去库存问题模型建立与优化

选题：A

组员：

摘要

在这个商品经济蓬勃发展的时代，物流管理、库存管理都是其中不可避免的重要问题。本文建立了商品流通网络中，所涉及到的各个公司的库存、警戒值、订单量等、生产能力重要指标的计算模型。本模型使得整个销售网络，在到货率一定的前提下，生产能力一定和生产能力有所提高两种前提下，分别以最少的库存满足所有终端及其上级公司的订单请求。

在模型一中，我们通过所给出的数据，建立能够准确描述初始库存，现有库存，订单量，到货率之间关系的模型。我们分析得到，生产能力一定时，总库存随着到货率的提高而增大，具体数值关系见正文。在模型二中，我们在满足到货率 90%的条件下，基于上述几个变量之间的关系，通过对订单过程进行模拟，找到了每一种商品的最小流通库存，并得到了相应的警戒值 r 和订单量 q。在问题三中，将到货率提高至 95%，我们利用了与模型二中类似的方法得到了初始库存，然后根据题中生产能力提高的要求，对模型进行优化，并最终得到了不同生产能力下的总库存。

在文章最后对模型进行了评价。

关键词：运筹学 去库存 订货点法 java matlab

1、 问题重述

某行业货物供应商（Supplier）通过各公司（Fac），进而向下级子公司（Fac）直至零售商发行某种专业商品，货物的流通过程如下图。一般地，某个发货商有可能同时在其它订单中也作为收货商，所以该图只是显示了某批货物可能的运输销售流程，但不足以表示发货商与收获商的上下级关系，通常它们会形成一个网状结构，如附件。

另一附件数据集“产品流转数据”是一份销售链行为记录表。数据集的每条观测记录了一次订单情况，即某个上级发货商到下级收货商的某货物批次的情况，包括某上级供应商的某批次产品在某天流通到某个下级收货商的具体信息。在平日里，各公司都有一个初始库存，假设公司的库存量一旦小于某个值 r 就会立即向其某个上级下订单补货，订单量为常数 q。而上级供应商要向多个下级供货，因此下级发来的订单请求未必能得到满足，记下级收货商实际收到的货量占其需求量的百分比的值为到货率。目前该商品较为紧俏，末端收货商（实际使用部门）需求旺盛，到货率也仅有 90%。

该行业供应商关心如下问题，邀请你们小组在尚未提高生产能力之前提升到货率，降低流通库存。

（1）库存与到货率之间究竟有什么关系？

（2）求若要满足目前到货率 90%不变，并且使所有分销商的库存量总和最小，r 和 q 的值应该为多少？库存总和需要你自己定义。

（3）若生产能力提高，估算能使末端收货商的到货率提高至 95%，请重新估算供应商的最优库存。

2、 问题分析

本问题重点关注库存、到货率、警戒值和订单量之间的关系。在发货与收货的过程中，发货商与收货商两方的库存在完成一次订单后，都发生了一定的变化。根据题目的要求，收货商发出订单请求后，当发货商的货物充足时，则可以向收货商提供要所求订单量 q 的货物；而当发货商的货物不充足时，则尽可能满足收货商的订单请求，但这种情况下不能使到货率到达 100%。

本题目对所有终端的到货率有最小值的要求，由于生产能力没有得到提高，显然，对于上级公司的初始库存就必然有一定量的要求。否则不能满足随后所有订单的订货要求，而初始库存小会使得库存不足而且终端到货率极低。因此，本题的根本，就在于平衡到货率与初始库存之间的关系，来建立合适的模型，得到总库存值并对其进行优化。

在已经提供的数据中，详细地记录了每一笔供货订单的相关信息，通过数据处理与分析，我们可以每一次交易完成后的各个公司的库存状态，并以此作为初步分析的依据，找到符合题目要求的警戒值 r 和订单量 q，进而得到两者相互之间的关系。

初步确定各公司的警戒值 r 和订单量 q 后，我们将对这个结果进行优化。通过在一定的范围内，适当地改变初始库存量和生产能力，使得结果在最佳状态，同时满足题意。

2.1 问题一

在问题一中，要求出库存与到货率的关系。到货率与发货商库存和收货商请求量有关，而发货商的库存与其初始库存和中间订单引起的库存变化有关，只要根据以上几个变量的关系，就可以找到库存与到货率的理论关系。通过控制到货率的变化，进而使库存发生改变，通过绘制两者的数据变化曲线，可以清晰地看出库存与到货率之间的量变关系。

2.2 问题二

问题二中，要求满足到货率 90%不变，并使所有分销商的总库存最小，在此

条件下，求出相应的警戒值 r 和订单量 q。根据分析，生产能力不变，要想使分销商的总库存最小，最理想的情况是使每一个公司，在完成所有订单的过程中最小库存等于 0，且保持到货率恰好为 90%。这样可以使得在满足题意的前提下，总库存达到最佳状态。我们假设终端的库存最小，再通过各个上级发货商的最小库存，来对某一种产品销售过程中所涉及到的每一个公司初始库存进行计算。通过不同商品、流通当中公司处于的不同位置级别，可以归纳分析出各个类型公司警戒值 r、订单量 q 之间的相似之处。

2.3 问题三

问题三中，提高生产能力后，使得到货率增加到 95%，在这种情况下，重新估算供应商的最优库存。相比于到货率为 90%的情况，显然，在 95%的到货率情况下，所有尾端发货商的库存必定都有增加。在问题二的基础上，我们对程序当中的参数进行修改，用类似的方法可以算出本条件下的各个公司的初始库存。同时，由于生产能力提高，则生产商的初始库存可能会略有降低，通过随后的生产可以满足更多的货物需求。通过每一天最高级发货商的发货量，来确定提高后的生产能力，随后可以估算出每一种商品的最优库存。

在解决这三个问题中，我们通过每一次订单带来的变化，洞悉每一次变化后的结果，建立模型并通过程序对模型进行优化，以达到题目中要求的最佳状态。

3、 模型假设

1. 假设公司有一定的初始库存 old，在整个销售过程中，若库存 origin 小于安全库存 r，即向上级请求订单量，可知，某公司每一次请求订单时都有，

origin<=r，则此时的 max（origin）=r，更新 stockmax=origin，stockmax 为所求 r；

2. 某公司每一次向下级分发的订单量假设为下级请求的 90%，此时满足公司的库存最小以及流通库存最小。某公司每一次库存变化，记录总过程的库存最大值 max、最小值 min。

分析有，该公司在流通过程中库存最小值为 min，若改变初始库存 old，使 min 降为 0，即满足总库存最小的条件；

3. 所关心的仅为终端的到货率，假设每一次分法给终端的订单量已满足 90%的到货率，此时求最小的库存以及满足条件的 q 即可。记录每一次终端向上级请求订单时的订单量 count 和当前库存 origin。

4、符号说明

商品 i 的初始库存	N_{i0}
公司 i 的初始库存	M_{i0}
公司 i 的 j 时间的库存	W_{ij}
公司 i 的警戒值	R_i
某订单需求量	Q_i
实际供货量	S_i
生产能力	C
库存总和	L
到货率	h
实际收货量	G_i
公司等级	K_i
当地平均消费量	D_i

表 1 符号表

5、模型的建立与求解

5.1 问题一：库存与到货率的关系

5.1.1 模型建立

根据题意，到货率为实际发货量$\mathrm{S_i}$与订单请求量$\mathrm{Q_i}$之比，即

$$h = \frac{\mathrm{S}_i}{\mathrm{Q_i}}$$

根据题意，实际发货量$\mathrm{S_i}$与发货商现有库存$\mathrm{W_{ij}}$和订单请求量$\mathrm{Q_i}$的关系如下，

$$\mathrm{S_i} = \begin{cases} \mathrm{Q_m}\left(\mathrm{W_{ij}} \geq \mathrm{Q_i}\right) \\ \mathrm{W_{ij}}\,(\mathrm{W_{ij}} < \mathrm{Q_i}) \end{cases}$$

某公司的现有库存$\mathrm{W_{ij}}$与初始库存$\mathrm{M_{i0}}$、一段时间内的总发货量和总收货量的关系如下，

$$\mathrm{W_{ij}} = \mathrm{M_{i0}} - \sum_{1}^{\mathrm{i-1}} \mathrm{S_i} + \sum_{1}^{\mathrm{i-1}} \mathrm{G_i}$$

综合以上各个关系式，得到到货率 h 与初始库存$\mathrm{W_i}$的关系如下，

$$h = \begin{cases} \dfrac{\mathrm{S_i}}{\mathrm{Q}_i} = 1 \quad (\mathrm{W_{ij}} \geq \mathrm{Q_i}) \\ \dfrac{\mathrm{M_{i0}} - \sum_1^{\mathrm{i-1}} \mathrm{S_i} + \sum_1^{\mathrm{i-1}} \mathrm{G_i}}{\mathrm{Q_i}} \qquad (\mathrm{W_{ij}} < \mathrm{Q_i}) \end{cases}$$

5.1.2 模型求解

在解答模型之前，我们首先对所给出的数据进行了分析，并绘制了销售流程图如下（以 ZSYQTZDK 和 BFZDKZSY 和两种商品为例）：

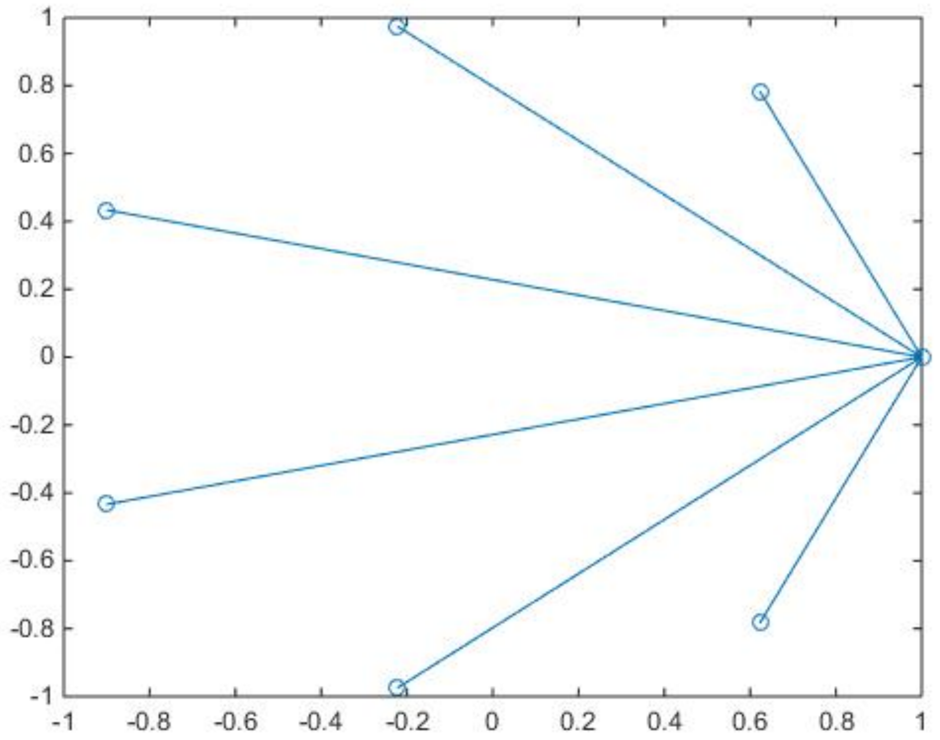


图 1 商品 ZSYQTZDK 的供应网络

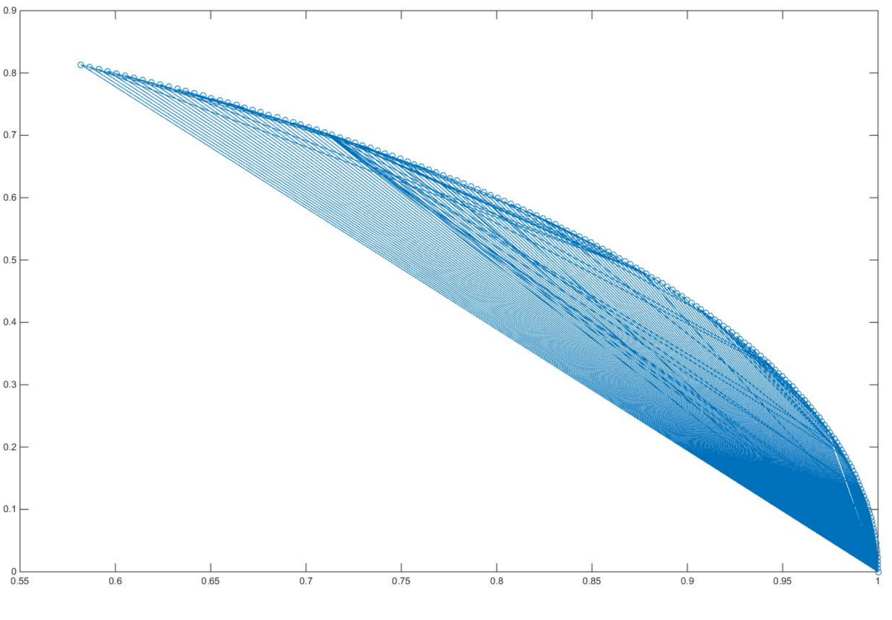


图 2 商品 BFZDKZSY 的供应网络

建立模型后，通过编程，控制到货率的变化，程序流程图如下：

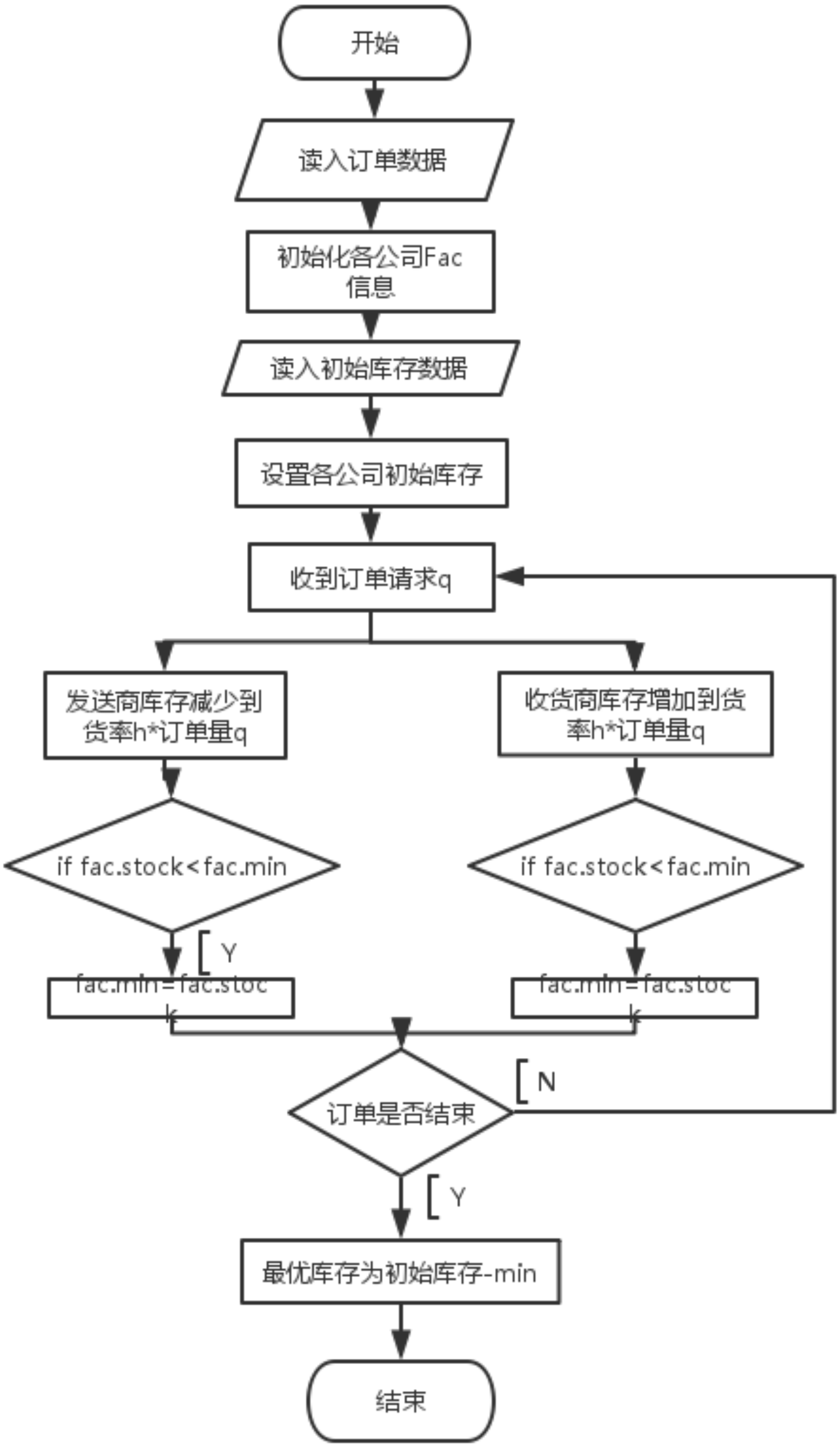


图 3 到货率控制程序流程图

得到了与其相对应的的初始库存总量，具体数据及相应的趋势折线图如下：

（1）JYECGRSα-2aZSY

到货率	45%	50%	55%	60%	65%
总库存	306232	340258	374283	408309	442335
到货率	70%	75%	80%	85%	
总库存	476361	510386	544412	578438	

表 2 商品 JYECGRSα-2aZSY 的库存与到货率关系

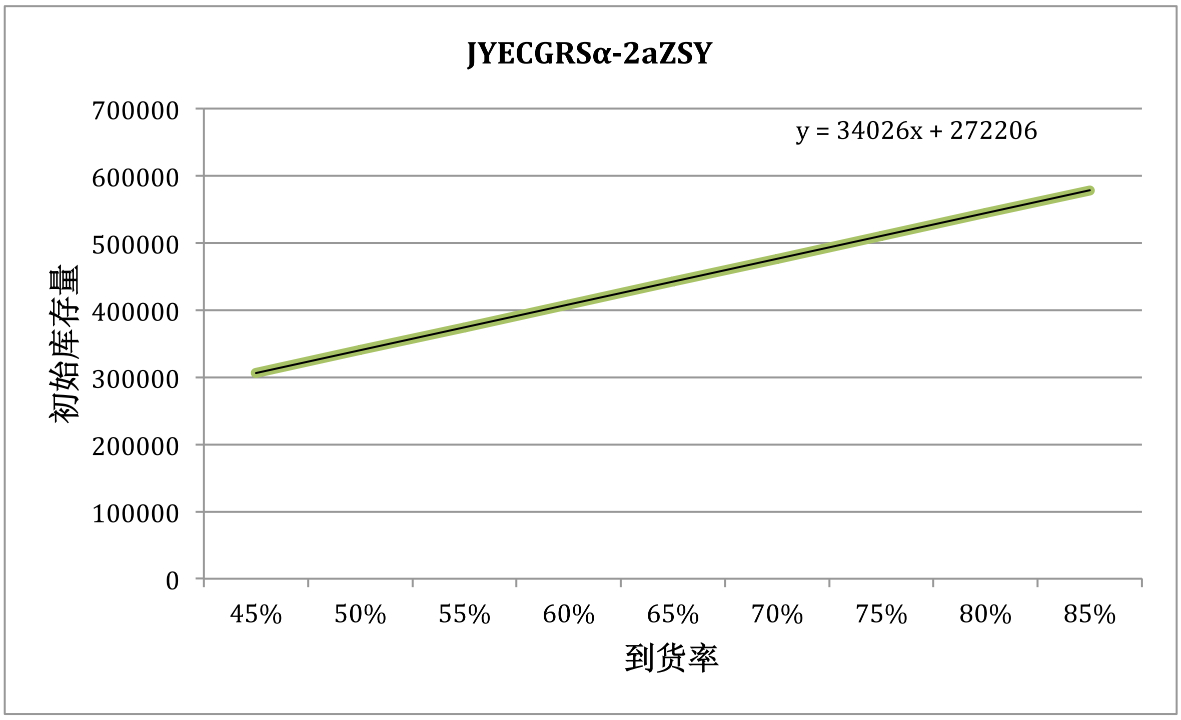


图 4 商品 JYECGRSα-2aZSY 的库存与到货率关系拟合曲线

得到商品 JYECGRSα-2aZSY 的库存与到货率关系为：

$$N_{10} = 34026h + 272206$$

（2）KPTBP

到货率	45%	50%	55%	60%	65%
总库存	1410128	1566809	1723490	1880171	2036852
到货率	70%	75%	80%	85%	
总库存	2193533	2350214	2506852	2663576	

表 3 商品 KPTBP 的库存与到货率关系

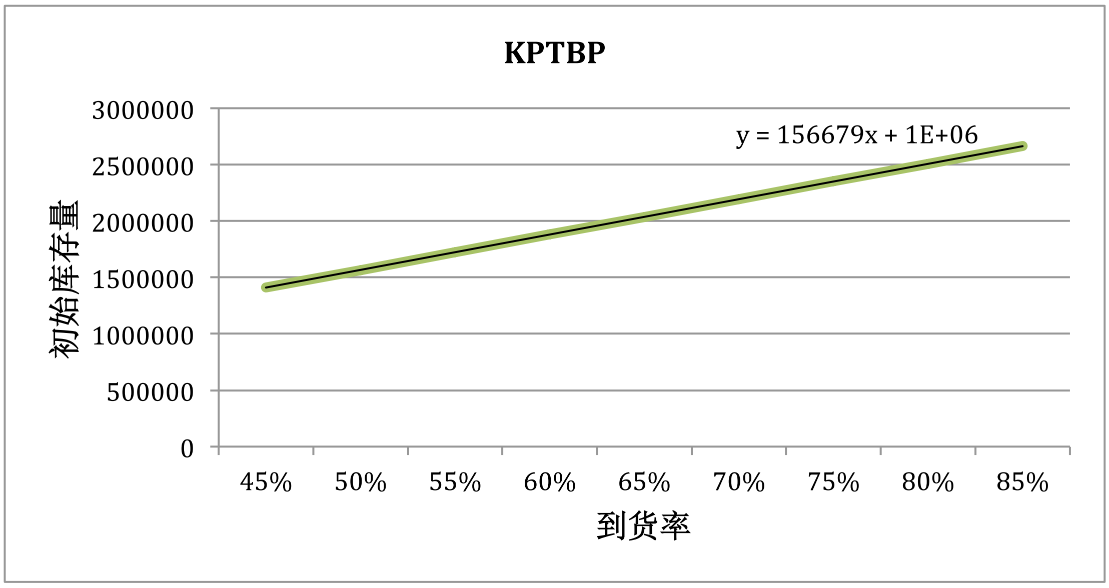


图 5 商品 KPTBP 的库存与到货率拟合曲线

得到关系式为：

$$N_{20} = 156679h + (1E + 06)$$

（3）MTMKF 酯 JN

到货率	45%	50%	55%	60%	65%
总库存	926275	1062528	1168780	1275033	1381286
到货率	70%	75%	80%	85%	
总库存	1487539	1593791	1700044	1806297	

表 4 商品 MTMKF 酯 JN 库存与到货率关系

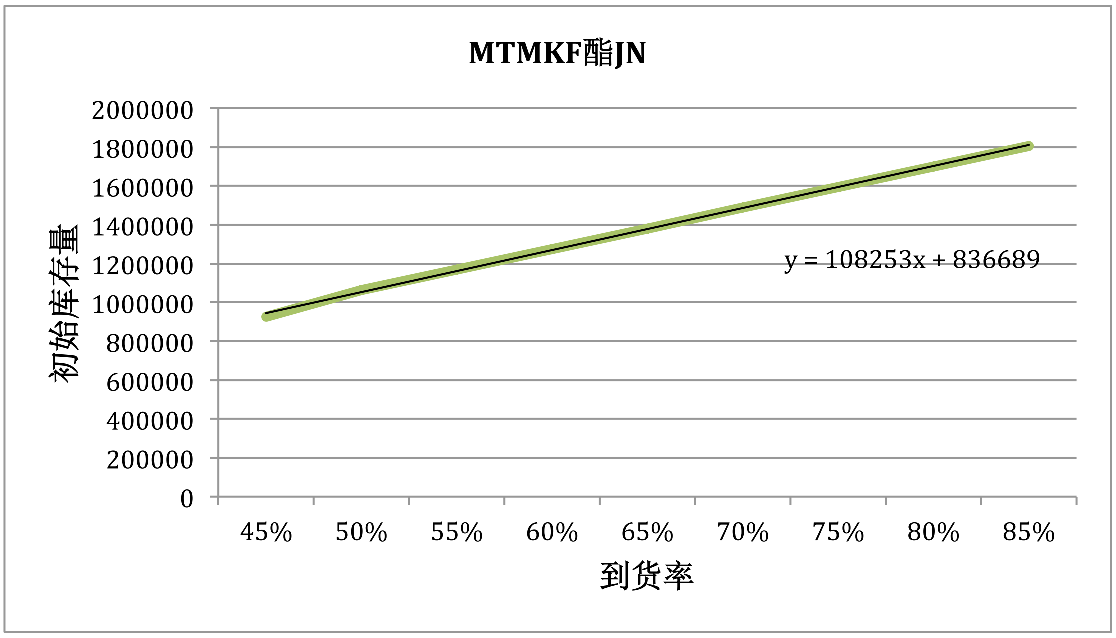


图 6 商品 MTMKF 酯 JN 库存与到货率关系拟合曲线

得到关系式为：

$$N_{30} = 108253h + 836689$$

由以上三组图表可知，对于任意一种商品可知，库存量随着到货率的增加而增加，且接近于正线性关系，即：

$$N_{i0} = k \cdot h + b \quad k,b 为常数$$

5.2 问题二：到货率 90%不变，使所有分销商的库存量总和最小，r和q的取值

5.2.1 模型建立

要想使分销商库存总和最小，在满足题意的情况下，应当使到货率全部等于 90%。由于一笔订单会直接影响到两个公司的库存，并会引发连锁的库存变化和收发货状态，本题目中的库存使在时刻变化的，应属于一个动态规划问题。

本问题的约束条件为：

（1） 对于任意一个终端的任意一笔订单到货率等于 90%，

$$\mathrm{h} = 90\%$$

（2） 知 max（origin）=r，即 stockmax，对于公司 i，Ri 的控制保证了一定的库存量，使得公司在库存充裕的情况下可以满足下级的订单量，从而影响了下级的到货率。而分销商公司等级不同，收到的订单请求量不同，则所需的库存不同，则不同公司因收到的订单量不同，需要维持的库存量就不同，所以有 r 的取值不仅与初始库存$\mathrm{M_{i0}}$有关，而且与公司所在等级K_i有关，

$$\mathrm{R_i = F(M_{i0}, K_i)}$$

分销商的安全库存$\mathrm{R_i}$用以保证下级的到货率，则对于下级同一时期的请求订单量，应满足到货率需求的部分，

$$\mathrm{R_i = h \times \sum_{j=1}^{n} Q_j}$$

零售商的安全库存 Ri 应能保证基本销售，Di 视为当地平均消费量，那么

$$\mathrm{R_i = D_i}$$

（3） 公司向上级的请求订单量以补充库存，由未补货之前的库存和补货后的库

存可知，

$$\mathrm{W_{ij}} < \mathrm{R_j}$$

$$\mathrm{W_{ij}} + \mathrm{Q_i} \geq \mathrm{R_i}$$

零售商在当前库存$\mathrm{W_t}$不满足基本销售，即小于安全库存时，需向上级请求订单，订单量至少能保证库存最终能达到安全库存，即，

$$\mathrm{Q_i} = \mathrm{R_i} - W_{ij} = \mathrm{R_i} - (\,W_{i(j-1)} - D_t)$$

初始状态下关系为，

$$\mathrm{Q_i} = \mathrm{R_i} - \mathrm{W_{i0}} = \mathrm{R_i} - \mathrm{M_{i0}} = \mathrm{D_i} - \mathrm{M_{i0}}$$

（4） 当分销商库存小于安全库存时，需要向上级请求订单时有，

$$Q_i = R_i - (W_{ij} - S_i) = R_i - W_{ij} + h \times \sum_{j=1}^{m} Q_j$$

初始状态下关系为，

$$Q_i = R_i - W_0 + h \times \sum_{j=1}^{m} Q_j = R_i - M_{i0} + h \times \sum_{j=1}^{m} Q_j$$

此问题的最终目的是使库存最小，则目标函数为

$$\mathrm{L} = \min \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{M_{i0}}$$

5.2.2 模型求解

解决问题时，首先要通过编程对于所给出的商品订单的数据进行处理，其流程图如下，

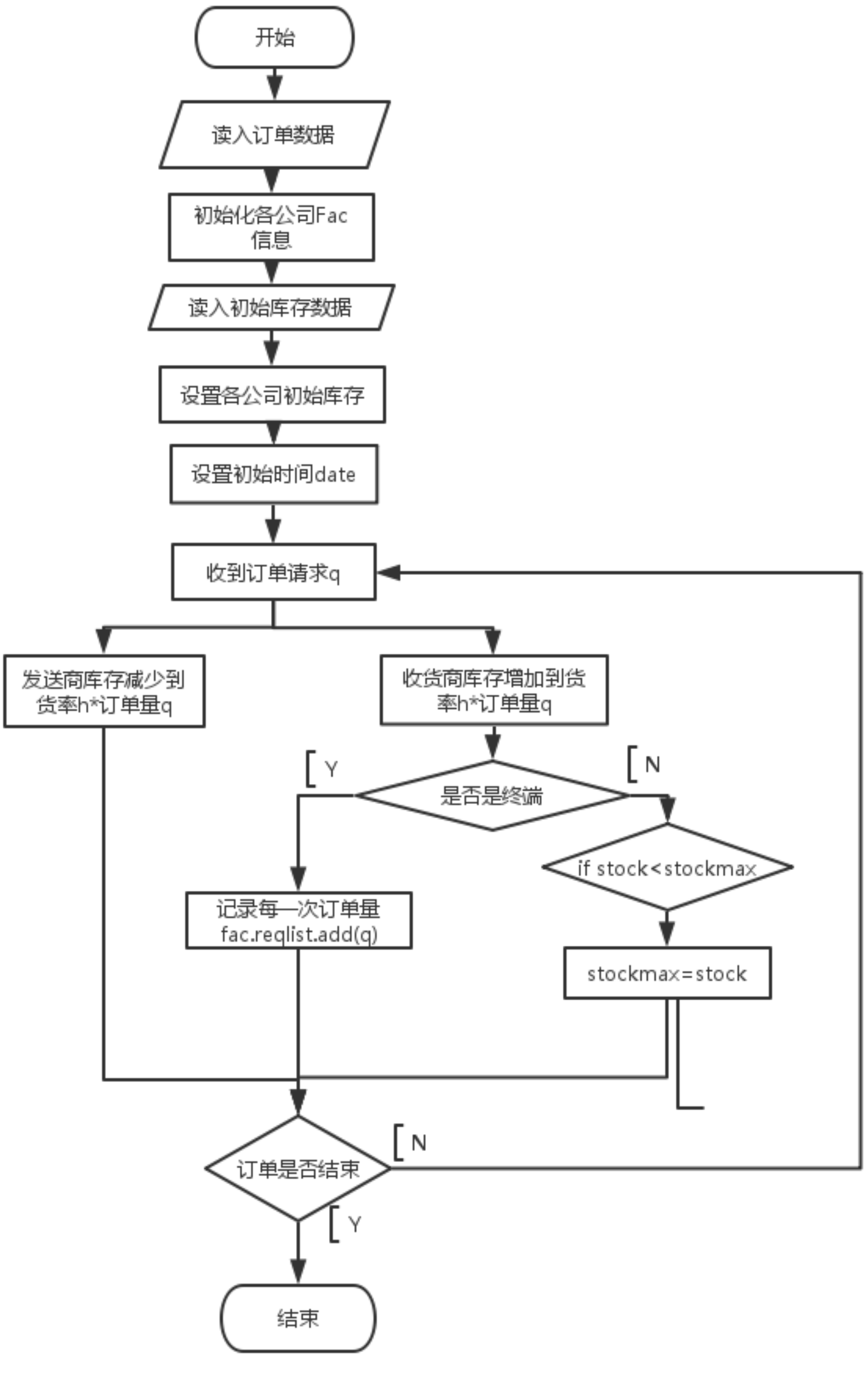


图 7 订单处理程序流图

具体步骤如下：

（1） 由于每一种商品的需求不一致，那么所要求的初始库存也必然不尽相同，

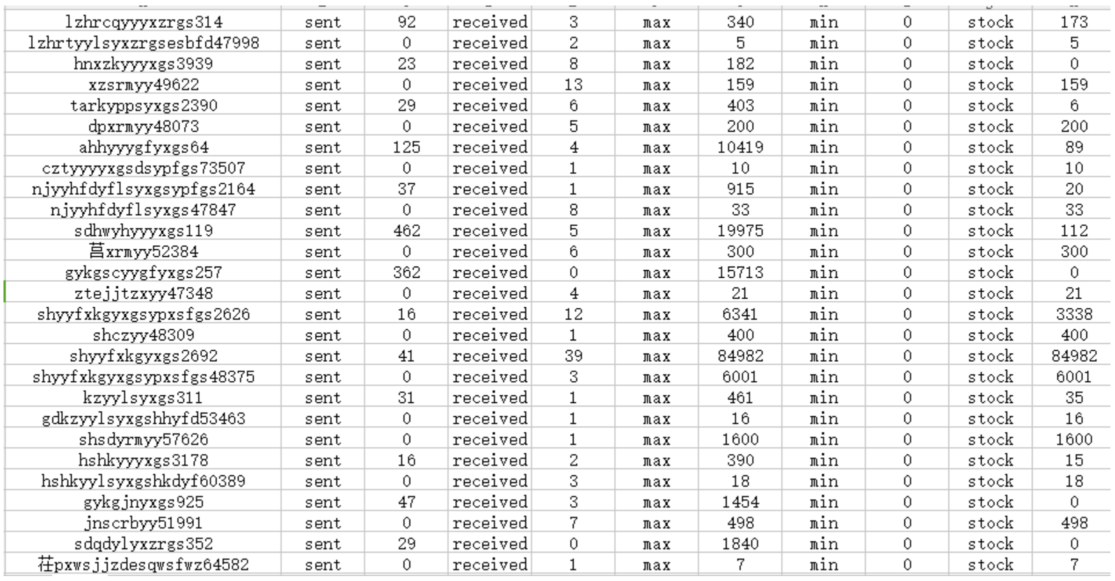
为了便于对每一类产品进行有针对性的分析，首先根据商品种类对数据进行分类并重新输出；

（2） 对于已经完成分类的数据，计算设定生产商，分销商以及终端初始库存；设计类 Fac 用来存储计算过程中每个公司的有关属性值。其成员有，公司名 name，初始库存 old，当前库存 origin，分发给下级的总次数 sendTime，分发给下级的订单量总和 countsend，收到订单的总次数 recTime，收到的订单量总和 countrec，总过程中库存的最大值 max、最小值 min，以及向上级申请订单时的最大库存 stockmax，是否为终端 isEnd 以及每次的请求订单量列表 req；

（3） 读取数据 Read()，对于各个公司的信息进行初始化，并设置初始库存 set()；

（4） 将订单记录按照时间顺序排序，并根据每一条订单进行交易模拟 work()，记录相关的数据变化，并将其输出 writeOut()；

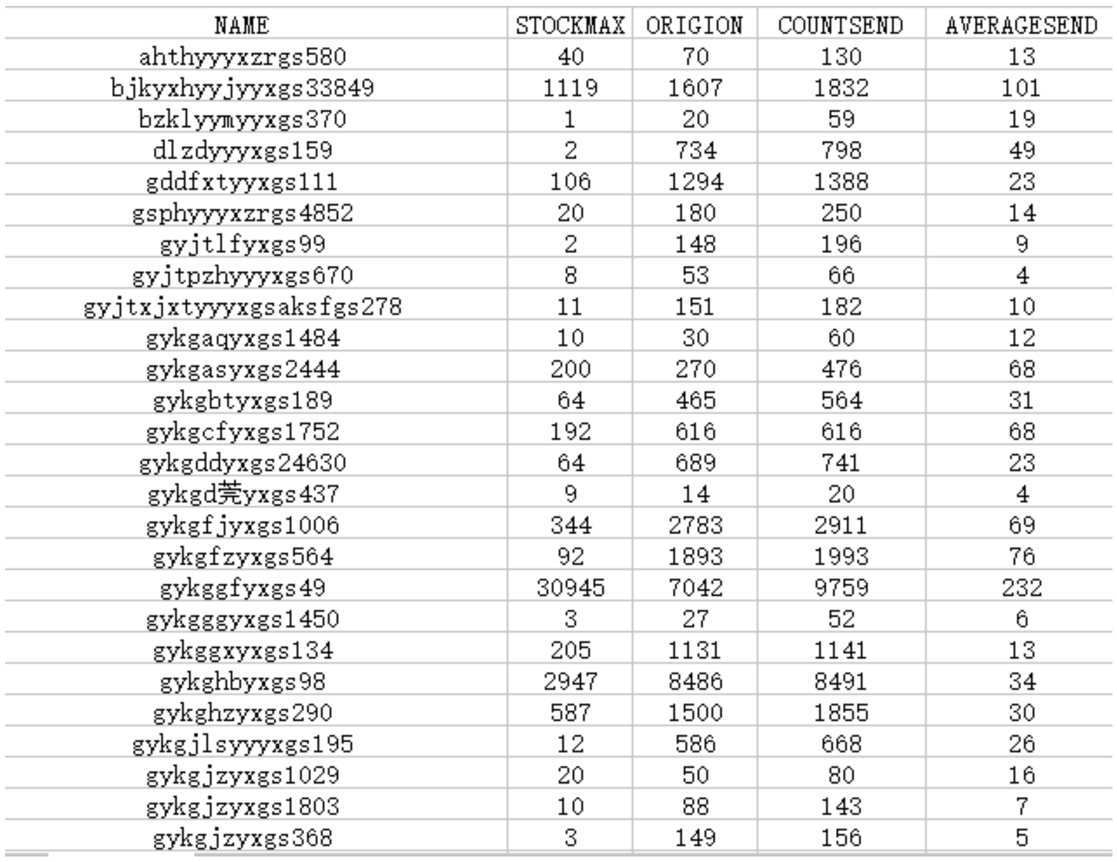
（5） 调整初始库存，重复（1）（2）（3），使得各公司在整个过程中的最小库存 min 为 0，此时初始库存为最优值；



lzhrcqyyyxzrgs314	sent	92	received	3	max	340	min	0	stock	173
lzhrtyylsyxzrgsesbfd47998	sent	0	received	2	max	5	min	0	stock	5
hnxzkyyyxgs3939	sent	23	received	8	max	182	min	0	stock	0
xzsrmyy49622	sent	0	received	13	max	159	min	0	stock	159
tarkyppsyxgs2390	sent	29	received	6	max	403	min	0	stock	6
dpxrmyy48073	sent	0	received	5	max	200	min	0	stock	200
ahhyyygfyxgs64	sent	125	received	4	max	10419	min	0	stock	89
cztyyyyxgsdsypfgs73507	sent	0	received	1	max	10	min	0	stock	10
njyyhfdyflsyxgsypfgs2164	sent	37	received	1	max	915	min	0	stock	20
njyyhfdyflsyxgs47847	sent	0	received	8	max	33	min	0	stock	33
sdhwyhyyyxgs119	sent	462	received	5	max	19975	min	0	stock	112
莒xrmyy52384	sent	0	received	6	max	300	min	0	stock	300
gykgscyygfyxgs257	sent	362	received	0	max	15713	min	0	stock	0
ztejjtzxyy47348	sent	0	received	4	max	21	min	0	stock	21
shyyfxkgyxgsypxsfgs2626	sent	16	received	12	max	6341	min	0	stock	3338
shczyy48309	sent	0	received	1	max	400	min	0	stock	400
shyyfxkgyxgs2692	sent	41	received	39	max	84982	min	0	stock	84982
shyyfxkgyxgsypxsfgs48375	sent	0	received	3	max	6001	min	0	stock	6001
kzyylsyxgs311	sent	31	received	1	max	461	min	0	stock	35
gdkzyylsyxgshhyfd53463	sent	0	received	1	max	16	min	0	stock	16
shsdyrmyy57626	sent	0	received	1	max	1600	min	0	stock	1600
hshkyyyxgs3178	sent	16	received	2	max	390	min	0	stock	15
hshkyylsyxgshkdyf60389	sent	0	received	3	max	18	min	0	stock	18
gykgjnyxgs925	sent	47	received	3	max	1454	min	0	stock	0
jnscrbyy51991	sent	0	received	7	max	498	min	0	stock	498
sdqdylyxzrgs352	sent	29	received	0	max	1840	min	0	stock	0
茌pxwsjjzdesqwsfwz64582	sent	0	received	1	max	7	min	0	stock	7

表 5 初始库存最优情况

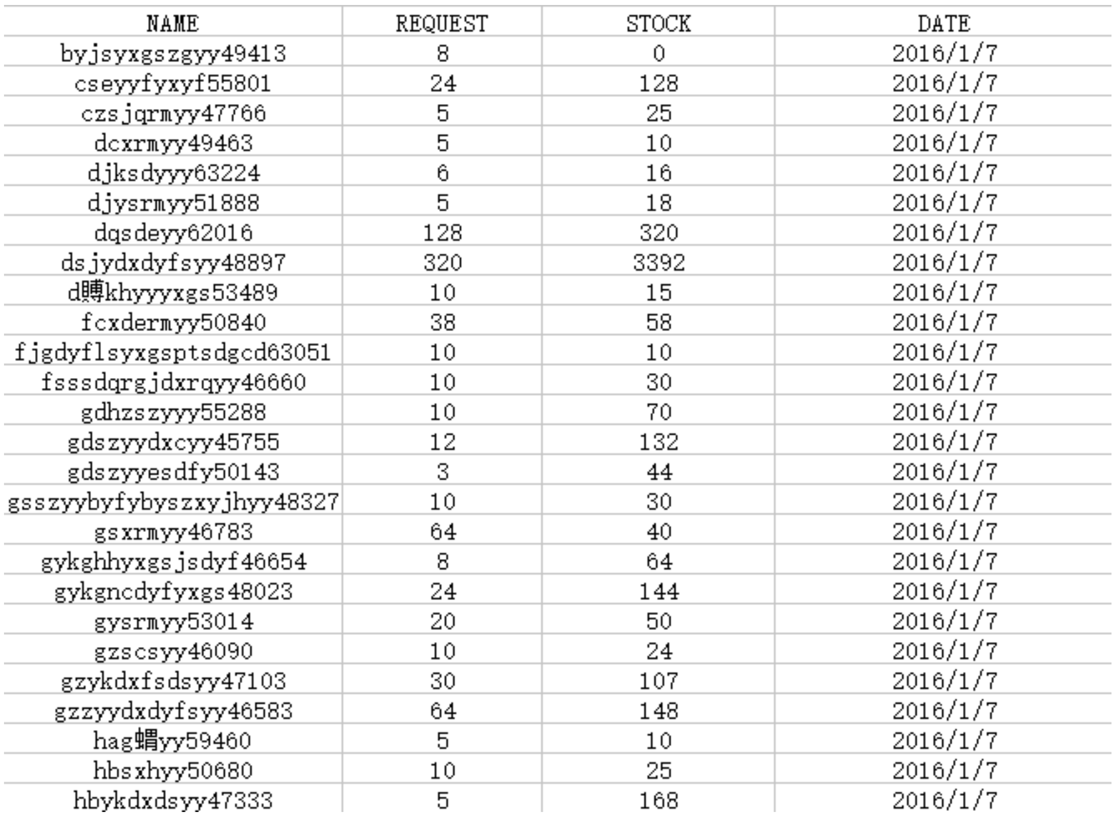
（6） 得到分销商的初始库存以及向上级请求时最大的库存 stockmax，对此进行分析求解；



NAME	STOCKMAX	ORIGION	COUNTSEND	AVERAGESEND
ahthyyyxzrgs580	40	70	130	13
bjkyxhyyjyyxgs33849	1119	1607	1832	101
bzklyymyyxgs370	1	20	59	19
dlzdyyyxgs159	2	734	798	49
gddfxtyyxgs111	106	1294	1388	23
gsphyyyxzrgs4852	20	180	250	14
gyjtlfyxgs99	2	148	196	9
gyjtpzhyyyxgs670	8	53	66	4
gyjtxjxtyyyxgsaksfgs278	11	151	182	10
gykgaqyxgs1484	10	30	60	12
gykgasyxgs2444	200	270	476	68
gykgbtyxgs189	64	465	564	31
gykgcfyxgs1752	192	616	616	68
gykgddyxgs24630	64	689	741	23
gykgd莞yxgs437	9	14	20	4
gykgfjyxgs1006	344	2783	2911	69
gykgfzyxgs564	92	1893	1993	76
gykggfyxgs49	30945	7042	9759	232
gykgggyxgs1450	3	27	52	6
gykggxyxgs134	205	1131	1141	13
gykghbyxgs98	2947	8486	8491	34
gykghzyxgs290	587	1500	1855	30
gykgjlsyyyxgs195	12	586	668	26
gykgjzyxgs1029	20	50	80	16
gykgjzyxgs1803	10	88	143	7
gykgjzyxgs368	3	149	156	5

表 6 分析求解初始库存和向上请求最大库存

（7） 得到终端每次请求 req 和当前库存 origin 的数据，对此进行分析求解；



NAME	REQUEST	STOCK	DATE
byjsyxgszgyy49413	8	0	2016/1/7
cseyyfyxyf55801	24	128	2016/1/7
czsjqrmyy47766	5	25	2016/1/7
dcxrmyy49463	5	10	2016/1/7
djksdyyy63224	6	16	2016/1/7
djysrmyy51888	5	18	2016/1/7
dqsdeyy62016	128	320	2016/1/7
dsjydxdyfsyy48897	320	3392	2016/1/7
d膊khyyyxgs53489	10	15	2016/1/7
fcxdermyy50840	38	58	2016/1/7
fjgdyflsyxgsptsdgcd63051	10	10	2016/1/7
fsssdqrgjdxrqyy46660	10	30	2016/1/7
gdhzszyyy55288	10	70	2016/1/7
gdszyydxcyy45755	12	132	2016/1/7
gdszyyesdfy50143	3	44	2016/1/7
gsszyybyfybyszxyjhyy48327	10	30	2016/1/7
gsxrmyy46783	64	40	2016/1/7
gykghhyxgsjsdyf46654	8	64	2016/1/7
gykgncdyfyxgs48023	24	144	2016/1/7
gysrmyy53014	20	50	2016/1/7
gzscsyy46090	10	24	2016/1/7
gzykdxfsdsyy47103	30	107	2016/1/7
gzzyydxdyfsyy46583	64	148	2016/1/7
hag蜎yy59460	5	10	2016/1/7
hbsxhyy50680	10	25	2016/1/7
hbykdxdsyy47333	5	168	2016/1/7

表 7 分析求解 r.q

上表中，stock 是警戒值 r，request 是订单量 q。

5.3 问题三：若生产能力提高，估算能使末端收货商的到货率提高至 95%，供应商的最优库存

5.3.1 模型建立

生产能力提高，且到货率要求也提高。整体思路上与第二题类似，只是由于生产能力的提高，初始库存略有改变。

根据题目要求，需要使每一天的库存和生产之和大于订单需求量，因此我们得到其约束条件为

$$M + C > Q_1$$

$$M + 2C - Q_1 > Q_2$$

$$M + 3C - Q_1 - Q_2 > Q_3$$

……

$$M + nC - Q_1 - Q_2 \dots\dots - Q_{n-1} > Q_n$$

5.3.2 模型求解

求解第三个问题与第二个问题的过程类似，不同之处有两点，首先是到货率提高到了 95%。程序流程图如下：

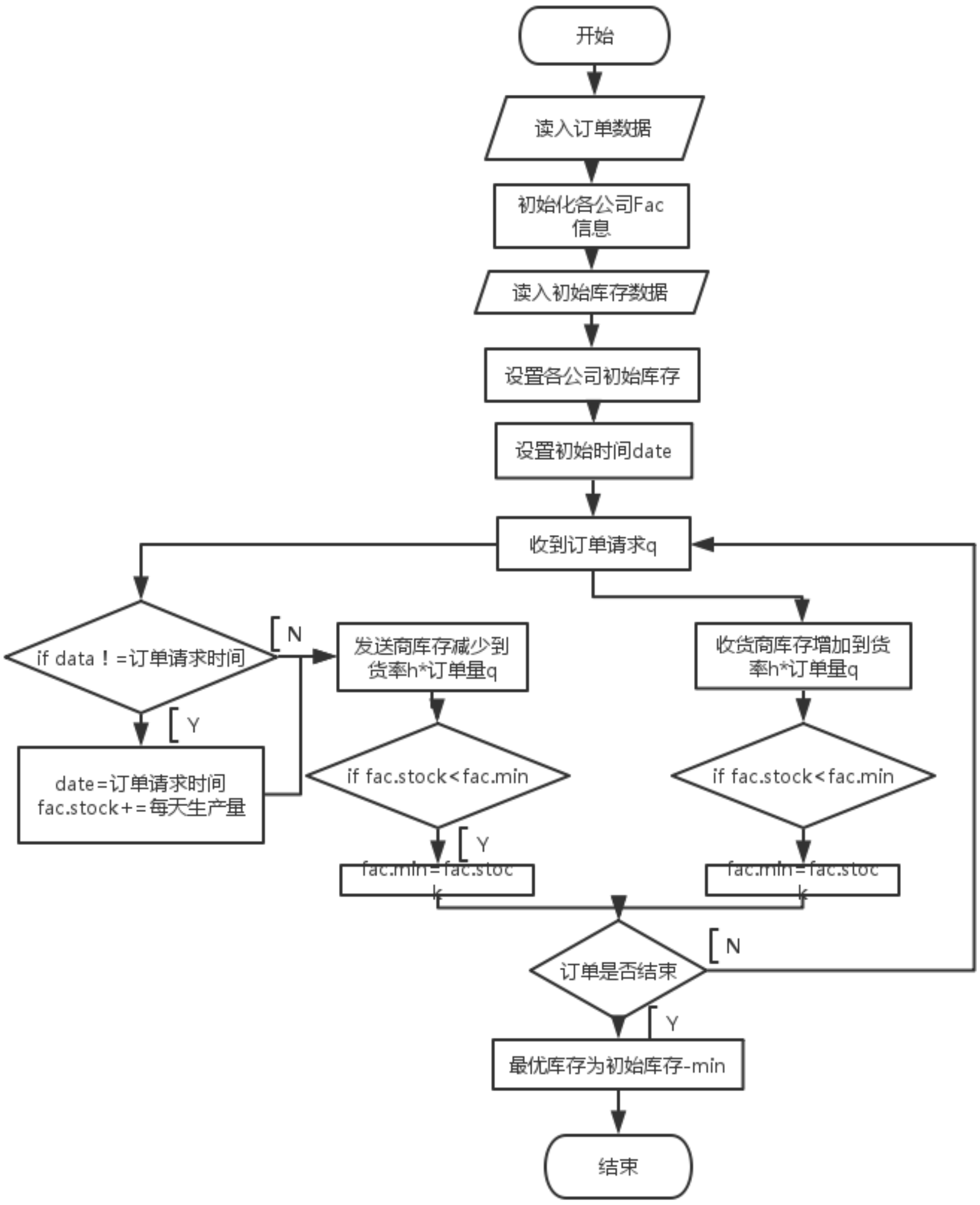


图 8 到货率提高条件下的最优库存求解程序流图

进行模拟交易的核心代码如下：

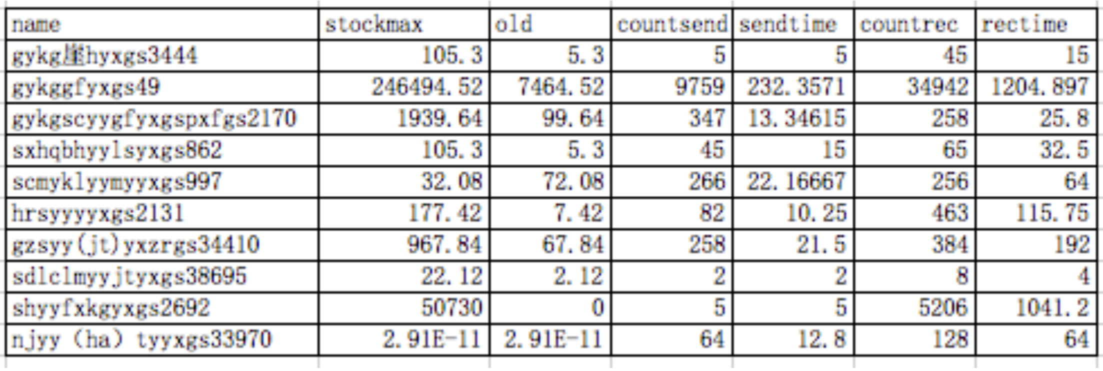
Work()核心代码:

```
        //发货商的信息处理
        Fac
```

```
fac=list.get(Integer.parseInt(FacName.get(deliver.getContents())));
                 int num=fac.getOrigin()-Integer.parseInt(count.getContents());
                 if(num>fac.max) fac.max=num;
          if(num<fac.min) fac.min=num;
                 if(fac.isEnd){//若为终端，输出当前订单量和其当前库存
          File reqf=new File("endsend.txt");
                      BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));

writerq.write(deliver.getContents()+","+count.getContents()+","+fac.getOrigin()
+"\r\n");
                    writerq.close();
                 }



writer.write(deliver.getContents()+","+String.valueOf(fac.isEnd())+","+fac.getOri
gin()+","+num+","+date.getContents()+"\r\n");
                 fac.setOrigin(num);
                  //处理收货商的信息
          fac=list.get(Integer.parseInt(FacName.get(recevier.getContents())));
                 num=fac.getOrigin()+Integer.parseInt(count.getContents());

writer.write(recevier.getContents()+","+String.valueOf(fac.isEnd())+","+fac.getO
rigin()+","+num+","+date.getContents()+"\r\n");
                    if(num>fac.max) fac.max=num;
               if(num<fac.min) fac.min=num;
                    if(fac.isEnd){//若为终端，输出当前订单量和其当前库存
                    File reqf=new File("req.txt");
                   BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));
                   writerq.write(recevier.getContents());
```

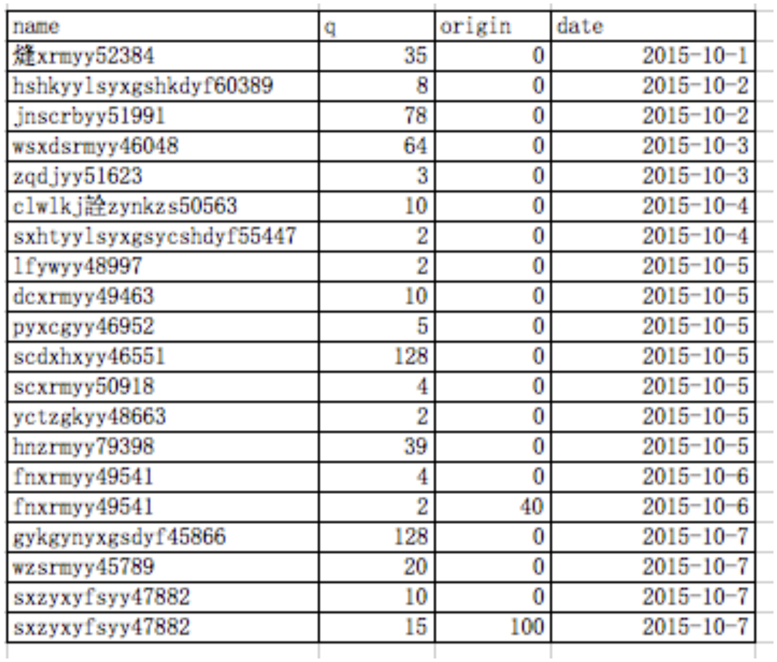
```
    writerq.write(","+count.getContents()+","+fac.getOrigin()+","+fac.old+","+da
te.getContents()+"\r\n");
                    writerq.close();

                  }else{//若不是终端
             if(fac.getSendTime()!=0){//若不是生产商，即为分销商，输出其每次
订单的要求数，以及当前库存量
             File reqf=new File("rec.txt");
                         BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));

writerq.write(recevier.getContents()+","+count.getContents()+","+fac.getOrigin(
)+","+sheet.getCell(19,i).getContents()+","+date.getContents()+"\r\n");

                         writerq.close();
                         if(fac.getOrigin()>fac.stockmax)
                        fac.stockmax=fac.getOrigin();

             }
                   }
               fac.setOrigin(num);
```

在到货率 95%的前提下，以 JYECGRSα-2aZSY 商品为例，得到结果如下：



name	stockmax	old	countsend	sendtime	countrec	rectime
gykg崖hyxgs3444	105.3	5.3	5	5	45	15
gykggfyxgs49	246494.52	7464.52	9759	232.3571	34942	1204.897
gykgscyygfyxgspxfgs2170	1939.64	99.64	347	13.34615	258	25.8
sxhqbhyylsyxgs862	105.3	5.3	45	15	65	32.5
scmyklyymyyxgs997	32.08	72.08	266	22.16667	256	64
hrsyyyyxgs2131	177.42	7.42	82	10.25	463	115.75
gzsyy(jt)yxzrgs34410	967.84	67.84	258	21.5	384	192
sdlclmyyjtyxgs38695	22.12	2.12	2	2	8	4
shyyfxkgyxgs2692	50730	0	5	5	5206	1041.2
njyy（ha）tyyxgs33970	2.91E-11	2.91E-11	64	12.8	128	64

表 8 商品 JYECGRSα-2aZSY 到货率 95%时的交易明细



name	q	origin	date
縫xrmyy52384	35	0	2015-10-1
hshkyylsyxgshkdyf60389	8	0	2015-10-2
jnscrbyy51991	78	0	2015-10-2
wsxdsrmyy46048	64	0	2015-10-3
zqdjyy51623	3	0	2015-10-3
clwlkj詮zynkzs50563	10	0	2015-10-4
sxhtyylsyxgsycshdyf55447	2	0	2015-10-4
lfywyy48997	2	0	2015-10-5
dcxrmyy49463	10	0	2015-10-5
pyxcgyy46952	5	0	2015-10-5
scdxhxyy46551	128	0	2015-10-5
scxrmyy50918	4	0	2015-10-5
yctzgkyy48663	2	0	2015-10-5
hnzrmyy79398	39	0	2015-10-5
fnxrmyy49541	4	0	2015-10-6
fnxrmyy49541	2	40	2015-10-6
gykgynyxgsdyf45866	128	0	2015-10-7
wzsrmyy45789	20	0	2015-10-7
sxzyxyfsyy47882	10	0	2015-10-7
sxzyxyfsyy47882	15	100	2015-10-7

表 9 商品 JYECGRSα-2aZSY 到货率 95%时订单量与初始库存

在 95%到货率的前提下，各种商品的总库存如下：

商品名称	总库存
JYECGRSα-2aZSY	649211
KPTBP	2976938
MTMKF 酯 JN	2018802
ZSYQTZDK	1195510
BFZDKZSY+D10	1346600

表 10 到货率 95%时各商品总库存

在到货率为 95%的情况下，各种商品总库存随着生产能力改变的变化如下列图表所示：

（1）KPTBP

生产能力	50	100	150	200	250	300	350	400
总库存	2969759	2962559	2948159	2948159	2940959	2933759	2926559	2919359
生产能力	500	600	700	800	900	1000	1100	
总库存	2904959	2890559	2876159	2861759	2847359	2832959	2818559	

表 11 商品 KPTBP 总库存随着生产能力改变的变化

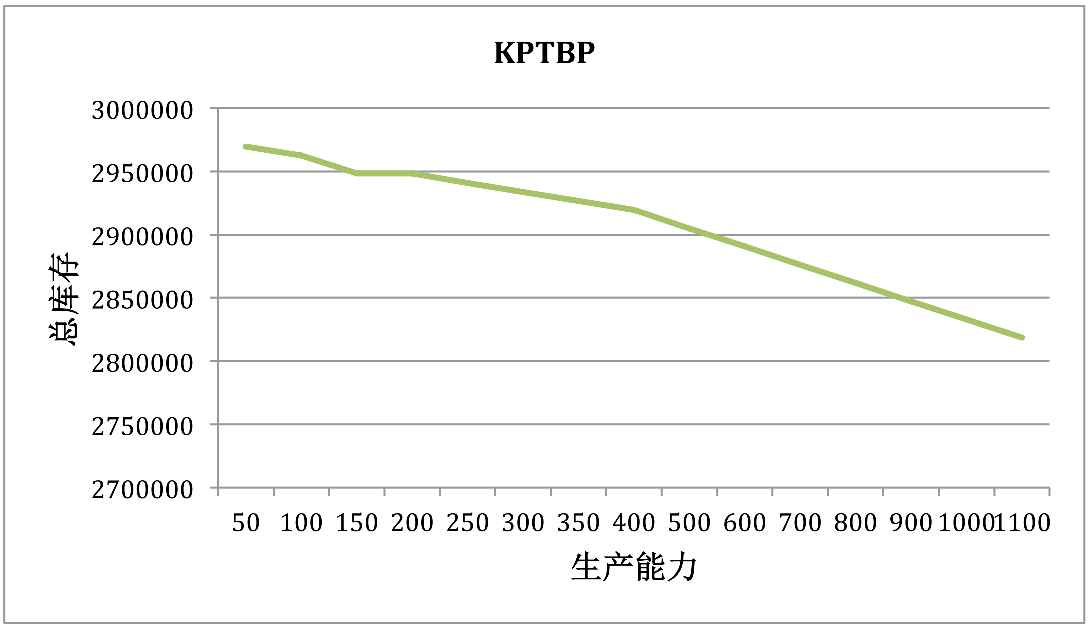


图 9　商品 KPTBP 生产能力与总库存曲线拟合

（2）JYECGRSα-2aZSY

生产能力	50	100	150	200
总库存	639550	632600	625650	618700
生产能力	250	300	350	400
总库存	611750	604800	597850	590900

表 12 商品 JYECGRSα-2aZSY 总库存随着生产能力改变的变化

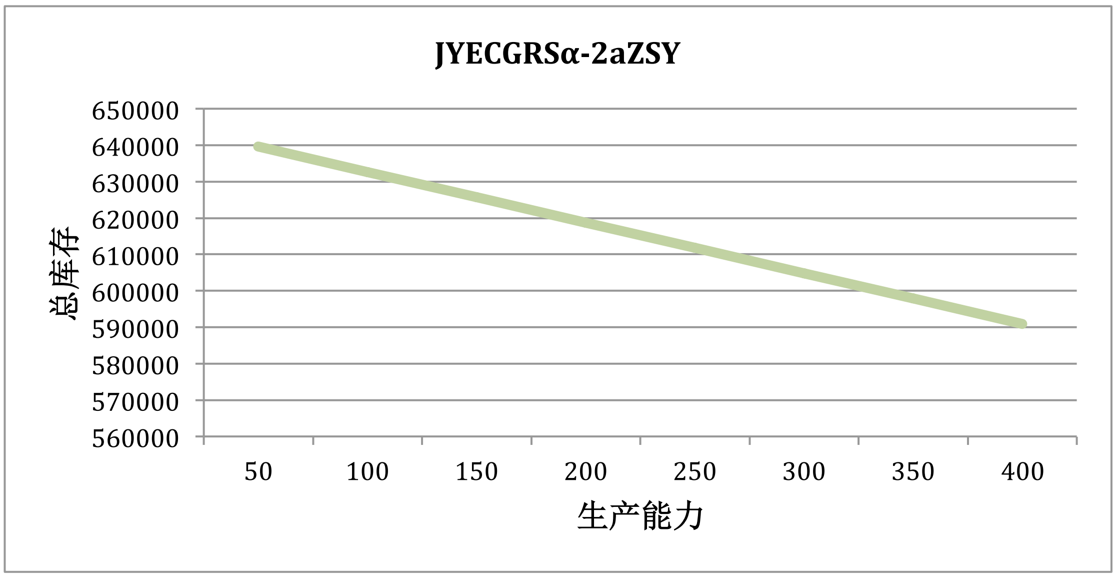


图 10　商品 JYECGRSα-2aZSY 生产能力与总库存曲线拟合

（3）MTMKF 酯 JN

生产能力	50	100	150	200	250	300
总库存	2011554	2004304	1997054	1989804	1982554	1975304
生产能力	350	400	500	600	700	800
总库存	1968054	1960804	1946304	1931804	1917304	1902804

表 13 商品 MTMKF 酯 JN 总库存随着生产能力改变的变化

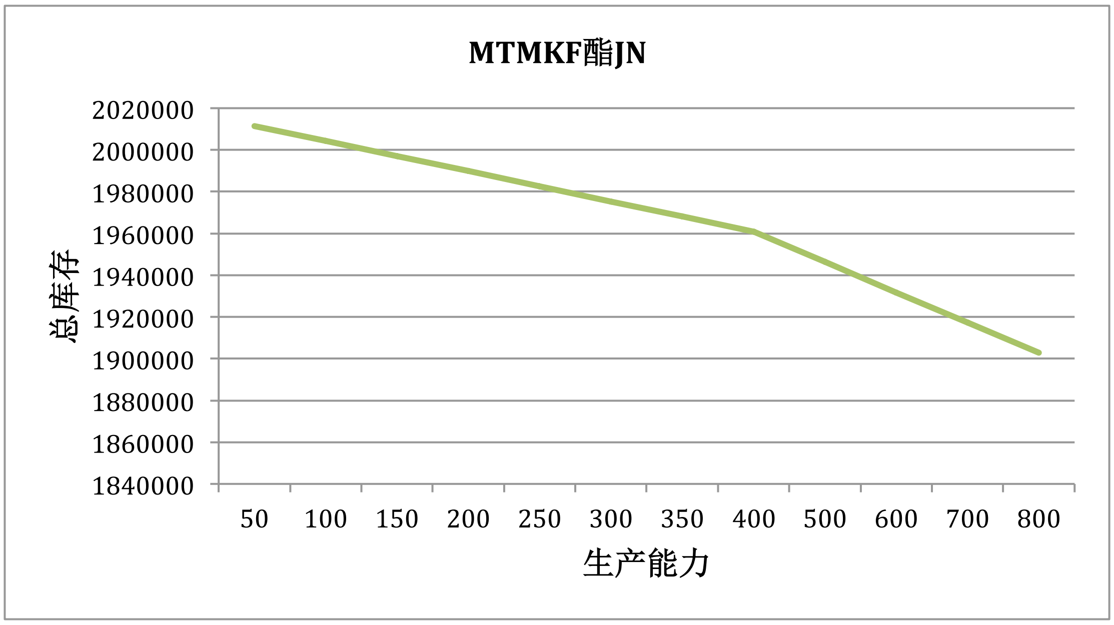


图 11　商品 MTMKF 酯 JN 生产能力与总库存曲线拟合

由上述图表可以看出，随着生产力的增加，各种商品的初始库存呈下降状态。也就是说，到货率和生产能力同时提高，带来的结果可能是初始库存的降低。对于到货率 90%和 95%两种情况的库存我们分别进行了计算，发现当生产力提高到一定程度，总有一个时刻后者的库存低于前者。

6、模型评价

6.1 模型的优点

本次问题中，我们所建立的模型是基于所给出的数据的，在数据处理和分析的基础上进行了后续的一系列操作，比较了不同的到货率，不同的生产能力条件下的库存，并得到了对于每一个公司的订单量 q 和警戒值 r。我们对每一类商品，每一类公司都做了分别的研究，在满足题意的前提下，得到了比较准确的答案。

6.2 模型的缺点

总体上来说，我们在有限的条件下，解决了题目所给出的三个问题，但是，我们的模型也受到了所给数据的限制，没能够很好的模拟销售商随机销售的情况。现实中，实际的销售情况会更加分散，也更加随机。但基于实际情况和数据分析，订单量 q 和警戒值 r 是一个定值，因此我们认为有限的数据也足以得到其相对准确的数值。

参考文献

[1] 陈义文 存货管理中经济进货批量的几种数学模型.黄冈职业技术学院学报，2002,4(3)：46

[2] 孙磊;朱琼;张洁；随机提前期下考虑动态紧急订货的库存模型.管理科学，2010，3

[3] 彭红军 两级生成与需求不确定下煤炭企业供应链模型研究. 中国矿业大学，2011

附录

1. fac.java(用于存放公司各属性的类)

```
package mathtest;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class Fac {
    String name;
    int received;
    int send;
    boolean isEnd;
    int recTime;
    int sendTime;
    double origin;
    public double max;
    public double min;
    public String Date;
    public List rec;
    public List req;
    public List stock;
    public double old;
    public double stockmax;
    public double countsend;
    public double countrec;
    public double getOrigin() {
        return origin;
    }

    public void setOrigin(double origin) {
        this.origin = origin;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getReceived() {
        return received;
    }

    public void setReceived(int received) {
        this.received += received;
    }

    public int getSend() {
        return send;
    }

    public void setSend(int send) {
        this.send += send;
    }

    public boolean isEnd() {
        return isEnd;
    }

    public void setEnd(boolean isEnd) {
        this.isEnd = isEnd;
    }

    public int getRecTime() {
        return recTime;
    }

    public void setRecTime() {
        this.recTime++;
    }

    public int getSendTime() {
        return sendTime;
    }

    public void setSendTime() {
        this.sendTime++;
    }

public Fac(){}
public Fac(String _name){
    rec=new ArrayList();
    req=new ArrayList();
    stock=new ArrayList();
    name=_name;
    received=0;
    send=0;
    isEnd=false;
    recTime=0;
    sendTime=0;
    origin=0;
    max=-999999999;
    min=999999999;
    stockmax=-9999999;
}
}
```

2.classification.java(用于分离不同商品的订单)

```
package mathtest;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;
import jxl.write.biff.RowsExceededException;

public class classification {
    public Map<String,String> name;
    public List count;
    public  void cut() throws BiffException, IOException{
       name=new HashMap<String,String>();
       count=new ArrayList();
       File file=new File("test.xls");

        Workbook book= Workbook.getWorkbook(file);
        Sheet sheet=book.getSheet(0);
        int count1=0;
        int count2=0;
        int count3=0;
        int count4=0;
        int count5=0;
        int countname=0;

        for(int i=1;i<sheet.getRows();i++)
      {
             Cell cell=sheet.getCell(0,i);
             int pos=0;
            if(name.containsKey(cell.getContents())){
               pos=(int)
count.get(Integer.parseInt(name.get(cell.getContents())));
count.set(Integer.parseInt(name.get(cell.getContents())),pos+1);
        }else{
               name.put(cell.getContents(),
String.valueOf(countname));
     countname++;
     count.add(1);;
            }
     File f =new File(cell.getContents()+".txt");
     BufferedWriter writers = new BufferedWriter(new FileWriter(f,
true));
   writers.write(sheet.getCell(0,i).getContents()+","+
                      sheet.getCell(1,i).getContents()+","+
                      sheet.getCell(2,i).getContents()+","+
                      sheet.getCell(3,i).getContents()+","+
                      sheet.getCell(4,i).getContents()+","+
                      sheet.getCell(5,i).getContents()+","+
                      sheet.getCell(6,i).getContents()+","+
                      sheet.getCell(7,i).getContents()+","+
                      sheet.getCell(8,i).getContents()+","+
                      sheet.getCell(9,i).getContents()+","+
                      sheet.getCell(10,i).getContents()+","+
                      sheet.getCell(11,i).getContents()+","+
                      sheet.getCell(12,i).getContents()+","+
                      sheet.getCell(13,i).getContents()+","+
                      sheet.getCell(14,i).getContents()+","+
                      sheet.getCell(15,i).getContents()+","+
                      sheet.getCell(16,i).getContents()+","+
                      sheet.getCell(17,i).getContents()+","+
                      sheet.getCell(18,i).getContents()+","+
                      sheet.getCell(19,i).getContents()+","+
                      sheet.getCell(20,i).getContents()+"\r\n");
              writers.close();
      }
        book.close();
    }
    public void test() throws BiffException, IOException,
RowsExceededException, WriteException{
       File file=new File("stockmax.xls");
        Workbook book= Workbook.getWorkbook(file);
        Sheet sheet=book.getSheet(0);
        int countFac=0;
WritableWorkbook workbook = Workbook.createWorkbook(file,book);
                            WritableSheet ws = workbook.getSheet(0);
        for(int i=0;i<sheet.getRows();i++)
     {
            Cell cell=sheet.getCell(3,i);
            int y=(int)testwork(Integer.parseInt(cell.getContents()));
            Label label =new Label(7,i,String.valueOf(y));
            ws.addCell(label);
     }
        workbook.write();
        workbook.close();
        book.close();
    }
    public double testwork(int x){
       return
+2.6796456497063676e-91*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x-3.5345458360732273e-87*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x+1.2021364252267405e-83*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x-2.6050065894927842e-80*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x+2.5866436922781046e-77*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x-2.6043054761334593e-75*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x+4.0384329845610147e-72*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x-2.031146917980219e-70*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+
1.1713034768586835e-67*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
```

```
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-9.4516468
44972022e-66*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+3.891218165581184e-64
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-9.49961271760889e-63*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x-1.201631466302957e-60*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+9
.056729192543332e-59*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-1.5853041710378675e-5
7*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x-2.4150335379104684e-56*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x+6.817516795872157e-55*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+1.7758135435958387e-53
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x-2.504988809986608e-52*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-1.36372030
97653484e-50*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+2.1457873565456116e-49*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-4.337
614201885195e-48*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-3.9257687761419596e-47*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+8.5953963
30993783e-47*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x-1.7626226494477829e-44*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+8.31020378281618e-43*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x+1.9919675472193433e-41*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x-3.375441183189099e-40*x*x*x*x*x*x*x*x*x*x*
x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+3.5755219145293346e-3
9*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+2
.9785023384542157e-38*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
```

```
*x*x*x*x*x*x*x*x-1.4842034964743565e-36*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+2.58460506353346e-35*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-5.2542235949810295e-34*x
*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-3.8346884159
949957e-32*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x-3.
1790426377122798e-31*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x-1.406078489963069e-30*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x-1.00839080209407e-29*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x-2.7365721931901727e-28*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x*x-4.41011422274469e-26*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x*x-2.1768892896163862e-25*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x*x*x-1.5856786360560416e-23*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x
*x*x*x+6.064875645296991e-23*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+
5.592542598427084e-21*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+2.1533228
286180623e-19*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+2.559115742580842e-
18*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x*x+1.4255327992808455e-16*x*x*x*x*x
*x*x*x*x*x*x*x*x*x*x*x+9.426825100801819e-16*x*x*x*x*x*x*x*x*x*x*x*x*
x*x*x-2.04327607540316e-14*x*x*x*x*x*x*x*x*x*x*x*x*x*x-4.328886543290
388e-13*x*x*x*x*x*x*x*x*x*x*x*x*x+4.1048683088520764e-12*x*x*x*x*x*x*
x*x*x*x*x*x+2.0742904792102063e-10*x*x*x*x*x*x*x*x*x*x*x+4.6105080819
54596e-9*x*x*x*x*x*x*x*x*x*x-6.294560007524235e-8*x*x*x*x*x*x*x*x*x-0
.0000028370789485797405*x*x*x*x*x*x*x*x+0.000014436178418825486*x*x*x
*x*x*x*x+0.000908689640414568*x*x*x*x*x*x-0.008513840228280042*x*x*x*
x*x+0.00884959226858893*x*x*x*x-0.07438501719226913*x*x*x+0.275164982
0804235*x*x+3.311391116322225*x+7.786319792539923;
        }
}
```

3.convey.java(进行库存计算，订单量转运等运算)

```
package mathtest;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.lang.reflect.Array;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;
import jxl.write.biff.RowsExceededException;

public class convey {
    public static List<Fac> list=new ArrayList<Fac>();
    public static  Map<String,String> FacName=new
HashMap<String,String>();
    public static Connection connect;
    public static String productor;
    public static void Read() throws IOException, BiffException{
        File file=new File("KPTBP.xls");

         Workbook book= Workbook.getWorkbook(file);
        Sheet sheet=book.getSheet(0);
        int countFac=0;
        for(int i=0;i<sheet.getRows();i++)
        {
           Cell name=sheet.getCell(7,i);//（列，行）
           if(sheet.getCell(11,i).getContents().indexOf("生产")!=-1){
              productor=sheet.getCell(7,i).getContents();
           }
           if(!FacName.containsKey(name.getContents())){
          FacName.put(name.getContents(),String.valueOf(countFac));
              countFac++;
              Fac fac=new Fac(name.getContents());
               Cell send=sheet.getCell(18,i);
              fac.setSend(Integer.parseInt(send.getContents()));
              fac.setSendTime();
              fac.countsend+=Integer.parseInt(send.getContents());
          double
num=fac.getOrigin()-Integer.parseInt(send.getContents());
               list.add(fac);
           }else{
              Fac
fac=list.get(Integer.parseInt(FacName.get(name.getContents())));
              Cell send=sheet.getCell(18,i);
              fac.setSend(Integer.parseInt(send.getContents()));
              fac.setSendTime();
              double
num=fac.getOrigin()-Integer.parseInt(send.getContents());
              fac.countsend+=Integer.parseInt(send.getContents());
           }
           Cell name2=sheet.getCell(12,i);
           if(!FacName.containsKey(name2.getContents())){
          FacName.put(name2.getContents(),String.valueOf(countFac));
              countFac++;
              Fac fac=new Fac(name2.getContents());
               Cell rec=sheet.getCell(18,i);
```

```
                fac.setReceived(Integer.parseInt(rec.getContents()));
                fac.setRecTime();
                Cell end=sheet.getCell(16,i);
                fac.countrec+=Integer.parseInt(rec.getContents());
                if(end.getContents().indexOf("终端")!=-1){
                    fac.setEnd(true);

                }else{
                    fac.rec.add(Integer.parseInt(rec.getContents()));
                    fac.stock.add(fac.getOrigin());
                }
                double
num=fac.getOrigin()+Integer.parseInt(rec.getContents());

                 list.add(fac);
            }else{
                Fac
fac=list.get(Integer.parseInt(FacName.get(name2.getContents())));
                Cell rec=sheet.getCell(18,i);
                fac.setReceived(Integer.parseInt(rec.getContents()));
                fac.setRecTime();
                Cell end=sheet.getCell(16,i);
                fac.countrec+=Integer.parseInt(rec.getContents());
                if(end.getContents().indexOf("终端")!=-1){
                    fac.setEnd(true);
                    fac.req.add(Integer.parseInt(rec.getContents()));
                }else{
                    fac.rec.add(Integer.parseInt(rec.getContents()));
                }
                double
num=fac.getOrigin()+Integer.parseInt(rec.getContents());

            }
```

```
         }
         book.close();
        //writeOut();
    }

    public static void writeOut() throws IOException{
         int counmax=0;
         int counmin=0;
         for(int i=0;i<list.size();i++){
             BufferedWriter writer = new BufferedWriter(new
FileWriter(new File("mildout.txt"), true));
             BufferedWriter writer2 = new BufferedWriter(new
FileWriter(new File("top.txt"), true));
             BufferedWriter writer3 = new BufferedWriter(new
FileWriter(new File("endout.txt"), true));
             BufferedWriter writer4 = new BufferedWriter(new
FileWriter(new File("out.txt"), true));
             Fac fac=list.get(i);
             writer4.write(fac.getName()+" has sent "+fac.getSendTime()+
                   " times and received "+fac.getRecTime()+
 " the ans= "+String.valueOf(fac.getReceived()-fac.getSend())+
                   " origin "+String.valueOf(fac.getOrigin())+
                   " max="+fac.max+"  min="+fac.min+" stock=
"+fac.getOrigin()+"\r\n");
               writer4.close();
         if(fac.isEnd){
             writer3.write(fac.getName()+" has sent "+fac.getSendTime()+
                   " times and received "+fac.getRecTime()
    " the ans= "+String.valueOf(fac.getReceived()-fac.getSend())+
                   " origin "+String.valueOf(fac.getOrigin())+

 " max="+fac.max+"  min="+fac.min+" stock= "+fac.getOrigin()+"\r\n");
```

```
                writer3.close();
                if(fac.getRecTime()==0){
                    counmax++;
     writer2.write(fac.getName()+" the ans=
"+String.valueOf(fac.getReceived()-fac.getSend())+"\r\n");
                }
                writer2.close();
                if(fac.getSendTime()==0)counmin++;
                }else{
                    if(fac.getSendTime()!=0){
    writer.write(fac.getName()+" has sent "+fac.getSendTime()+
     " times and received "+fac.getRecTime()+
     " the ans= "+String.valueOf(fac.getReceived()-fac.getSend())+
     " origin "+String.valueOf(fac.getOrigin())+
     "max="+fac.max+"  min="+fac.min+" stock=
"+fac.getOrigin()+"\r\n");
        writer.close();
                    }
                }
            }

    }

    public static Fac search(String name){
        Fac fac=null;
        for(int i=0;i<list.size();i++){
            if(list.get(i).getName().equals(name))
                return list.get(i);
        }

        return fac;
    }
    public static void setstock() throws BiffException, IOException,
RowsExceededException, WriteException{
       File file=new File("KPTBPset.xls");
   Workbook book= Workbook.getWorkbook(file);
        Sheet sheet=book.getSheet(0);
        int countFac=0;
    WritableWorkbook workbook = Workbook.createWorkbook(file,book);
           WritableSheet ws = workbook.getSheet(0);
        for(int i=0;i<sheet.getRows();i++)
      {
           Cell cell=sheet.getCell(0,i);
           Cell cell2=sheet.getCell(1,i);
           Fac fac=search(cell.getContents());

           Label label=new
Label(28,i,String.valueOf(Double.parseDouble(cell2.getContents())-fac
.min));
           ws.addCell(label);
      }
        workbook.write();
        workbook.close();
    }
    public static void work() throws BiffException, IOException,
SQLException, RowsExceededException, WriteException{
    File file=new File("KPTBP.xls");
       double p1=0.94444444;
       double p2=0.88888889
       double p3=0.83333333;
       double p4=0.77777778;
       double p5=0.72222222;
       double p6=0.66666667;
       double p7=0.61111111;
       double p8=0.55555556;
       double p9=0.5;
```

```
        double p10=1.0555556;
        double p11=1.1111111;
         double per=p10;
         double make=1100;
         Workbook book= Workbook.getWorkbook(file);
         Sheet sheet=book.getSheet(0);
         int countFac=0;
         String today="";
         for(int i=0;i<sheet.getRows();i++)
        {
            Cell deliver=sheet.getCell(7,i);//（列，行）
            Cell recevier=sheet.getCell(12,i);
            Cell date=sheet.getCell(17,i);
            Cell count=sheet.getCell(18,i);
             if(i==0) today=date.getContents();
            File f=new File("sqlout.txt");
    BufferedWriter writer = new BufferedWriter(new FileWriter(f, true));
            if(!today.equals(date.getContents())){
                System.out.println(productor);
        Fac pro=list.get(Integer.parseInt(FacName.get(productor)));
            pro.setOrigin(pro.getOrigin()+make);
            today=date.getContents();
            }
             Fac
fac=list.get(Integer.parseInt(FacName.get(deliver.getContents())));
             double
num=fac.getOrigin()-per*Integer.parseInt(count.getContents());
            if(num>fac.max) fac.max=num;


        if(num<fac.min) fac.min=num;
             if(fac.isEnd)
                 File reqf=new File("endsend.txt");
                BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));
writerq.write(deliver.getContents()+","+count.getContents()+","+fac.g
etOrigin()+"\r\n");
             writerq.close();
            }
writer.write(deliver.getContents()+","+String.valueOf(fac.isEnd())+",
"+fac.getOrigin()+","+num+","+date.getContents()+"\r\n");
            fac.setOrigin(num);
fac=list.get(Integer.parseInt(FacName.get(recevier.getContents())));

num=fac.getOrigin()+per*Integer.parseInt(count.getContents());
writer.write(recevier.getContents()+","+String.valueOf(fac.isEnd())+"
,"+fac.getOrigin()+","+num+","+date.getContents()+"\r\n");
             if(num>fac.max) fac.max=num;
              if(num<fac.min) fac.min=num;
             if(fac.isEnd){
               File reqf=new File("req.txt");
              BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));
            writerq.write(recevier.getContents());
            double x=fac.getReceived()/fac.getRecTime()-fac.old;
          //
writerq.write(","+fac.getReceived()/fac.getRecTime()+","+x+"\r\n")
   writerq.write(","+count.getContents()+","+fac.getOrigin()+","+fac
.old+","+date.getContents()+"\r\n");
             writerq.close();
          }else{
     if(fac.getSendTime()!=0){
               File reqf=new File("rec.txt");
         BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));
writerq.write(recevier.getContents()+","+count.getContents()+","+fac.
getOrigin()+","+sheet.getCell(19,i).getContents()+","+date.getContent
s()+"\r\n");
    //writerq.write(recevier.getContents()+","+fac.countsend+","+fac.
getOrigin()+"\r\n");
                // fac.countsend=0;
                    writerq.close();
                    if(fac.getOrigin()>fac.stockmax)
                       fac.stockmax=fac.getOrigin();
                  }
                }

               fac.setOrigin(num);
               writer.close();
                System.out.println(i);
        }
    }

    public static void cleandata() throws BiffException, IOException{
        File file=new File("rec.xls");
         Workbook book= Workbook.getWorkbook(file);
         Sheet sheet=book.getSheet(0);
         for(int i=0;i<sheet.getRows();i++)
        {
            if(sheet.getCell(3,i).getContents().indexOf("销售")!=-1){
                File reqf=new File("rec2.txt");
                   BufferedWriter writerq = new BufferedWriter(new
FileWriter(reqf, true));
    writerq.write(sheet.getCell(0,i).getContents()+","+sheet.getCell(
1,i).getContents()+","+sheet.getCell(2,i).getContents()+"\r\n");
                  writerq.close();
            }
        }
         book.close();
    }
    public static void set() throws BiffException, IOException,
RowsExceededException, WriteException{
        File file=new File("KPTBPset.xls");
          Workbook book= Workbook.getWorkbook(file);
         Sheet sheet=book.getSheet(0);
         int countFac=0;
        WritableWorkbook workbook = Workbook.createWorkbook(file,book);
   WritableSheet ws = workbook.getSheet(0);
         for(int i=0;i<sheet.getRows();i++)
        {
         Cell cell=sheet.getCell(0,i);
             Cell cell2=sheet.getCell(1,i);
             Fac fac=search(cell.getContents());
             fac.setOrigin(Double.parseDouble(cell2.getContents()));
             if(Double.parseDouble(cell2.getContents())>fac.max)
fac.max=Double.parseDouble(cell2.getContents());
                if(Double.parseDouble(cell2.getContents())<fac.min)
fac.min=Double.parseDouble(cell2.getContents());
             fac.old=fac.getOrigin();
        }
         workbook.write();
         workbook.close();
    }
    public static void maxwrite() throws IOException{
        for(int i=0;i<list.size();i++){
            Fac fac=list.get(i);
            if(fac.stockmax>0){
             File reqs=new File("stockmax.txt");
             File req1=new File("1.txt");
             File req2=new File("2.txt");
             File req3=new File("3.txt");
             File req4=new File("4.txt");
             BufferedWriter writers = new BufferedWriter(new
FileWriter(reqs, true));
            BufferedWriter writer1 = new BufferedWriter(new
FileWriter(req1, true));
            BufferedWriter writer2 = new BufferedWriter(new
FileWriter(req2, true));
            BufferedWriter writer3 = new BufferedWriter(new
FileWriter(req3, true));
            BufferedWriter writer4 = new BufferedWriter(new
FileWriter(req4, true));

writers.write(fac.getName()+","+fac.stockmax+","+fac.old+","+fac.coun
tsend+","+fac.countsend/fac.getSendTime()+","+fac.countrec+","+fac.co
untrec/fac.getRecTime()+"\r\n");
            //writers.write(+fac.old+","+fac.stockmax+"\r\n");
            writer1.write(","+fac.stockmax);
            writer2.write(","+fac.old);
            writer3.write(","+fac.countsend/fac.getSendTime());
            writer4.write(","+fac.countrec/fac.getRecTime());
             writers.close();
             writer1.close();
             writer2.close();
             writer3.close();
             writer4.close();
            }
        }
    }
    public static void main(String[] args) throws BiffException,
IOException{

        try {
 Class.forName("com.mysql.jdbc.Driver");     //加载MYSQL JDBC驱动程序
              //Class.forName("org.gjt.mm.mysql.Driver");
             System.out.println("Success loading Mysql Driver!");
```

```
          }
          catch (Exception e) {
            System.out.print("Error loading Mysql Driver!");
            e.printStackTrace();
          }
          try {
            classification c=new classification();
           // c.test();
          //  c.cut();
           Read();
           set();
           work();
           setstock();
            //cleandata();
          ///  writeOut();
           maxwrite();
          }
          catch (Exception e) {
            System.out.print("get data error!");
            e.printStackTrace();
          }
      //  */
    }
}
```

4.FacTo.java

```
package dataDraw;
import jxl.*;
import java.awt.List;
import java.awt.print.Book;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.util.ArrayList;
import javax.swing.text.html.HTMLDocument.Iterator;
import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;
import jxl.write.biff.RowsExceededException;
public class Fac {
    public int orderPointR;
    public int orderNumQ;
    public int originStore;
    public int currentStore;
    public ArrayList nast=new ArrayList();
    Workbook book1,book2;
    WritableWorkbook wwb;
    ArrayList tf=new ArrayList();
    ArrayList facname=new ArrayList();
    int[][] tradeMat=new int[1085][1085];
    public void getOriginStore() throws BiffException, IOException{
        book1=Workbook.getWorkbook(new
File("C:/Users/sse430-ssed/Desktop/fname.xls"));
        Sheet sheet=book1.getSheet(0);
    Cell cell1,cell2;
        for(int i=0;i<sheet.getRows();i++){
                cell1=sheet.getCell(0, i);
                cell2=sheet.getCell(2, i);
                  ArrayList natemp=new ArrayList();
                String facname=cell1.getContents();
                int ori=Integer.parseInt(cell2.getContents());
                natemp.add(facname);
                natemp.add(ori);
                nast.add(natemp);
        }
        book1.close();
        System.out.print(nast);
    }
    public int getIndex(ArrayList a,String s){
        for(int i=0;i<a.size();i++){
            if(a.get(i).equals(s)){
            return i;
            }
        }
        System.out.print("false");
        return 0;
    }
    public void getMatrix() throws BiffException, IOException{
        //得到某类商品的供应关系
book2=Workbook.getWorkbook(new File("C:/Users/qq1/Desktop/jdd.xls"));
        Sheet s=book2.getSheet(0);
        Cell cell1,cell2,cell3;
        for(int i=0;i<s.getRows();i++){
            cell1=s.getCell(0,i);
            cell2=s.getCell(1,i);
            String deliver=cell1.getContents();
            String receiver=cell1.getContents();
            if(!facname.contains(deliver)){
              facname.add(deliver);
            }
            if(!facname.contains(receiver)){
                facname.add(receiver);
            }
        }
        for(int i=0;i<s.getRows();i++){
            cell2=s.getCell(0,i);
            cell3=s.getCell(1,i);
            String deliver=cell2.getContents();
            String receiver=cell3.getContents();
            int row=getIndex(facname,deliver);
            int colum=getIndex(facname,receiver);
            tradeMat[row][colum]=1;
        }
        book2.close();
    }
    public void ifGetItem() throws BiffException, IOException,
WriteException{
        //检测订货是否送达
        book2=Workbook.getWorkbook(new
File("C:/Users/sse430-ssed/Desktop/JD.xls"));
        Sheet s=book2.getSheet(0);
        Cell cell1d,cell2r,cell3type,cell4num;
        for(int i=0;i<s.getRows();i++){
            cell1d=s.getCell(7,i);
            cell2r=s.getCell(12,i);
            cell3type=s.getCell(16,i);
            cell4num=s.getCell(18,i);
            String deliver=cell1d.getContents();
            String receiver=cell2r.getContents();
            String type=cell3type.getContents();
            int itemNum=Integer.parseInt(cell4num.getContents());
            int get2=0;
            int isDel=0;
            for(int j=0;j<3835;j++){
                ArrayList temp1=(ArrayList) nast.get(j);
        if(temp1.get(0).equals(deliver)){
            get2=get2+1;
            if((Integer)temp1.get(1)>=itemNum){
                isDel=1;
                currentStore=(Integer)temp1.get(1)-itemNum;
                ArrayList t1=new ArrayList();
                t1.add(deliver);
                t1.add(currentStore);
                nast.set(j,t1);
                tf.add("1");
            }
            else{
                isDel=0;
                tf.add("0");
                break;
            }
        }
    }
    for(int k=0;k<3835;k++){
        ArrayList temp2=(ArrayList) nast.get(k);
        if(temp2.get(0).equals(receiver)){
            get2=get2+1;
            if(isDel==1){
                currentStore=(Integer)temp2.get(1)+itemNum;
                ArrayList t1=new ArrayList();
                t1.add(receiver);
                t1.add(currentStore);
                nast.set(k,t1);
            }
            else{
                break;
            }
        }
```

```
            }
        }
        book2.close();
    }
    public void print() throws IOException, WriteException{
        wwb=Workbook.createWorkbook(new
File("C:/Users/sse430-ssed/Desktop/JD.xls"));/      WritableSheet
ws=wwb.createSheet("res", 1);
        for(int i=0;i<10224;i++){
        Label label=new Label(1,i,(String) tf.get(i));
        try {
            ws.addCell(label);
        } catch (RowsExceededException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (WriteException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
        wwb.close();
        FileOutputStream fs = new FileOutputStream(new
File("C:/Users/qq1/Desktop/matout.txt"));
        PrintStream p = new PrintStream(fs);
        for(int i=0;i<1085;i++){
            for(int j=0;j<1085;j++){
                    p.print(tradeMat[i][j]+" ");
            }
            p.print(";"+"\n");
        }
        p.close();
    //  System.out.println(tf);
    }
    public static void main(String[] args) throws BiffException, 
IOException, WriteException{
        Fac fac=new Fac();
        fac.getMatrix();
        fac.print();
    }
}
```

5.data.m

```
%Variable y,x1,x2,x3;
y=[5,8925,92,20,2,161,116,40,30,64,480,2947,40,10,25,10,68,746,44,168
,3188,34,56,10,708,192,342,64,40,75,12,15,162,11,22,4,16,66,20,64,36,
108,1,5,197,30,30945,64,94,106,40,252,6,25,30,102,4,11988,3,5,103,587
,1,2,64,298,64,20,64,10,34,12,98,38,344,5,20,60,90,10,20,57,8,50,256,
226,205,44,30,64,200,60,245,60,20,20,278,6,64,26,4,24,65,2,51,10,13,3
0,13,10,64,6,32,6,30,192,22,11,8,2,1,15,5,3,64,20,64,24,1,1,9,1119,15
4,4,84,5073];
x1=[20,22322,1893,157,148,1007,4424,70,170,500,570,8486,508,61,243,13
9,484,2833,608,1060,8672,66,359,88,2246,510,2766,447,606,800,586,5,22
18,151,872,789,1377,497,180,689,185,488,100,134,2079,631,7042,563,356
,1294,58,594,72,76,164,376,194,37883,149,10,1402,1500,45,34,198,1951,
465,300,960,694,178,84,542,334,2783,74,118,107,480,852,199,393,188,21
0,1181,631,1131,317,136,277,270,328,675,148,818,70,94,24,421,464,155,
444,114,734,298,30,30,69,25,20,144,35,111,83,44,616,364,34,53,40,989,
5,20,27,320,50,68,7,20,15,14,1607,64,2,252,0];
x2=[6,51,76,14,9,21,70,13,5,35,35,34,19,6,12,6,18,58,17,27,42,7,8,7,4
1,13,20,25,18,21,26,5,27,10,44,15,40,19,14,23,9,28,9,14,27,14,232,41,
15,23,19,39,14,7,22,22,36,687,5,6,48,30,6,18,30,36,31,19,71,119,44,9,
22,18,69,8,6,7,16,71,14,33,10,22,136,29,13,17,14,23,68,41,61,49,43,5,
13,4,42,39,14,29,16,49,28,12,7,6,6,7,25,10,23,9,10,68,29,5,4,12,93,15
,5,6,76,16,22,10,19,5,4,101,21,2,54,5];
x3=[7,2432,100,26,30,150,336,30,40,47,157,5,96,6,32,17,64,320,32,396,
1706,40,25,9,320,96,192,40,96,128,46,15,320,16,160,96,320,72,25,29,27
,106,8,10,128,85,1204,128,35,64,14,102,30,18,35,53,60,1151,12,7,128,4
16,4,82,34,341,67,22,76,56,44,17,170,40,128,17,28,30,27,97,37,82,7,40
,278,315,3,32,26,48,135,57,192,54,192,3,25,5,64,96,64,42,16,64,36,47,
11,13,20,10,41,8,24,9,42,160,42,10,7,5,224,32,5,6,96,16,64,115,20,2,2
,864,192,4,128,1041];
X=[20,22322,1893,157,148,1007,4424,70,170,500,570,8486,508,61,243,139
,484,2833,608,1060,8672,66,359,88,2246,510,2766,447,606,800,586,5,221
8,151,872,789,1377,497,180,689,185,488,100,134,2079,631,7042,563,356,
1294,58,594,72,76,164,376,194,37883,149,10,1402,1500,45,34,198,1951,4
65,300,960,694,178,84,542,334,2783,74,118,107,480,852,199,393,188,210
,1181,631,1131,317,136,277,270,328,675,148,818,70,94,24,421,464,155,4
44,114,734,298,30,30,69,25,20,144,35,111,83,44,616,364,34,53,40,989,5
,20,27,320,50,68,7,20,15,14,1607,64,2,252,0;
6,51,76,14,9,21,70,13,5,35,35,34,19,6,12,6,18,58,17,27,42,7,8,7,41,13
,20,25,18,21,26,5,27,10,44,15,40,19,14,23,9,28,9,14,27,14,232,41,15,2
3,19,39,14,7,22,22,36,687,5,6,48,30,6,18,30,36,31,19,71,119,44,9,22,1
8,69,8,6,7,16,71,14,33,10,22,136,29,13,17,14,23,68,41,61,49,43,5,13,4
,42,39,14,29,16,49,28,12,7,6,6,7,25,10,23,9,10,68,29,5,4,12,93,15,5,6
,76,16,22,10,19,5,4,101,21,2,54,5;7,2432,100,26,30,150,336,30,40,47,1
57,5,96,6,32,17,64,320,32,396,1706,40,25,9,320,96,192,40,96,128,46,15
,320,16,160,96,320,72,25,29,27,106,8,10,128,85,1204,128,35,64,14,102,
30,18,35,53,60,1151,12,7,128,416,4,82,34,341,67,22,76,56,44,17,170,40
,128,17,28,30,27,97,37,82,7,40,278,315,3,32,26,48,135,57,192,54,192,3
,25,5,64,96,64,42,16,64,36,47,11,13,20,10,41,8,24,9,42,160,42,10,7,5,
224,32,5,6,96,16,64,115,20,2,2,864,192,4,128,1041];
beta0=[0.5 1 1 1];
myfun=inline('beta(1)*x(:,1).^3+beta(2)*x(:,2).^3+beta(3)*x(:,3).^3+b
eta(4)','beta','x');


X1=X';
y1=y';
beta=nlinfit(X1,y1,'model',beta0);
 a=beta(1);
 b=beta(2);  %每个元素
 c=beta(3);
 d=beta(4);
 yy=a*x1.^3+b*x2.^3+c*x3.^3+d;
 fc=0;
 for i=1:136
     fc=fc+((a*x1(i).^3+b*x2(i).^3+c*x3(i).^3+d)-y(i)).^2;
 end
 fc=fc/136;
 xx=min(x1):max(x2);
s=sprintf('%f %f %f %f %f\n',beta(1),beta(2),beta(3),beta(4),fc);
disp(s);
```

6.model.m

```
function yy=model(beta0,X)
```

 %一定是两个参数，第一个为系数数组，b(1),b(2),…b(n) %分别代表每个系数，而第二个参数代表所有的自变量， %是一个矩阵，它的每一列分别代表一个自变量。

```
 a=beta0(1);
 b=beta0(2);  %每个元素
 c=beta0(3);
 d=beta0(4);
 x1=X(:,1);  %每一列
 x2=X(:,2);
 x3=X(:,3);
 yy=a*x1.^5+b*x2.^5+c*x3.^5+d;
```

7.netplot.m

```
function netplot(G)
    n = size(G, 1);
    [x y] = pol2cart((0:n-1)*2*pi/n, 1);
    gplot(G, [x' y'], '-o');
end
```

8.testxy.m

```
y=[5,8925,92,20,2,161,116,40,30,64,480,2947,40,10,25,10,68,746,44,168
,3188,34,56,10,708,192,342,64,40,75,12,15,162,11,22,4,16,66,20,64,36,
108,1,5,197,30,30945,64,94,106,40,252,6,25,30,102,4,11988,3,5,103,587
,1,2,64,298,64,20,64,10,34,12,98,38,344,5,20,60,90,10,20,57,8,50,256,
226,205,44,30,64,200,60,245,60,20,20,278,6,64,26,4,24,65,2,51,10,13,3
0,13,10,64,6,32,6,30,192,22,11,8,2,1,15,5,3,64,20,64,24,1,1,9,1119,15
4,4,84,5073];
x1=[20,22322,1893,157,148,1007,4424,70,170,500,570,8486,508,61,243,13
9,484,2833,608,1060,8672,66,359,88,2246,510,2766,447,606,800,586,5,22
18,151,872,789,1377,497,180,689,185,488,100,134,2079,631,7042,563,356
,1294,58,594,72,76,164,376,194,37883,149,10,1402,1500,45,34,198,1951,
465,300,960,694,178,84,542,334,2783,74,118,107,480,852,199,393,188,21
0,1181,631,1131,317,136,277,270,328,675,148,818,70,94,24,421,464,155,
444,114,734,298,30,30,69,25,20,144,35,111,83,44,616,364,34,53,40,989,
5,20,27,320,50,68,7,20,15,14,1607,64,2,252,0];
x2=[6,51,76,14,9,21,70,13,5,35,35,34,19,6,12,6,18,58,17,27,42,7,8,7,4
1,13,20,25,18,21,26,5,27,10,44,15,40,19,14,23,9,28,9,14,27,14,232,41,
15,23,19,39,14,7,22,22,36,687,5,6,48,30,6,18,30,36,31,19,71,119,44,9,
22,18,69,8,6,7,16,71,14,33,10,22,136,29,13,17,14,23,68,41,61,49,43,5,
13,4,42,39,14,29,16,49,28,12,7,6,6,7,25,10,23,9,10,68,29,5,4,12,93,15
,5,6,76,16,22,10,19,5,4,101,21,2,54,5];
```