水果生产运输成本问题的建模

2021 年 10 月 3 日

目录

1 摘要

题目中 2 个问题可以分为最短路径问题和产销平衡问题，而这两种问题都有成熟的解决方法。附件中给出了城市之间的距离数据和水果店所在城市与水果店编号之间的对应关系，对于最短路径问题，我们利用表一的城市之间距离数据建立了城市距离的邻接矩阵，利用 Floyd 算法进行求解，得到各生产基地与各水果店的最短距离和经过的各城市节点。然后我们利用以上求得的数据对问题 1 建立产销平衡的整数规划，求得最小成本、各生产基地每天的生产总量、从各生产基地到各水果店每天的供应量以及运输该供应量使用的大车和小车数量。

对于问题 2，和问题 1 相同的步骤，我们对存储天数进行了假设，改造了问题 1 的模型，求解方法类似，最后我们得到了类似问题 1 的各变量数据，以及得到了存储天数与平均成本呈现负相关的结论。

关键词：最短路径　Floyd 算法　产销平衡　整数规划

2 问题重述

某水果连锁是一家水果运输和销售的公司，公司在某省县级及以上城镇设立连锁店。公司规模目前拥有 23 家连锁店，以及 3 个生产基地。

问题 1：目前该公司现有 3 个生产基地、23 家连锁店，生产基地设在 16 号和 63 号和 120 号城镇，为 23 家连锁店提供水果，2017 年连锁店的日销售量见附录 2。运输方式有两种：大车可载 5 吨（少于 5 吨按 5 吨计），运输成本为 2.25 元/车公里；小车可载 3 吨（少于 3 吨按 3 吨计），运输成本为 1.5 元/车公里。请你为公司设计生产基地的日生产量与日配送方案。

问题 2：在日生产总量不变的情况下，如果水果在生产基地最多容许保放 5 日，水果每天的存储费用为 0.05 元/公斤。请你为公司重新设计生产（不同生产基地的日产量可能变化）与配送方案，平均每隔几天向连锁店配送一次，可使运输和存储成本最低。

3 问题分析

题目中 2 个问题具有一定的相似性，可以分为最短路径问题和产销平衡问题分别求解。在求解出问题 1 的基础上对其进行相应的改造，问题 2 可以类似地求解。

对于问题 1，这是个典型的产销平衡问题，即在需求量固定的情况下求最小的运输成本 (问题 2 为运输成本和储存成本)，关键在于得到将水果从各个生产基地运输到销售地所需的大车和小车数量，接着需要得到各生产基地的生产总量和发往各销售地的量。其本质是个整数规划问题。而为了求解此问题，需要各城市间距离矩阵，而附件 1 给出了城市之

间的距离，可以使用一定算法求得最短路径，得到生产基地所在城市与水果店所在城市之间的最短距离。

所以我们按照求解最短路径 → 给出路径节点 → 求解产销平衡问题 → 求解问题 2 的带储存天数产销平衡问题的顺序给出解决方案。

4 符号说明

表 4-1: Parameter values

符号	解释	符号	解释
i	生产地序号	j	连锁店序号
A_i	第 i 个生产地城市	B_j	第 j 个销售城市
a_i	第 i 个生产地的生产总量	b_j	第 j 个销售城市的需求总量
x_{ij}	从 i 地运往 j 地的水果量	d_{ij}	i 地到 j 地的最短距离
l_{ij}	从 i 地运往 j 地使用的大车数目	s_{ij}	从 i 地运往 j 地使用的小车数目
y	问题 2 中水果保存的天数	yy	问题 2 中水果真正放置的天数
a_{ik}	问题 2 中第 i 个生产地第 k 天的生产量		

表 4-2: Parameter values

符号	解释
i	生产地序号
j	连锁店序号
A_i	第 i 个生产地城市
B_j	第 j 个销售城市
a_i	第 i 个生产地的生产总量
b_j	第 j 个销售城市的需求总量
x_{ij}	从 i 地运往 j 地的水果量
d_{ij}	i 地到 j 地的最短距离
l_{ij}	从 i 地运往 j 地使用的大车数目
s_{ij}	从 i 地运往 j 地使用的小车数目
y	问题 2 中水果保存的天数
yy	问题 2 中水果真正放置的天数
a_{ik}	问题 2 中第 i 个生产地第 k 天的生产量

5 基本假设

1. 各城市的道路是无向的，它们之间构成无向图；

表 4-3: Add caption

一哈哈哈 哈哈哈	彼此彼此 vcv	股份回购	官方大范 甘迪	有何吩咐	哈哈哈官方	好好干
48	89	5	7	9	11	13
58	856	5	7	3250	4048	4846
57	1623	5	7	85	85	85
56	2390	5	7	1191	1456	1720
55	3157	5	7	1229	1493	1756

2. 从附件 2 的数据可以看出，每个店的需求量都大于 5 吨，因此车是沿着最短路径直达的，而不会出现到达一个店卸货后继续前往下一个店的情况；
3. 保放天数只记录存储在生产基地的天数，发货后水果店无法卖出水果的天数不算作保放天数；
4. 生产基地每天的生产能力是无限的。

6 模型建立求解与结果分析

6.1 最短路径问题模型的求解

6.1.1 模型的建立

该问题本质上是无向图的最短路问题。示意图如下：

还记得开啥会觉得还是大街上的教科书电竞盛典 SD 卡算法等级双击打开富士康几乎都是宽度防守打法放得开的生活会打开哈哈哈可视电话很多事恢复对方是否很多事恢复邯郸市考核很多事恢复好地方很多事恢复的话合适的和很多时候速度很好的恢复好的好好读书恢复好的很好的和防守打法的机会水电费很多事复活甲都是很多事和的话的时候觉得好的时候都会给大家发公司就换个环境很好的和速度还是大家好伏见稻荷大社好好读书恢复

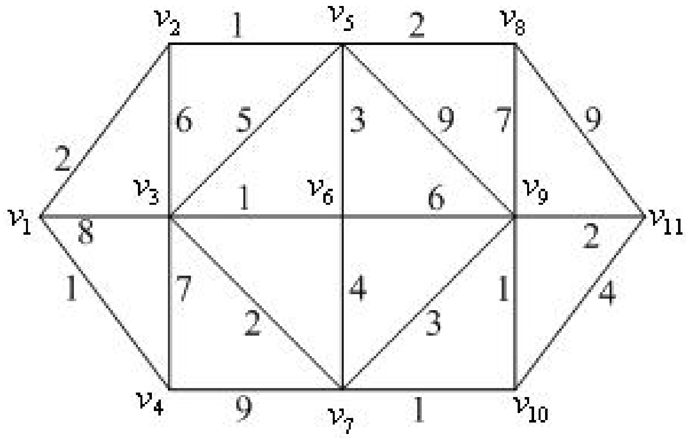

图 1: 示意图：无向图的最短路问题

计算每对顶点之间的最短路径计算赋权图中各对顶点之间最短路径，显然可以调用 Dijkstra 算法。这种算法的时间复杂度为 $O(n^3)$。第二种解决这一问题的方法是由 Floyd R W 提出的算法，称之为 Floyd 算法。假设图 G 权的邻接矩阵 D_0 为：

$$D_0 = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{bmatrix}$$

来存放各边长度，其中：$d_{ii} = 0 \quad i = 1, 2, \cdots, n$;
$d_{ij} = \infty \quad i, j$ 之间没有边，在程序中以各边都不可能达到的充分大的数代替;
$d_{ij} = w_{ij} \quad w_{ij}$ 是 i, j 之间边的长度，$i, j = 1, 2, \cdots, n$ 。对于无向图, A_0 是对称矩阵, $d_{ij} = d_{ji}$ 。
Floyd 算法的基本思想是：递推产生一个矩阵序列 $D_0, D_1, \cdots, D_k, \cdots, D_n$, 其中 $D_k(i, j)$ 表示从顶点 v_i 到顶点 v_j 的路径上所经过的顶点序号不大于 k 的最短路径长度。
计算时用迭代公式:

$$D_k(i, j) = \min\left(D_{k-1}(i, j), D_{k-1}(i, k) + D_{k-1}(k, j)\right)$$

k 是迭代次数, $i, j, k = 1, 2, \cdots, n$。最后，当 $k = n$ 时，A_n 即是各顶点之间的最短通路值。

6.1.2 模型求解

对附件 2 的数据进行预处理，我们发现同一城市可能有多个水果店，城市编号和水果店数量的关系如如图2所示:

所在城镇编号	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
水果店数量	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	2	1	1	1

图 2: 每个城市水果店的数目

可以看到，第 63 号和第 120 号城市有 2 个水果店，为此，我们对在同一城市的水果店的需求量进行了合并，从而有利于后面问题的简化和求解。接着，我们按照水果店所在城市的编号大小进行了重新排序，得到 21 个需求城市的需求量与编号的对应关系如图3所示:

连锁店编号	所在城镇编号	所在城镇排序	日销售量（公斤）
9	1	1	14744
6	10	2	8481
16	11	3	16103
21	16	4	14783
20	22	5	16375
17	24	6	31251
12	27	7	9265
4	31	8	23947
13	34	9	11451
11	36	10	11503
14	42	11	9489
3/18	63	12	50028
23	64	13	22840
7	65	14	15570
8	79	15	38759
15	94	16	12773
2	106	17	38223
1/10	120	18	61250
22	123	19	18081
5	141	20	9258
19	145	21	39653

图 3: 21 个需求城市的需求量与编号的对应关系

有 3 个生产地城市，21 个需求地城市，对附件 1 中给出城市间距离，我们使用 matlab 进行了 excel 数据的读取，建立邻接矩阵，使用 Floyd 算法求解，得到各城市之间最短距离，然后提取的所需要的各生产基地所在城市到水果店所在城市的最短距离和经过的中间节点。4所示：

两两之间经过的节点城市编号列表如图5所示（见下页）, 第 1 列为起始城市也即生产基地所在城市，每行为各路径的节点，每行末尾数为终止城市，也即销售地城市。

销售地 / 生产地	1	10	11	16	22	24	27	31	34	36	42
16	30.67	264.35	336.47	0	326.27	286.26	291.09	270.65	223.18	254.83	214.22
63	187.99	108.36	179.15	157.32	168.95	128.94	135.1	114.66	162.07	193.72	153.11
120	134.31	175.67	239.26	103.64	252.61	218.39	202.41	169.37	119.54	151.19	110.58
销售地 / 生产地	63	64	65	79	94	106	120	123	141	145	
16	157.32	164.63	176.41	181.25	97.78	72.81	103.64	99.72	41.92	38.43	
63	0	7.31	19.09	28.17	190.98	84.51	89.45	94.56	122.56	137.83	
120	89.45	96.76	108.54	117.62	170.17	63.7	0	5.11	61.72	72.85	

图 4: 各生产地城市与需求地城市的最短距离矩阵

6.2 产销平衡问题模型的求解

6.2.1 模型的建立

运输方式有两种：大车可载 5 吨（少于 5 吨按 5 吨计），运输成本为 2.25 元/车公里；小车可载 3 吨（少于 3 吨按 3 吨计），运输成本为 1.5 元/车公里。在满足生产量等于销售量的前提下，使得运输成本尽可能小。可得目标函数为：

$$\min = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(2.25 l_{ij} + 1.5 s_{ij}\right) \cdot d_{ij}$$

约束条件如下：

- 每个产地发往销地的总和等于该产地的生产量;
- 各生产基地发往一个销地的总和等于该销地的需求量;
- 从某产地发往某销地的水果量应该大于 0，且等于大车和小车的运输量的和;
- 大车数量和小车数量应该是整数;

因此，建立如下整数规划模型:

node_1	node_2	node_3	node_4	node_5	node_6	node_7	node_8
16	1						
16	143	10					
16	143	11					
16							
16	143	22					
16	143	24					
16	143	27					
16	143	31					
16	143	34					
16	143	36					
16	143	42					
16	143	63					
16	143	64					
16	143	65					
16	143	79					
16	101	102	94				
16	143	106					
16	143	120					
16	143	123					
16	143	141					
16	143	145					
63	62	108	143	1			
63	6	51	10				
63	6	66	67	68	69	11	
63	62	108	143	16			
63	6	66	67	68	69	22	
63	6	66	67	68	69	24	
63	6	51	27				
63	6	51	31				
63	62	34					
63	62	36					
63	62	42					
63							
63	64						
63	64	65					
63	6	66	79				
63	62	108	94				
63	62	108	106				
63	62	111	112	115	116	120	
63	62	111	112	115	116	120	123
63	62	108	109	131	132	141	
63	62	108	142	145			
120	125	133	141	142	143	1	
120	119	10					
120	119	11					
120	125	133	141	142	143	16	
120	119	22					
120	116	24					
120	119	27					
120	119	31					
120	119	34					
120	119	36					
120	119	42					
120	116	63					
120	116	64					
120	116	65					
120	116	79					
120	125	131	94				
120	125	131	106				
120							
120	123						
120	125	133	141				
120	123	134	139	149	145		

图 5: 两两之间经过的节点城市编号列表

$$\min = \sum_{i=1}^{m}\sum_{j=1}^{n}(2.25l_{ij}+1.5s_{ij})\cdot d_{ij}$$

$$\text{s.t.}\quad\begin{cases}\sum_{j=1}^{n}x_{ij}=a_i\\ \sum_{i=1}^{m}x_{ij}=b_j\\ x_{ij}\geq 0\\ x_{ij}\leq 5000l_{ij}+3000s_{ij}\\ l_{ij},s_{ij}\text{为整数}\\ i=1,2,\dots m\quad j=1,2,\dots n\\ m=3,n=21\end{cases}$$

6.2.2 模型的求解

对模型进行分析，发现该问题变量较多，不适合用 matlab 求解，因此我们使用了 lingo 软件进行计算. 我们得到目标值为 12907.07 元。即最低成本为 12907.07 元，在最优情况下各变量的值如下：

从 $i\to j$ 的发货量 x_{ij} 的值如图6所示

从 $i\to j$ 的大车数目 l_{ij} 的值如图7所示

从 $i\to j$ 的小车数目 s_{ij} 的值如图8所示

销售地 / 生产地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	14744	0	0	14783	0	0	0	0	0	0	0	0	0	0	0	12773	0	0	0	9258	39653
63	0	8481	16103	0	16375	31251	9265	23947	0	0	0	50028	22840	15570	38759	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	11451	11503	9489	0	0	0	0	0	38223	61250	18081	0	0

图 6: 从第 i 个生产地到第 j 销售地的发货量矩阵

销售地 / 生产地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	3	0	0	3	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	2	8
63	0	2	3	0	3	4	2	5	0	0	0	11	4	2	8	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	2	2	2	0	0	0	0	0	8	13	4	0	0

图 7: 从第 i 个生产地到第 j 销售地的所用的大车数量矩阵

生产地\销售地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
63	0	0	1	0	1	4	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0

图 8: 从第 i 个生产地到第 j 销售地的所用的小车数量矩阵

得到的结果是城市与城市之间的发货量，转换为生产基地与各水果店的发货量时，对于一个城市有多个水果店的情况，只需要将发往该城市的货量按照各水果店的需求量进行相应的分配即可，对于一个城市只有一个水果店的情况，则直接发货。

6.3 问题二模型的求解

6.3.1 模型的建立

同问题 1，根据题意，我们假设水果在生产基地存储 y 天
a_{ik} 为第 i 个生产地出货日前第 k 天的生产量, $yy = y + 1$ 表示实际放置的天数 (保放天数加上当天 1 天) 成本应为 (运输成本 + 保放成本)/(保放天数 +1), 所以建立如下整数规划：

$$\min = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(2.25l_{ij}+1.5s_{ij})\cdot d_{ij}+0.05\sum_{i=1}^{m}\sum_{k=1}^{y}a_{ik}}{yy}$$

$$\text{s.t.} \quad \begin{cases} \sum_{j=1}^{n} x_{ij} = \sum_{k=1}^{yy} a_{ik} \\ \sum_{i=1}^{m} x_{ij} = b_j \cdot (y+1) \\ x_{ij} > 0 \\ x_{ij} \le 5000 l_{ij} + 3000 s_{ij} \\ l_{ij}, s_{ij} \text{为整数} \\ yy = y + 1 \\ i = 1, 2, \ldots m \quad j = 1, 2, \ldots n \quad k = 1, 2, \ldots y \\ m = 3, n = 21 \\ y \le 5 \end{cases}$$

6.3.2 模型的求解

使用 Lingo 程序，设置 y 的值分别为 1,2,3,4,5, 得到目标函数值如图9所示：

y	0	1	2	3	4	5
平均成本/元	12907.07	12282.94	12155.72	12113.80	12080.54	12048.14

图 9: 从第 i 个生产地到第 j 销售地的所用的小车数量矩阵

保放天数 y 与平均成本的关系如图10所示

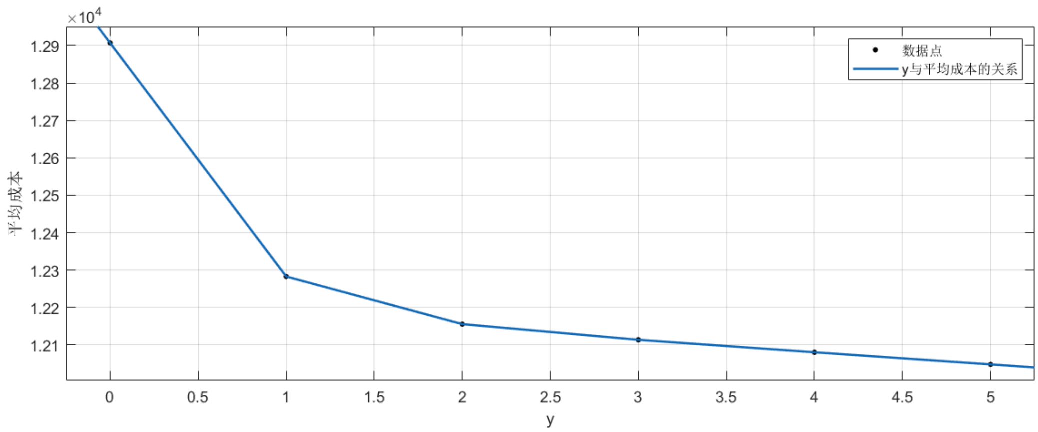

图 10: 从第 i 个生产地到第 j 销售地的所用的小车数量矩阵

由曲线可知，平均成本随着 y 的增大而减小，因此保放 5 天成本最低。
y=1,2,3,4,5 时个生产基地每天的生产量如图11所示

分析 lingo 程序运行结果, 如图11所示，可以得知：无论 y 的值是多少，水果都是当天生产当天发出，也就是说没有保放的水果。

下面是保放 5 天时的各变量数据：

3 处生产基地该天的生产量如图12所示

从 $i \rightarrow j$ 的发货量 x_{ij} 的值如图13所示

从 $i \rightarrow j$ 的大车数目 l_{ij} 的值如图14所示

从 $i \rightarrow j$ 的小车数目 s_{ij} 的值如图15所示

SU(1, 1)	182422.0	0.000000
SU(1, 2)	0.000000	0.2500000E-01
SU(1, 3)	0.000000	0.000000
SU(1, 4)	0.000000	0.000000
SU(1, 5)	0.000000	0.000000
SU(2, 1)	465238.0	0.000000
SU(2, 2)	0.000000	0.2500000E-01
SU(2, 3)	0.000000	0.000000
SU(2, 4)	0.000000	0.000000
SU(2, 5)	0.000000	0.000000
SU(3, 1)	299994.0	0.000000
SU(3, 2)	0.000000	0.2500000E-01
SU(3, 3)	0.000000	0.000000
SU(3, 4)	0.000000	0.000000
SU(3, 5)	0.000000	0.000000

(a) $y = 1$

SU(1, 1)	0.000000	0.1666667E-01
SU(1, 2)	273633.0	0.000000
SU(1, 3)	0.000000	0.1666667E-01
SU(1, 4)	0.000000	0.000000
SU(1, 5)	0.000000	0.000000
SU(2, 1)	0.000000	0.1666667E-01
SU(2, 2)	697857.0	0.000000
SU(2, 3)	0.000000	0.1666667E-01
SU(2, 4)	0.000000	0.000000
SU(2, 5)	0.000000	0.000000
SU(3, 1)	0.000000	0.1666667E-01
SU(3, 2)	449991.0	0.000000
SU(3, 3)	0.000000	0.1666667E-01
SU(3, 4)	0.000000	0.000000
SU(3, 5)	0.000000	0.000000

(b) $y = 2$

SU(1, 1)	0.000000	0.1250000E-01
SU(1, 2)	0.000000	0.1250000E-01
SU(1, 3)	364844.0	0.000000
SU(1, 4)	0.000000	0.1250000E-01
SU(1, 5)	0.000000	0.000000
SU(2, 1)	0.000000	0.1250000E-01
SU(2, 2)	0.000000	0.1250000E-01
SU(2, 3)	930476.0	0.000000
SU(2, 4)	0.000000	0.1250000E-01
SU(2, 5)	0.000000	0.000000
SU(3, 1)	0.000000	0.1250000E-01
SU(3, 2)	0.000000	0.1250000E-01
SU(3, 3)	599988.0	0.000000
SU(3, 4)	0.000000	0.1250000E-01
SU(3, 5)	0.000000	0.000000

(c) $y = 3$

SU(1, 1)	0.000000	0.1000000E-01
SU(1, 2)	0.000000	0.1000000E-01
SU(1, 3)	0.000000	0.1000000E-01
SU(1, 4)	456055.0	0.000000
SU(1, 5)	0.000000	0.1000000E-01
SU(2, 1)	0.000000	0.1000000E-01
SU(2, 2)	0.000000	0.1000000E-01
SU(2, 3)	0.000000	0.1000000E-01
SU(2, 4)	1163095.	0.000000
SU(2, 5)	0.000000	0.1000000E-01
SU(3, 1)	0.000000	0.1000000E-01
SU(3, 2)	0.000000	0.1000000E-01
SU(3, 3)	0.000000	0.1000000E-01
SU(3, 4)	749985.0	0.000000
SU(3, 5)	0.000000	0.1000000E-01

(d) $y = 4$

SU(1, 1)	0.000000	0.8333333E-02
SU(1, 2)	0.000000	0.8333333E-02
SU(1, 3)	0.000000	0.8333333E-02
SU(1, 4)	0.000000	0.8333333E-02
SU(1, 5)	547266.0	0.000000
SU(2, 1)	0.000000	0.8333333E-02
SU(2, 2)	0.000000	0.8333333E-02
SU(2, 3)	0.000000	0.8333333E-02
SU(2, 4)	0.000000	0.8333333E-02
SU(2, 5)	1395714.	0.000000
SU(3, 1)	0.000000	0.8333333E-02
SU(3, 2)	0.000000	0.8333333E-02
SU(3, 3)	0.000000	0.8333333E-02
SU(3, 4)	0.000000	0.8333333E-02
SU(3, 5)	899982.0	0.000000

(e) $y = 5$

图 11: y=1,2,3,4,5 时个生产基地每天的生产量

生产地	生产量/公斤
16	547266
63	1395714
120	899982

图 12: 3 处生产基地该天的生产量

生产地\销售地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	88464	0	0	88698	0	0	0	0	0	0	0	0	0	0	0	76638	0	0	0	55548	237918
63	0	50886	96618	0	98250	187506	55590	143682	0	0	0	300168	137040	93420	232554	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	68706	69018	56934	0	0	0	0	0	229338	367500	108486	0	0

图 13: y=5 时从第 i 个生产地到第 j 个销售地的发货量矩阵

生产地\销售地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	18	0	0	18	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	10	47
63	0	9	19	0	20	37	10	29	0	0	0	61	27	19	46	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	14	14	11	0	0	0	0	0	46	74	22	0	0

图 14: y=5 时从第 i 个生产地到第 j 个销售地的所用的大车数量矩阵

生产地\销售地	1	10	11	16	22	24	27	31	34	36	42	63	64	65	79	94	106	120	123	141	145
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	1
63	0	2	1	0	0	1	2	0	0	0	0	0	1	0	1	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

图 15: y=5 时从第 i 个生产地到第 j 个销售地的所用的小车数量矩阵

7 结果分析

对问题 1 进行灵敏度分析：
问题 1 l_{ij} 和 s_{ij} 的 Reduced Cost 值如图16所示：

可以看到 l_{ij}(大车) 的 Reduced Cost 值大多在 300-700，s_{ij}(小车) 的 Reduced Cost 值大多在 100-400，说明大车的数量变化对结果的影响更大一些，这与实际情况也是相符的。

Variable	Value	Reduced Cost
L(1, 1)	3.000000	69.00750
L(1, 2)	0.000000	594.7875
L(1, 3)	0.000000	757.0575
L(1, 4)	3.000000	0.000000
L(1, 5)	0.000000	734.1075
L(1, 6)	0.000000	644.0850
L(1, 7)	0.000000	654.9525
L(1, 8)	0.000000	608.9625
L(1, 9)	0.000000	502.1550
L(1, 10)	0.000000	573.3675
L(1, 11)	0.000000	481.9950
L(1, 12)	0.000000	353.9700
L(1, 13)	0.000000	370.4175
L(1, 14)	0.000000	396.9225
L(1, 15)	0.000000	407.8125
L(1, 16)	2.000000	220.0050
L(1, 17)	0.000000	163.8225
L(1, 18)	0.000000	233.1900
L(1, 19)	0.000000	224.3700
L(1, 20)	2.000000	94.32000
L(1, 21)	8.000000	86.46750
L(2, 1)	0.000000	422.9775
L(2, 2)	2.000000	243.8100
L(2, 3)	3.000000	403.0875
L(2, 4)	0.000000	353.9700
L(2, 5)	3.000000	380.1375

(a) l_{ij} 的 Reduced Cost 值

Variable	Value	Reduced Cost
S(1, 1)	0.000000	46.00500
S(1, 2)	0.000000	396.5250
S(1, 3)	0.000000	504.7050
S(1, 4)	0.000000	0.000000
S(1, 5)	0.000000	489.4050
S(1, 6)	0.000000	429.3900
S(1, 7)	0.000000	436.6350
S(1, 8)	0.000000	405.9750
S(1, 9)	0.000000	334.7700
S(1, 10)	0.000000	382.2450
S(1, 11)	0.000000	321.3300
S(1, 12)	0.000000	235.9800
S(1, 13)	0.000000	246.9450
S(1, 14)	0.000000	264.6150
S(1, 15)	0.000000	271.8750
S(1, 16)	1.000000	146.6700
S(1, 17)	0.000000	109.2150
S(1, 18)	0.000000	155.4600
S(1, 19)	0.000000	149.5800
S(1, 20)	0.000000	62.88000
S(1, 21)	0.000000	57.64500
S(2, 1)	0.000000	281.9850
S(2, 2)	0.000000	162.5400
S(2, 3)	1.000000	268.7250
S(2, 4)	0.000000	235.9800
S(2, 5)	1.000000	253.4250

(b) s_{ij} 的 Reduced Cost 值

图 16: s_{ij} 和 l_{ij} 的 Reduced Cost 值

8 模型优缺点及改进方向

- **优点：**本文模型较为全面地分析了运输成本的优化问题，建立了从求解最短路径 → 给出路径节点 → 求解产销平衡问题 → 求解问题 2 的带储存天数产销平衡问题等一系列问题的解决方案，
- **缺点：**在算法上还有很多不足，比如求解问题 2 时，Lingo 程序基础不牢，导致求和上限为变量的程序不知道如何编写，导致 y 值只能一个一个地试。
- **改进方向：**我们假设生产基地的每天生产能力为无限，实际上其不可能为无限。如果能够得到生产能力上限的数据，可以对模型进行改进与优化；另外，水果的保存天数应该等于在生产基地和销售地保存天数的总和，如果有水果店的销售时间数据，也可对模型作进一步的改进。

9 附录

A 求最短路径的 matlab 程序

```
clear;warning('off');
n=154; d=zeros(n);
```

```
x=xlsread('1.xlsx');
start=floor(x(:,1));ending=floor(x(:,2));dij=x(:,3);
for i=1:length(start)
    d(start(i),ending(i))=dij(i);
end
d=d+d';
d(d==0)=inf; %把所有零元素替换成无穷
d([1:n+1:n^2])=0; %对角线元素替换成零，Matlab中数据是逐列存储的
[d_all,path]=myfloyd(d);
startcity=[16,63,120];
endcity=[1,10,11,16,22,24,27,31,34,36,42,63,64,65,79,94,106,120,123,141,145];
d_simp=d_all(startcity,endcity);
xlswrite('distance.xlsx',d_simp);
% figure(1);
mypath=cell(length(startcity)*length(endcity),1);
for i=1:length(startcity)
    for j=1:length(endcity)
        sb=startcity(i);db=endcity(j);
        mypath((i-1)*length(endcity)+j,1)={findpath(path,sb,db)};
    end
end
m=cell2table(mypath);
writetable(m,'node.xlsx')

function [dist,mypath]=myfloyd(a)
% 输入：a—邻接矩阵，元素(aij)是顶点i到j之间的直达距离，可以是有向的
% sb—起点的标号；db—终点的标号
% 输出：dist—最短路的距离；% mypath—最短路的路径
n=size(a,1); path=zeros(n);
for k=1:n
    for i=1:n
        for j=1:n
            if a(i,j)>a(i,k)+a(k,j)
                a(i,j)=a(i,k)+a(k,j);
                path(i,j)=k;
            end
        end
    end
end
dist=a;
mypath=path;
end

function mypath=findpath(path,sb,db)
parent=path(sb,:); %从起点sb到终点db的最短路上各顶点的前驱顶点
parent(parent==0)=sb; %path中的分量为0，表示该顶点的前驱是起点
mypath=db; t=db;
while t~=sb
    p=parent(t); mypath=[p,mypath];
    t=p;
end
end
```

B 求解问题一的 lingo 程序

```
sets:
supplys/1..3/: A;
demands/1..21/: B;
links(supplys, demands): x, d, l, s;
endsets

data:
B = 14744 8481 16103 14783 16375 31251 9265 23947 11451 11503 9489 50028 22840 15570 38759 12773 38223 61250 18081 9258 39653;
d = 30.67 264.35 336.47 0 326.27 286.26 291.09 270.65 223.18 254.83 214.22 157.32 164.63 176.41 181.25 97.78 72.81 103.64 99.72 41.92 38.43
187.99 108.36 179.15 157.32 168.95 128.94 135.1 114.66 162.07 193.72 153.11 0 7.31 19.09 28.17 190.98 84.51 89.45 94.56 122.56 137.83
134.31 175.67 239.26 103.64 252.61 218.39 202.41 169.37 119.54 151.19 110.58 89.45 96.76 108.54 117.62 170.17 63.7 0 5.11 61.72 72.85;
@ole(output.xls,'x') = x;!Run Lingo as an administrator and open this excel file;
@ole(output.xls,'l') = l;
@ole(output.xls,'s') = s;
@ole(output.xls,'a') = a;
@ole(output.xls,'b') = b;
@ole(output.xls,'d') = d;
enddata

min = @sum(links(i,j): (2.25*l(i,j)+1.5*s(i,j)) * d(i,j));
@for(supplys(i): @sum(demands(j): x(i,j)) = A(i));
@for(demands(j): @sum(supplys(i): x(i,j)) = B(j));
@for(links(i,j): @gin(l(i,j)));
@for(links(i,j): @gin(s(i,j)));
@for(links(i,j): x(i,j)<=5000*l(i,j)+3000*s(i,j));
end

```

C 求解问题二的 lingo 程序

```
sets:
supplysplace/1..3/: A;
demands/1..21/: B;
savedate/1..5/;
supplys(supplysplace,savedate):su;
links(supplysplace, demands): x, d, l, s;
endsets

data:
B = 14744 8481 16103 14783 16375 31251 9265 23947 11451 11503 9489 50028 22840 15570 38759 12773 38223 61250 18081 9258 39653;
d = 30.67 264.35 336.47 0 326.27 286.26 291.09 270.65 223.18 254.83 214.22 157.32 164.63 176.41 181.25 97.78 72.81 103.64 99.72 41.92 38.43
187.99 108.36 179.15 157.32 168.95 128.94 135.1 114.66 162.07 193.72 153.11 0 7.31 19.09 28.17 190.98 84.51 89.45 94.56 122.56 137.83
```

```
134.31 175.67 239.26 103.64 252.61 218.39 202.41 169.37 119.54 151.19 110.58 89.45 96.76 108.54
    117.62 170.17 63.7 0 5.11 61.72 72.85;
@ole(out2.xls,'su') = su;
@ole(out2.xls,'x') = x;!Run Lingo as an administrator;
@ole(out2.xls,'l') = l;
@ole(out2.xls,'s') = s;
enddata
min = cost/yy;

cost = @sum(links(i,j): (2.25*l(i,j)+1.5*s(i,j)) * d(i,j))+0.05*@sum(supplysplace(i):A(i)-su(i,y));
@for(supplysplace(i):A(i)=@sum(supplys(i,k)|k#le#yy:su(i,k)));
@for(supplysplace(i): @sum(demands(j): x(i,j)) = A(i));
@for(demands(j): @sum(supplysplace(i): x(i,j)) = B(j)*yy);
yy=y+1;
y=5;!y=1,2,3,4,5;
@gin(y);
@for(links(i,j): @gin(l(i,j)));
@for(links(i,j): @gin(s(i,j)));
@for(links(i,j): x(i,j)<=5000*l(i,j)+3000*s(i,j));
end
```

参考文献