技基金资助项目(2006XZ011)
作者简介: 吕盛坪(1982—), 男, 湖南新邵人, 博士生. email: lvshengping@yeah.net.
陆一平(1965—), 男, 广西上林人, 副教授.

用的 BL 解码算法获得更好的结果. 2001 年, Hopper and Turbon[5] 分别采用了 BL 和 BLF 的解码算法和 SA, GA, NE 这些元启发式算法相结合做了一个总结性的试验, 各种算法对于一系列布局算例的试验结果的对比是从解的结果和获得相应结果所需计算时间来考虑的. SA 获得最好的布局结果, 但是对同一问题其所用的计算时间就更长. Leung 等人[6] 于 2003 年采用了不同过程 DP(Difference Process) 布局策略与 GA 和 SA 的复合算法相结合来求解该问题, 这种复合算法一般能获得较好的结果, 但是计算时间增加.

采用交叉熵(Cross- Entropy, CE) 元启发式算法来求组合优化问题的研究时间并不很长. 1999 年以色列教授 Rubinstein[7] 将原本用于小概率事件估计的交叉熵算法改进为用于求解优化问题, 近年来被不同的研究者成功地应用于其他组合优化问题. 例如: 货郎担问题[7], 最大割和二分问题[8], 排队论中的缓冲溢出问题[9] 和网络可靠性问题[10] 等. 受其启发, 本文作者提出采用交叉熵算法来求解 2D- R - ODP 问题, 试验结果显示算法稳定有效, 较经典元启发式算法在提高空间利用率上有较大提高.

1 交叉熵算法

采用交叉熵算法来求优化问题, 其基本思想就是将待求优化问题转化为对应的小概率事件估计问题, 然后采用交叉熵估计小概率事件的算法求取相应参数的估计值. 然后再将求得的参数估计值返回于求解原始优化问题.

假设现打算求解下面的优化问题

$$\gamma^* = \min_{x \in \chi} S(\boldsymbol{x}) \tag{1}$$

首先在集合 χ 中定义一个不同水平阶段 $\gamma(\gamma \in \mathbf{R})$ 的0- 1 函数集$\{I_{\{S(\boldsymbol{X}) \leqslant \gamma\}}\}$, 然后设$\{f(\cdot; \boldsymbol{v}), \boldsymbol{v} \in V\}$ 是定义于 χ 且以 $\boldsymbol{v}$ 为参数的概率密度函数族. 这里 $S(\boldsymbol{x})$ 是待优化函数, $\boldsymbol{v}$ 是一个实参数(向量), V 是其对应的空间集. 对于确定的实参(向量) $\boldsymbol{u}$, 我们将其转换为求解下面的小概率事件的估计问题

$$l(\gamma) = P_{\boldsymbol{u}}(S(\boldsymbol{X}) \leqslant \gamma) = E_{\boldsymbol{u}} I_{\{S(\boldsymbol{x}) \leqslant \gamma\}} \tag{2}$$

其中 $\boldsymbol{X} = \{X_1, \cdots, X_n\}$ 是概率密度限于$\{f(\cdot; \boldsymbol{v}), \boldsymbol{v} \in V\}$族用 $f(\cdot; \boldsymbol{u})$ 表示的随机向量, $E_{\boldsymbol{u}} I_{\{S(\boldsymbol{x}) \leqslant \gamma\}}$ 为 $I_{\{S(\boldsymbol{x}) \leqslant \gamma\}}$ 期望值. 这样就将这确定性的优化问题转化为相应统计问题. 根据交叉熵估计小概率事件的 Kullback-leiber 距离最短原理, 对于确定阶段 $\gamma = \gamma^*$, 对应参数 $\boldsymbol{v}^*$ 的估计量可表示如下

$$\hat{\boldsymbol{v}}^* = \arg\max_{\boldsymbol{v}} \frac{1}{N} \sum_{i=0}^{N} I_{\{S(X_i) \leqslant \gamma\}} \ln f(\boldsymbol{X}_i; \boldsymbol{v}) \tag{3}$$

现将交叉熵算法求解优化问题的算法描述如下.

算法 1(交叉熵求解优化问题的主算法):

1) 初始 $\boldsymbol{v}$ 的估计值 $\hat{\boldsymbol{v}}_0 = \boldsymbol{u}$, 置 $t = 1$, 初始化 α, ρ, N, d, 其中 α 为平滑系数, ρ 为分位数, N 为每步迭代过程中的样本容量, d 为迭代终止参数, t 为迭代步骤.

2) 根据概率密度 $f(\cdot; \boldsymbol{v}_{t-1})$) 生成样本群 $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_N$ 并计算对应目标函数 ρ 的分位数值

$$\hat{\gamma}_t = S_{(\lceil \rho N \rceil)}.$$

3) 采用同一样本群 $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_N$ 求解式(3) 更新迭式, 设其为 $\hat{\boldsymbol{v}}_t$, 然后采用 $\hat{\boldsymbol{v}}_t = \alpha \hat{\boldsymbol{v}}_t + (1 - \alpha) \hat{\boldsymbol{v}}_{t-1}$ 对其进行平滑.

4) 如果对于某个 $t \geqslant d$ 有 $\hat{\gamma}_t = \hat{\gamma}_{t-1} = \cdots \hat{\gamma}_{t-d}$, 停止迭代, 转第 5 步; 反之 $t = t + 1$, 返回第 2 步继续迭代.

5) 将最终参数返回求原优化问题.

从算法描述中可以看出采用交叉熵算法来求优化问题有以下两个主要问题要解决: ①如何通过一定的随机机制生成随机样本数据(序列, 轨迹等); ②如何在下次迭代过程中更新参数 $\boldsymbol{v}$ 而使得下次迭代能生成更优的样本.

2 2D- R- ODP 的交叉熵算法

本文主要给出 2D- R- ODP 的交叉熵算法, 2D - R- ODP 的定义在引言中已给出, 其名称来自于 W¾scher 等人[2] 给出的分类算法和命名规则.

2.1 由概率矩阵生成样本序列

为了将小矩形块集合中的矩形块布置到大的矩形中, 首先将这些小矩形块以数字进行编码, 这样随机生成的序列可以作为布局矩形块在布局过程中布置的顺序. 对小矩形块集进行编号的过程可以看作一个马尔可夫过程, 这样就可以引入马尔可夫链转移矩阵 $\boldsymbol{P}$ 来生成样本序列. 一般称 $\boldsymbol{P}$ 为概率矩阵, 因为在迭代过程中这矩阵的每个元素都是一个概率值. 如何用概率矩阵来生成序列由算法 2 给出.

算法 2(由概率矩阵生成序列算法):

1) 定义一个 n(待布局块数) 维的概率矩阵, 初始行号 $k = 1$.

2) 对 $\boldsymbol{P}$ 归一处理, 确保矩阵各行元素和为 1.

3) 然后随机生成一个随机数 r, 计算满足

$$\sum_{j=1}^{t} p_{kj} \geqslant r,$$

最小的 t. 如果 $t = n$, $X_k = t - 1$, 反之 $X_k = t$. 然后置该矩阵的第 X_k 列所有元素为 0, 以避免在同一序列中产生两个相同的数字.

4) 如果 $k=n$ 退出, 反之令 $k=k+1$, 返回第 2 步继续.

2.2　2D-R-ODP 的 DROP 和 DROPF 解码原则

由概率矩阵生成的序列来对小矩形块进行编码只能表示矩形块在布局过程中的顺序, 必须采用一定的解码原则对其编码进行解码, 这样解码后得到的高度就为对应序列的目标函数值. 本文提出采用 DROP 和 DROPF 启发式解码算法. DROP 解码主要包含 3 个移动操作: 首先将待布小矩形块从大矩形块的右边边界往下滑落, 直到其下边抵致大矩形块的底部或其他已布置好的小矩形块的上边, 第 2 个操作就是将其往左移动直到其他块阻止其继续前行或其已达到了大矩形块的左边沿. 第 3 操作是当在第 2 操作过程中待布矩形块“落空”, 将该待布矩形块的 x 坐标保持不变垂直下落, 直到碰到其他已经布局好的矩形块或大矩形块底部为止. 反复执行第 2、3 步直到将待布矩形块布置下来. DROPF 解码原则就是在 DROP 解码的基础上考虑用矩形块来填布局“空穴”. 由 2D-R-ODP 是可以旋转的特点, 对那些原始宽度大于长度的矩形块, 我们首先将其旋转 90°.

DROP 解码的时间复杂度在最坏的情况下为 $O(n^2)$. 每块待布矩形块 r_i 往前最多能移动 $i-1$ 步, 因为每次移动最多受已布置好的 $i-1$ 块限制, 所以采用 DROP 解码的待布矩形块 r_i 的时间复杂度为 $O(i)$, 而整个复杂度为 $O(n^2)$. 但是在大多数情况下, 矩形块在“受阻”时就停止前进了. 不像 BL 解码原则, DROP 解码没必要去考虑所有由已经布局好的矩形块形成的之字形结构放置一待布块所必需的高度. 对于 DROPF 解码, 我们首先搜索已布局模式所形成的“空穴”, 如果有任意一“空穴”能容下当前待布块(可以旋转 90°), 将该块放入该“空穴”并更新该“空穴”. 如果所有的“空穴”不能容下当前待布块, 再考虑 DROP 解码原则来布局. 所以最坏情况下布置块 r_i 时间复杂度为 $O(i^2)$, 而整个时间复杂度为 $O(n^3)$.

2.3　采用交叉熵求解 2D-R-ODP 参数的更新

在 2.1 节, 通过在概率密度中引入概率矩阵 $\boldsymbol{P}$ 解决了交叉熵求解优化问题的第一问题. 在 $\boldsymbol{P}$ 每行和为 1 的条件下, 根据拉格朗日乘子法可以得到参数 $\boldsymbol{P}$ 的各元素的迭代更新公式(证明略)

$$\hat{p}_{t,ij}=\frac{\sum_{k=1}^{N} I_{\{S(\boldsymbol{X}_k)\leqslant\hat{\gamma}_t\}} I_{\{X_{ki}=j\}}}{\sum_{k=1}^{N} I_{\{S(\boldsymbol{X}_k)\leqslant\hat{\gamma}_t\}}},\qquad(4)$$

其中 $\boldsymbol{X}_k$ 是用来表示由参数 $\hat{\boldsymbol{P}}_{t-1}$ 生成的序列, $S(\boldsymbol{X}_k)$ 为对 $\boldsymbol{X}_k$ 进行解码后得到的目标函数值, 这样就解决了交叉熵求解优化问题更新参数的问题. 如果是, 就可以采用算法 1 来求解 2D-R-ODP. 这里只要将算法描述中的参数 $\boldsymbol{v}$ 替换为概率矩阵 $\boldsymbol{P}$ 就可以了.

3　实验结果

为了说明交叉熵算法求解 2D-R-ODP 的效果, 本文给出对 12 个测试算例进行反复试验的结果. 算例来自于文献[5]中的表 13~15. 其中小矩形块数为 49~200 不等(表 1).

表 1　测试算例

Tab.1　Test problems

问题类型	矩形块块数	最优高度	大矩形尺寸	包含算例
C1	49	60	60×60	Case 1, 2, 3
C2	73	90	60×90	Case 4, 5, 6
C3	97	120	80×120	Case 7, 8, 9
C4	196/197	240	160×240	Case 10, 11, 12

实验环境为 AMD Sempron (tm) CPU 1.64 GHz, 内存 768 MB. 算法采用 C++ 编程模拟实现. 取平滑系数 $\alpha=0.7$, 置前 3 类问题的分位数 $\rho=0.1$, 迭代终止参数 $d=30$. 对于第 4 类问题(C4), 置 $\rho=0.075$, $d=25$. 表 2 给出了交叉熵求解算例的模拟试验结果. 对于算例 1 到 9 为 10 次模拟相关结果, 其他为 5 次模拟的相关结果. 样本容量直接影响到算法结果好坏和算法运行时间, 本文通过初始试验确定试验样本容量大小. 从表 2 的运行时间列可看出, 对同一算例, 在样本容量相同的情况下, 采用 DROPF 解码的实验运行时间相对 DROP 解码而言并没有按时间复杂度分析那样以算例规模的倍数增加, 甚至有些运行时间还减少. 主要原因是: 一方面对比表 2 的最后一列, 特别是规模较大时, 可以看出 DROPF 解码的循环迭代次数较 DROP 减少, 这样就减少了循环时间; 另外在将一待布块放置于已有“空穴”后就不考虑采用 DROP 解码了, 减少了其解码时间. 对于第一类问题 3 个算例, 取算例 1、2、3 样本容量 N 分别为 2 000, 3 000, 4 000, 可以看出随着样本容量增加, 得到平均和最优结果就越接近, 这说明在每次迭代时, “考察”更多的样本, 其结果就越稳定.

表 2 中每列的上面行和下面行分别对应为采用 DROP 解码和 DROPF 解码的试验结果统计(最优/平均). 列表示在 10(5) 次试验中得到的最优布局的边料损失比率和平均边料损失比率.

表2 采用交叉熵算法求的实验结果

Tab. 2 Results of CE for test cases

算例	最优/平均	运行时间/s	样本容量	平均迭代次数
1	0.05000/0.05667	122.8	2 000	141.5
	0.03333/0.03333	136.8		142.5
2	0.03333/0.05500	185.8	3 000	143.0
	0.01667/0.03167	191.2		129.2
3	0.03333/0.04500	258.5	4 000	129.2
	0.01667/0.02667	229.2		106.4
4	0.03333/0.04000	231.9	2 000	183.9
	0.02222/0.02667	252.6		159.0
5	0.04444/0.05444	301.5	3 000	155.3
	0.02222/0.02444	368.6		154.0
6	0.03333/0.06000	238.3	2 000	176.9
	0.01111/0.02667	238.7		151.5
7	0.05000/0.05417	606.7	3 000	210.0
	0.02500/0.02833	446.0	2 000	194.6
8	0.04167/0.04833	526.4	3 000	182.9
	0.01639/0.02417	387.1	2 000	162.5
9	0.04167/0.05083	618.0	3 000	215.0
	0.01639/0.02583	456.0	2 000	187.1
10	0.07917/0.08833	6 142.2	2 000	321.0
	0.03333/0.03833	9 416.6		207.2
11	0.06500/0.06666	4 967.8	2 000	253.4
	0.03333/0.03166	6 156.6		139.8
12	0.06666/0.07333	6 028.8	2 000	302.4
	0.02916/0.03250	6 917.8		163.8

因为计算机性能不同, 与其他论文中给出得算法在计算时间上无法进行对比. 表3给出采用交叉熵算法分别与DROP和DROPF解码结合求解这些算例的边料损失与文献[5]中采用的元启发式算(GA, SA, NE)分别与BL和BLF解码相结合得到的边料损失进行对比. 表3中CE+DROP和CE+DROPF行的结果为表1中给出的同一类型中3个算例的平均边料损失的平均, 与文献[5]统计方法同. 可以看出CE+DROP较文献[5]给出的GA, SA, NE与BL相结合的方法在空间利用率上有明显的提高. CE+DROPF较文献[5]给出的GA, SA, NE与BLF相结合的方法在空间利用率上也有一定的提高.

表3 算例的交叉熵算法与文献[5]算法边料损失对比

Tab. 3 Comparison of trim loss of CE method with methods from literature [5] for test cases

算法	C1	C2	C3	C4
GA+BL	0.09000	0.11000	0.15000	0.21000
NE+BL	0.08000	0.11000	0.13000	0.19000
SA+BL	0.06000	0.06000	0.07000	0.13000
CE+DROP	0.05333	0.05148	0.05111	0.07607
GA+BLF	0.03000	0.04000	0.04000	0.05000
NE+BLF	0.04000	0.04000	0.04000	0.05000
SA+BLF	0.03000	0.03000	0.03000	0.04000
CE+DROPF	0.03000	0.02593	0.02611	0.03416

图1、图2分别给出了采用CE+DROP和CE+DROPF求解算例8的最优布局图.

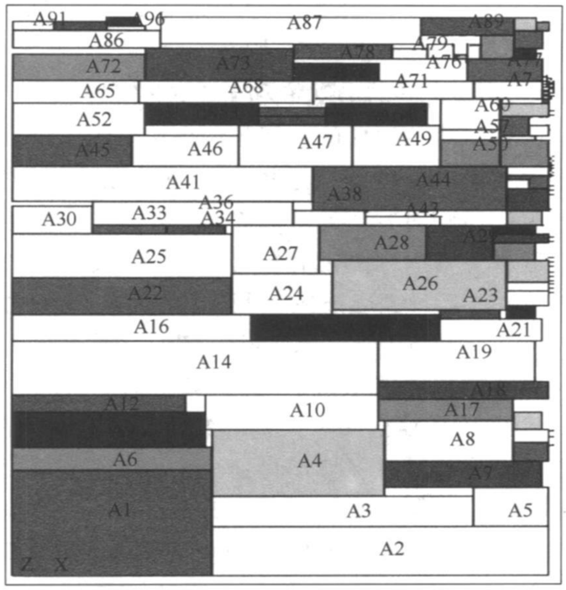

图1 **CE+DROP**得到算例**8**最优布局图

Fig. 1 Best layout of problem 8 with CE+DROP

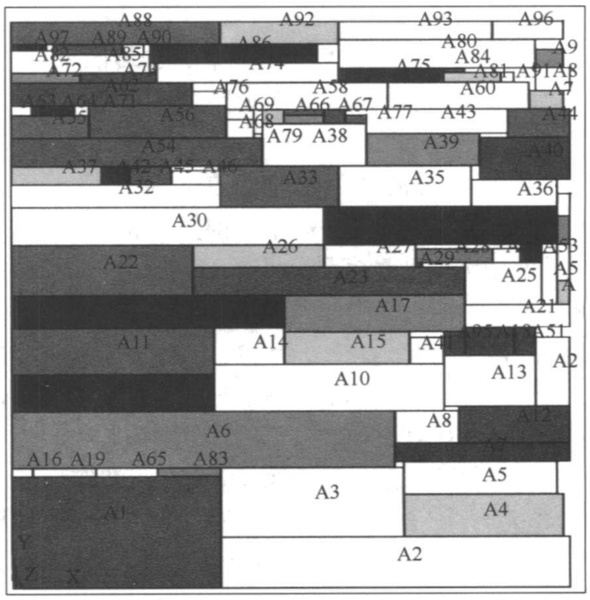

图2 **CE+DROPF**得到算例**8**最优布局图

Fig. 2 Best layout of case 8 with CE+DROPF

4 结语

通过采用交叉熵算法与DROP和DROPF的解码原则相结合来求解2D-R-DROP问题, 试验验证交叉熵法在求解该类布局问题较经典元启发式算法在提高空间利用率上有较大改进. 对于C4类型的较大规模的布局问题, 其时间也不超过3 h. 且其有并行性, 可以通过并行计算提高其运算效率. 所以交叉熵算法为求解布局问题提供了一种新的参考算法.

参考文献:

[1] Dycyhoff. A Typology of Cutting and Packing Problems [J]. European Journal of Operational Research, 1990, 44: 145- 159.

[2] W¾scher G, Hauβner H, Schumann H. An Improved Typology of Cutting and Packing Problems[J]. European Journal of Operattional Research, 2006, 183: 33- 51.

[3] Jakobs S. On Genetic Algorithms for the Packing of Polygons[J]. European Journal of Operattional Research, 1996, 88: 165- 181.

[4] Liu D, Teng H. An Improved BL-Algorithm for Genetic Algorithm of the Orthogonal Packing of Rectangles[J]. European Journal of Oprational Research, 1999, 112: 413- 420.

[5] Hopper E, Turton B C H. An Empirical Investigation of Meta- Heuristic and Heuristic Algorithm for A 2D Packing Problem[J]. European Journal of Operational Research, 2001, 128: 34- 57.

[6] Leung T W, Chan C K, Troutt M D. Application of A Mixed Simulated Annealing- Algorithm Heuristic for the Two- Dimensional Orthogonal Packing Problem[J]. European Journal of Operational Research, 2003, 145: 530- 542.

[7] Rubinstein R Y. Cross- Entropy Method for Combinatorial and Continuous Optimization [J]. Methodology and Computing in Applied Probability, 1999, 2: 127- 190.

[8] Rubinstein R Y. Cross- Entropy Method and Rare- Events for Maximal Cut and Bipartition Problems [J]. ACM Transactions on Modeling and Computer Simulation, 2002, 12: 27- 53.

[9] DeBoer P T, Kroese D P, Rubinstein R Y. A Fast Cross- Entropy Method for Estimating Buffer Overflows in Queueing[J]. Networks Management Science, 2004, 50: 883- 895.

[10] Hui K P, Bean N, Kraetzl M. Cross Entropy Method for Networks Estimation[J]. Annals of Operations Research, 2005, 134: 101- 118.

·简讯·

北京轨道交通运行控制系统国家工程研究中心有限公司在北京交通大学揭牌

2009 年 1 月 19 日, 北京轨道交通运行控制系统国家工程研究中心有限公司揭牌仪式在北京交通大校隆重举行, 同时召开了第一届董事会. 北京交通大校校长宁滨、国家发改委创新能力建设处处长阮高峰、教育部高新处处长董维国、铁道部运电处处长穆建成等出席. 会议由北京交通大学副校长李学伟主持.

宁滨校长表示: 运行控制系统是轨道交通的“大脑”, 其核心技术必须自主研发. 三家单位将精诚合作, 充分考虑三方定位和国家要求, 做好平台的技术支撑, 把成果转化为具有自主知识产权的产品.

会议一致同意聘任我校部春海副教授为公司总经理.