水果销售问题的建模与求解

摘要

在生产运输问题的解法中，路径规划与方案组合是解题的核心。本题中，以**图论**中的最短路径算法和整数规划算法为核心方法。可利用迪卡斯特拉算法对题一的数据求出最短路径，再用整数规划求解每个连锁店最优配送方案和最少运费。问题二增加了存储成本这一变量因素，由于最大存储周期较短，故列举不同间隔日期的存储费用和运输开支供直观比较，找到最优方案。最后分析了模型的优缺点，并对缺点进行了进一步的改进和扩展探究。

关键词：图论，迪卡斯特拉算法，整数规划

目录

问题重述

随着现代物流行业发展，控制运输的成本成为经销商和生产者最关心的问题之一，本题正是基于此。本题以水果生产与运输为背景，对运输成本控制问题进行了考察。

问题一是单纯的运输成本控制。给出了生产基地和连锁店所在的城市，要求设计运输方案使得运费最少。此处有两种车，根据每日的销售量选择使用哪种车进行运输使得运费更少。由于运费与车辆数和距离有关，真正的难点在于求解连锁店与基地间的最短距离，需要运用图论知识求解。

问题二在问题一的基础上增加了存储费用，这样可以一次运输多天，可能节省一些成本但是会有一定的存储成本。减少的运输成本和增加的存储成本何时达到平衡将成为解决问题二的关键。

问题分析

（1）问题一分析

问题一的求解首先要观察给出的数据。附件中给出了154座城市和它们的位置坐标，以及城市间道路的连接情况，这样一来就可以作出整个城市的交通网络图。实际上，在求解的过程中会更关心城市间道路的连接情况而并不关心城市的具体位置，所以也只需要描述出城市间的拓扑结构即可。注意到给出的23家连锁店中，16号城市有一家，63号城市有两家，120号城市也有两家。那么这23家连锁店有5家都与生产基地在一起，真正棘手的是余下18家，各自与三家生产基地的距离。求解最短距离的过程，是一个典型的图论问题，这里使用迪卡斯特拉算法求解。解出每一家连锁店距离哪一个生产基地最近，就可以选择这一路径进行运费计算。每一座城市3吨车和5吨车的安排方式和最优运费计算则是一个运筹规划问题，由于车辆数都是整数，就使用整数规划求解。

（2）问题二分析

问题二增加了问题的难度，可以连续输送多日。并设置了水果的存放费用。由于一次性可以运输多日的水果，运费可能就减少了。但是水果存储的费用，会根据销售情况和存放天数变化。不妨以平均每天的成本作为衡量指标。此题中水果最多存放五日，完全可以列举所有的情况进行比较。不一定要把所有城市的水果都按照同一个方案处理，是可以允许不同城市间有方案差异的。

基本假设

1. 不考虑本城市生产基地到本城市连锁店的运输成本和存储费用。
2. 城市的交通网络图与城市的具体坐标无关。
3. 基地到每一个城市的便利店配送的水果配送方案独立，不组合运输。
4. 问题二中仅考虑水果在基地的存储费用不考虑连锁店的存储费用。

符号约定

符号	意义
a,b,u,v	代表图中的顶点
$L_k(u)$	第 k 次迭代过程中 u 到起始点的最短距离
S_k	经过 k 次迭代后带有最小标记的点集合
$w(u,v)$	以 u,v 为端点边的长度
x,y	分别表示三吨车和五吨车数量
$weight$	某店的日销售量（公斤）
M	某店铺水果存储一日产生的存储费用
$[x,y]$	选择方案 x 辆 3 吨车,y 辆 5 吨车

模型建立

（1）问题一的模型建立

首先根据城市的坐标和连接情况可以绘制如图 1 所示的交通网络图：

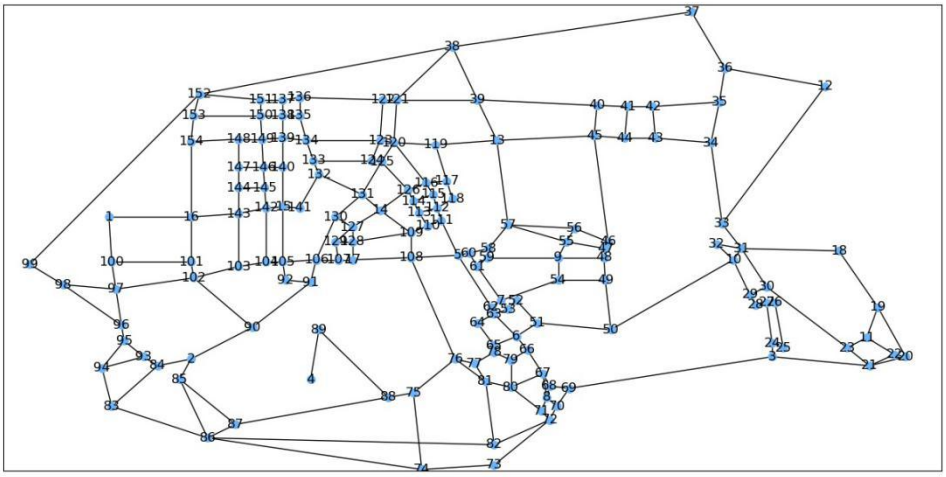

图 1. 绘制的城市网络情况图

那么，正如问题分析中所分析的那样，实际上只要找出 18 家连锁店相对于 3 个生产基地的最短路径，此题便迎刃而解了。最短路径问题是图论问题中最为经典的问题之一，对于本题可选择迪卡斯特拉算法（Dijkstra Algorithm）求解。

迪卡斯特拉算法是用于求解最短路径问题的常用算法之一，其基本思想是以起始点为中心向外层层扩展，直到扩展到终点为止[1]。下面给出迪卡斯特拉算法的具体原理：假设现在欲在图 G 中求解顶点 a 到顶点 b 的距离，并可暂时规定 a 到 a 的距离为 0 而 a 到其它顶点的距离为无穷，即

$$(1).L_0(a)=0, L_0(v)=\infty$$（这里的0表示第0次迭代）

迪卡斯特拉算法求最短路径可以看做一个求顶点序列集的问题[2]：设 S_k 表示经过第 k 次迭代后形成的顶点集合，并令 S_0 为空集。那么集合 S_k 是通过把不属于 S_{k-1} 的带最小标记的顶点 u 添加入 S_{k-1}。即：

$$(2).S_k = S_{k-1} + \{u\}$$

根据贪婪法则，每添加一个顶点 u 进入 S_k，便更新所有不属于集合 S_k 的顶点 v_i 到 a 的最短通路长度 $L_k(v_i)$，这将使得每一步都是最优。设 v 是不属于 S_k 的一个顶点，$L_k(v)$ 是只包含 S_k 中顶点的从 a 到 v 的最短路径长，那么这个递推关系可以描述为式（3）：

$$(3).L_k(a,v)=\min\{L_{k-1}(a,v), L_{k-1}(a,u)+w(u,v)\}$$

其中，w(u,v)是以 u,v 为端点的边的长度。如果不断把每个顶点添加进来，直到添加 b 为止，最后的标记就是 ab 之间最短路径长。

算法流程图如图 2 所示：

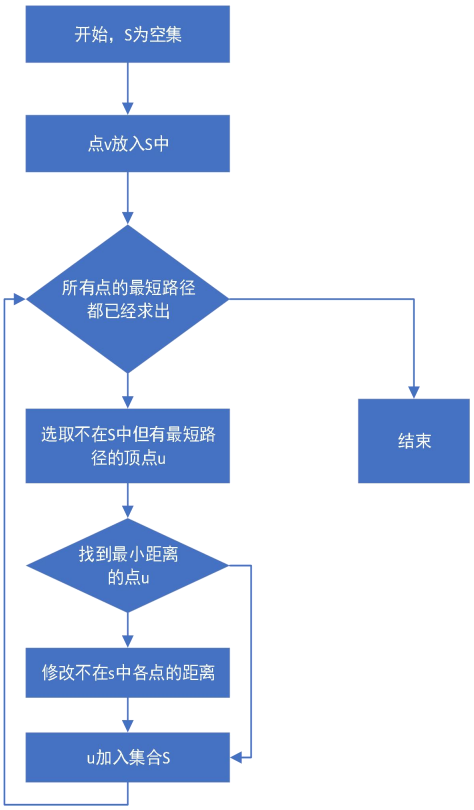

图 2. 迪卡斯特拉算法的流程图

最短路径的求解结果将在模型求解部分的表 1 展示。随即，需要找出每一个连锁店相对于三个基地哪个更近。然后根据日销量选择合适的运输方式。最后求解总生产量时注意，还要包括 16，63，120 三座城市的 5 家连锁店的销售量。

每一座城市的最优派车方案的求解如下：设派出 3 吨车 x 辆，5 吨车 y 辆，某一城市的日销售量为 weight，由于车辆数是整数，就可以把这个问题抽象为一个整数规划的典型形式：

$$\min(1.5x+2.25y), s.t.$$
$$\begin{cases} 3x+5y \geq weight \\ x, y \in N^{+} \end{cases}$$

求解整数规划的算法有很多，例如分支定界法，割平面法，隐式枚举法，匈牙利法和蒙特卡洛方法等。此处对两个经典的算法进行简要介绍：分支定界法和蒙特卡洛方法。这是解决整数规划最经典的两个方法。

1. 分支定界法：这一方法的思想和决策树有着相似之处。主要操作为：将原问题看作一个线性规划问题。求得最优解为 $x_i=b_i$；再通过二分法将原问题分为两个问题:$x_i<=b_i$, 和 $x_i>b_i$；不断这样下去，直到找到令 x_i 全为整数的最优解。筛选则是通过不符合条件（$0<=x_i<=b_i$）或不满足最优而筛。

2. 蒙特卡洛方法：蒙特卡洛方法的思想很简单，此处不妨把它总结为“通过大批量试验结果，以频率估计概率”。蒙特卡洛方法一方面是在求解规划问题（尤其是非线性整数规划）有良好作用，另一方面是在求解函数数值积分时的一个很好的数值方法。核心还是在于随机抽样，蒙特卡洛方法（Monte Carlo Method）本质上是从统计样本对概率分布的特征分析。比如从样本得到经验分别再得到总体分布；或均值估计期望方差等【3】。此方法核心还是在于随机抽样。待估计的目标则可以用随机变量的数学期望估计，

蒙特卡洛方法按照概率独立抽取并接受一批样本，求出它们的期望值就可以作为一个估计数值（当然和精确值可能有偏差）。这就是期望近似。

在求解整数规划的过程中，已经有了现成的规划问题工具包，也有一些专门求解规划问题的软件。所以这里求解过程中也使用了有关工具对整数规划进行求解。

（2）问题二的模型建立

问题二增加了存储成本，可以一次运输多日水果使运费可能减少。由于水果最多存放 5 日，所以可以直接了当使用列举法：间隔 1 天，2 天，3 天，4 天，5 天分别求解并比较，寻找最优解。

假设对于某一家店铺。水果的日销量乘 0.05 得到了存放一日的存放开销 M。如果间隔 1 日送，也就是说一次性送两日份的水果，除掉当天销售的一份，还剩下一份存放一天，这个周期（两日）的存放开销就是 M，平均下来就是每日 M/2 的存储费用；如果隔 2 日送，一次性送了三日份的水果，除掉当天的一份，有两份会存放 1 天。第二天再卖掉一份，还有 1 份又放了 1 天，第三天卖光，这一个周期（三日）存放开销就是 2M+M=3M，平均一日就是 M 的存储费用；如果隔 3 日送，以此类推，以四日为周期的总开销 6M，平均一日 1.5M 存储费用；隔 4 日也很简单，每日 10M/5=2M 的存储费用。间隔 5 日就是 2.5M。隔 5 日就有 3M。

第二是策略问题。如果所有的连锁店采用同一个安排进行运输，很显然，节省下来的运费远远弥补不了存储开支。所以分别对每一个连锁店作出考虑求解最优方案。与问题一的方法类似，计算一次运输的车辆安排方法以及一次运输的开支，然后计算平均每一天能够节省的运费。连锁店每天节省的运费和增加的存储费用比较，看看谁更大一些，然后决定采用隔几天的策略。

从上面求平均存储费用的过程可以看出：隔的天数 T 和存储费用 y 的关系为：

$$y = \frac{T}{2} M$$

这是一个线性增长的开支，但是靠节省运费节省下来的是靠化零为整的想法剪掉的零头，变化不会很大。好在限制最多只能存放 5 天，列举法比较是可行的。如果以每日的平均成本作为衡量指标，对求解结果的评价是比较合理的，也是相对方便求解的。

问题二的列举求解情况将在模型求解的表 4-表 7 中展示。

模型求解

（1）问题一的求解

使用计算机程序对问题一的最短路径进行求解得到最短路径如表 1 所示：

	1	10	11	22	24	27	31	34	36	42	64	65	79	94	106	123	141	145
16	30.67	264.35	336.47	326.27	286.26	291.09	270.65	223.18	254.83	214.22	164.63	176.41	181.25	97.78	72.81	99.72	41.92	38.43
63	187.99	108.36	179.15	168.95	128.94	135.1	114.66	162.07	193.72	153.11	7.31	19.09	28.11	190.98	84.51	94.56	122.56	137.83
120	134.31	175.67	239.26	252.61	218.39	202.41	169.37	119.54	151.19	110.58	96.76	108.54	117.62	170.17	63.7	5.11	61.72	72.85

表 1. 18 座城市的连锁店到三个基地的最短距离

表 1 的第一行是连锁店的所在城市，而第一列是三个基地的城市。每一项就是对应便利店到基地的最短距离。表格中每一列最小值对应的行号就是每一个连锁店销售的水果应该由哪一个基地生产和运输。

再根据每个连锁店的销售量,使用整数规划的方法确定使用何种运输方式更优。运输方式如表 2 所示，其中[a,b]表示使用 a 辆 3 吨车 b 辆 5 吨车：

城市编号	日销量	选择方案	每公里运费	运费
1	14744	[0,3]	6.75	207.02
10	8481	[0,2]	4.5	487.62
11	16103	[1,3]	8.25	1477.99
22	16375	[1,3]	8.25	1393.84
24	31251	[1,6]	15	1934.10
27	9295	[0,2]	4.5	607.95
31	23947	[0,5]	11.25	1289.93
34	11451	[1,2]	6	717.24
36	11503	[1,2]	6	907.14
42	9489	[0,2]	4.5	497.61
64	22840	[1,4]	10.5	76.76
65	15570	[1,3]	8.25	157.49
79	38759	[0,8]	18	505.98
94	12773	[1,2]	6	586.68
106	38223	[0,8]	18	1146.60
123	18081	[0,4]	9	45.99
141	9258	[0,2]	4.5	188.64
145	39653	[0,8]	18	691.74

表 2. 问题一每个城市的日销量，选择方案和运费

（2）问题二的求解

我们把隔 1，2，3，4，5 天的每次配送量也列一个表格。如表 3 所示：

连续	147	848	161	163	3125	926	2394	114	115	948	2284	155	3875	127	3822	180	925	3965
	44	1	03	75	1	5	7	51	03	9	0	70	9	73	3	81	8	3
隔一	294	169	322	327	6250	185	4789	229	230	189	4568	311	7751	255	7644	361	185	7930
天	88	62	06	50	2	30	4	02	06	78	0	40	8	46	6	62	16	6
隔两	442	254	483	491	9375	277	7184	343	345	284	6852	467	1162	383	1146	542	277	1189
天	32	43	09	25	3	95	1	53	09	67	0	10	77	19	69	43	74	59
隔三	589	339	644	655	1250	370	9578	458	460	379	9136	462	1550	510	1528	723	370	1586
天	76	24	12	00	04	60	8	04	12	56	0	80	36	92	92	24	32	12
隔四	737	424	805	818	1562	463	1197	572	575	474	1142	578	1937	638	1911	904	462	1982
天	20	05	15	75	55	25	35	55	15	45	00	50	95	65	15	05	90	65
隔五	884	508	966	982	1875	555	1436	687	690	569	1370	934	2325	766	2293	108	555	2379
天	64	86	18	50	06	90	82	06	18	34	40	20	54	38	38	486	48	18

表 3. 当天和隔 1，2，3，4 天的配送量

下面我们分别对 5 种间隔进行求解：

假设 18 座城市的日销售量加起来一天能够产生的存储费用为 M（可以求出 M=17388.3 元）

1. 间隔 1 天：（一次运输两天的量）然后运输方案可以列为表 4：

城市编号	选择方案	单次派送的运输成本(元/公里)	平均每天的运输成本（元）	每日运输成本降低量（元）	平均每日的存储成本（元）
1	[0,6]	13.5	207.02	0	368.60
10	[1,3]	8.25	446.98	40.64	212.03
11	[1,6]	15	1343.63	134.36	402.58
22	[1,6]	15	1267.13	126.71	409.38
24	[1,12]	28.5	1837.40	96.70	781.28
27	[0,4]	9	607.95	0	231.63
31	[1,9]	21.75	1246.93	43.00	598.68
34	[1,4]	10.5	627.59	89.65	286.28
36	[0,5]	11.25	850.44	56.70	287.58
42	[0,4]	9	497.61	0	237.23
64	[1,9]	21.75	79.50	-2.74	571.00
65	[1,6]	15	143.17	14.32	389.25
79	[1,15]	35.25	495.44	10.54	968.98
94	[1,5]	12.75	623.35	-36.67	319.33
106	[1,15]	35.25	1122.71	23.89	955.58
123	[1,7]	17.25	44.07	1.92	452.03
141	[0,4]	9	188.64	0	231.45
145	[0,16]	36	691.74	0	991.33

表 4. 间隔 1 天节约的运费和增加的存储费用

2. 间隔 2 天：（一次运输三天的量）。

城市编号	选择方案	单次派送的运输成本(元/公里)	平均每天的运输成本（元）	每日运输成本降低量（元）	平均每日的存储成本（元）
1	[0,9]	20.25	207.02	0	737.2
10	[1,5]	12.75	460.53	27.09	424.1
11	[0,10]	22.5	1343.63	134.36	805.2
22	[0,10]	22.5	1267.13	126.71	818.7
24	[0,19]	42.75	1837.40	96.70	1562.6
27	[1,5]	12.75	574.17	33.78	463.3
31	[1,14]	33	1261.26	28.67	1197.4
34	[0,7]	15.75	627.59	89.65	572.6
36	[0,7]	15.75	793.75	113.39	575.1
42	[3,4]	13.5	497.61	4.52	474.5
64	[0,14]	31.5	76.76	8.71	1142.0
65	[1,9]	21.75	138.40	19.09	778.5
79	[1,23]	53.25	498.95	7.03	1938.0
94	[0,8]	18	586.68	0	638.7
106	[0,23]	51.75	1098.83	47.78	1911.2
123	[0,11]	24.75	42.16	3.83	904.1
141	[1,5]	12.75	178.16	10.48	462.9
145	[0,24]	54	691.74	4.50	1982.7

表 5. 间隔 2 天节省的运费和增加的存储费用

3. 间隔 3 天：（一次运输四天的量）。

城市编号	选择方案	单次派送的运输成本(元/公里)	平均每天的运输成本（元）	每日运输成本降低量（元）	平均每日的存储成本（元）
1	[0,12]	27	207.02	0	1105.8
10	[0,7]	15.75	426.67	60.95	636.08
11	[0,13]	29.25	1310.03	167.95	1207.73
22	[1,13]	30.75	1298.80	95.03	1228.125
24	[1,25]	57.75	1861.57	72.53	2343.83
27	[1,7]	17.25	582.62	25.33	694.88
31	[1,19]	44.25	1268.43	21.50	1796.03
34	[1,9]	21.75	650.00	67.24	858.83
36	[1,9]	21.75	822.10	85.04	862.73
42	[1,7]	17.25	476.88	20.73	711.68
64	[1,18]	42	76.75	0	1713
65	[1,9]	21.75	103.80	53.69	1167.75
79	[2,30]	70.5	495.44	10.54	2906.93
94	[1,1]	24	586.68	0	957.98
106	[1,30]	69	1098.83	47.78	2866.73
123	[1,14]	33	42.16	3.83	1356.08
141	[1,7]	17.25	180.78	7.86	694.35

145	[0,32]	72	691.74	4.54	2973.98

表 6. 间隔 3 天节省的运费和增加的存储费用

4. 间隔 4 天：（一次运输五天的量）。

城市编号	选择方案	单次派送的运输成本(元/公里)	平均每天的运输成本（元）	每日运输成本降低量（元）	平均每日的存储成本（元）
1	[0,15]	33.75	207.02	0	1474.4
10	[1,8]	19.5	422.60	65.02	848.1
11	[1,16]	37.5	1343.62	134.36	1610.3
22	[1,16]	37.5	1267.12	126.71	1637.3
24	[1,31]	71.25	1837.39	96.71	3125.1
27	[1,9]	21.75	587.68	20.27	926.5
31	[0,24]	54	1238.33	51.60	2394.7
34	[1,11]	26.25	627.59	89.65	1145.1
36	[1,11]	26.25	793.75	113.39	1150.3
42	[1,9]	21.75	481.02	16.59	948.9
64	[0,23]	51.75	75.66	1.10	2284
65	[1,11]	26.25	100.22	57.27	1557
79	[0,39]	87.75	493.33	12.65	3875.9
94	[0,13]	29.25	572.01	14.67	1277.3
106	[1,38]	87	1108.38	38.22	3822.3
123	[1,18]	42	42.92	3.07	1888.1
141	[1,9]	21.75	182.35	6.29	925.8
145	[0,40]	90	691.74	4.57	3965.3

表 7. 间隔 4 天节省的运费和增加的存储费用

5. 间隔 5 天：（一次运输六天的量）：根据表 4-表 7，节省的运费似乎总是那么多没有多少变化，但是存储成本在线性增长。所以我们可以断定，这里无论哪一家间隔 5 天，运费的减少量都比存储成本小得多。

通过四张表格我们发现，对于任何一座城市的便利店，无论存放几天的水果，平均每日节省下来的运费是远远不能弥补存储开支的。所以，最终得到的方案和原计划一样。我们假设对于每一家连锁店，运输方案都是独立的，不会组合运输。那么显然，最短路径时运费是最少的，而利用整数规划求解的最优方案，也具有高可信度。

结果分析

对于问题一的结果：假设对于每一家连锁店，运输方案都是独立的，不会组合运输。那么显然，最短路径时运费是最少的。而利用整数规划求解的最优方案，也具有高可信度。通过表 1 可以看到如下事实：

1. 由 16 号基地负责生产和运输的连锁店有：1，94，141，145 号。加上 16 号城市自己的一个连锁店，根据日销量数据，16 号基地每日需要生产：91211 公斤。

2. 由 63 号基地负责生产和运输的连锁店有：10，11，22，24，27，31，64，65，79 号。加上 63 号城市自己的两个连锁店，63 号基地每日需要生产：232619 公斤。

3. 由 120 号基地负责生产和运输的连锁店有：34，36，42，106，123 号。加上 120 号城市自己的两个连锁店，120 号基地每日需要生产：148731 公斤。

求解的总运费为：12920.31（元）。

而对于问题二，采取同样的策略，不同的是加入了存储成本。观察表 4-表 7，对于任何一座城市的便利店，无论存放几天的水果，平均每日节省下来的运费是远远不能弥补存储开支的。所以，最终得到的方案和原计划一样。

这一点也很好解释，因为平均每一天运送的重量没有发生变化，是因为一次运送多日使得运费单价产生了“化零为整”的效果。所以，实际上单价减少的并不是很多而且随着间隔天数增长基本维持稳定，而且距离并不长，所以节省的运费并不多。但是存储费用按照公斤为单位，本身存储一天代价就很高昂，加上随着间隔天数增长平均每天的存储费用是线性增长的，所以节约的运费远远不足补齐存储成本。

一次性运输多少天的水果和花费可以绘制这样的曲线图如图 2：可以看到，随着间隔天数从 0 到 4，存储费用线性增加而运费虽然减少但是减少不显著。

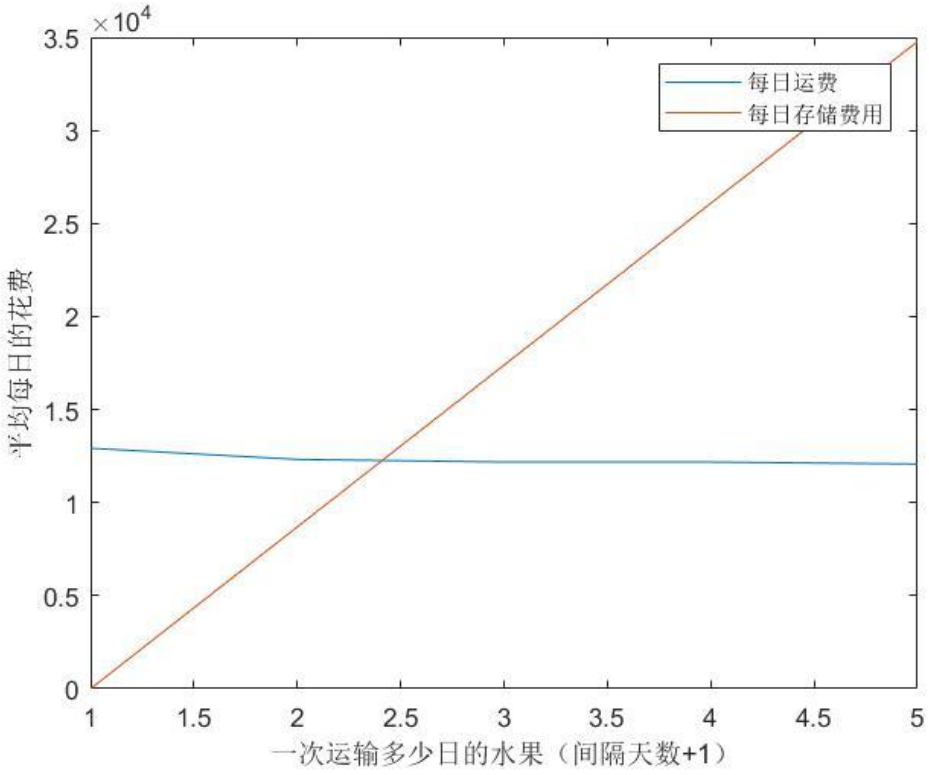

图 2. 一次运输的水果量与花费的图像

模型评价

一．优点

1. 模型明确，有很强的说服力
2. 选择的模型有软件与工具包辅助求解，计算方便
3. 算法思想纯粹，易于理解

二．缺点

1. 这道题的假设是在每个连锁店的运输方案相对独立的情况下讨论的。如果允许不同的连锁店水果在一起运输，情况可能存在更优解。

三．改进

如果去掉相对独立的条件允许一起运输，（也就是说，例如：从 120 到 145 的路上 123 刚好顺路。那能不能把 123 和 145 的货物组合起来一起运输，有没有可能使得组合后运费变小了呢？）我们把顺路的一些城市列出来：

在基地到每个连锁店的最短路径中，106，42，64 和 34 号便利店刚好也在其它便利店的最短路径中。我们可以讨论一下将顺路的城市与其他城市组合运输：

首先考虑如果另外选择新的基地来保证顺路：以 16 号到 10 号为例，如果 106 号和 10 号都由 16 基地输送，整个路径根据编程求得。106 也在路径中，两家一共需要 46704 公斤水果。使用 python 的 cvxpy 整数规划工具包，路径[16, 143.0, 103.0, 104.0, 105.0, 106.0, 107.0, 17.0, 108.0, 5.0, 60.0, 61.0, 7.0, 53.0, 52.0, 51.0, 50.0, 10.0]，需要 1 辆 3 吨 9 辆 5 吨。换而言之，有 2 辆 5 吨车需要跑完从 16 到 10 号的全程而 7 辆 5 吨和 1 辆 3 吨只需要跑到 106 号就可以停止。这样，运送给这两个城市的成本就是：2445.55 元，而按照原计划则 1634.22 元，运费变多了。利用工具把所有可能性列出来只有四种能够同时保证最短距离和顺路：10 号城市位于 27 和 31 号城市的线路上，42 号城市位于 36 号的线路上，64 号在 65 号城市的线路上。如果不能同时保证最短距离这一点，经过列举计算，即使顺路运费也是不会减少的。

其次讨论如果有便利店处于两条路线的交点。我们把十八家连锁店到基地的最短路径列出来：

连锁店编号	负责配送的城市	路线
1	16	[16, 1.0]
10	63	[63, 6.0, 51.0, 50.0, 10.0]
11	63	[63, 6.0, 66.0, 67.0, 68.0, 69.0, 3.0, 21.0, 23.0, 11.0]
22	63	[63, 6.0, 66.0, 67.0, 68.0, 69.0, 3.0, 21.0, 22.0]
24	63	[63, 6.0, 66.0, 67.0, 68.0, 69.0, 3.0, 24.0]
27	63	[63, 6.0, 51.0, 50.0, 10.0, 29.0, 28.0, 27.0]
31	63	[63, 6.0, 51.0, 50.0, 10.0, 31.0]
34	120	[120, 119.0, 13.0, 45.0, 44.0, 43.0, 34.0]
36	120	[120, 119.0, 13.0, 45.0, 40.0, 41.0, 42.0, 35.0, 36.0]
42	120	[120, 119.0, 13.0, 45.0, 40.0, 41.0, 42.0]
64	63	[63, 64.0]
65	63	[63, 64.0, 65.0]
79	63	[63, 6.0, 66.0, 79.0]
94	16	[16, 101.0, 102.0, 97.0, 96.0, 95.0, 94.0]
106	120	[120, 125.0, 131.0, 130.0, 106.0]

123	120	[120,123.0]
141	16	[16, 143.0, 142.0, 15.0, 141.0]
145	16	[16, 143.0, 142.0, 145.0]

可以观察到，10 号城市位于 27 和 31 号城市的线路上，42 号城市位于 36 号的线路上，64 号在 65 号城市的线路上（这也是在第一种情形中列出的情况）。除此以外不存在有便利店在两条路线交点的情况。那我们可以对这几种情况分别进行列举计算。

1. 考虑 10 和 27 组合，原计划要 4 辆 5 吨，现在只需要 3 辆 5 吨 1 辆 3 吨，其中有 2 辆 5 吨要从 63 到 27 而 1 辆 5 吨和 1 辆 3 吨只需要到 10。运费减少了 81.27 元。

2. 考虑 10 和 31 组合，原计划 7 辆 5 吨车，现在只需要 6 辆 5 吨和 1 辆 3 吨，其中 5 辆 5 吨跑到 31，1 辆 5 吨和 1 辆 3 吨仅跑到 10。运费也确实减少了，同样是 81.27 元。

3. 考虑 42 和 36 组合，原计划 1 辆 3 吨 4 辆 5 吨，现在仍然是 1 辆 3 吨 4 辆 5 吨。

4. 考虑 64 和 65 组合，原计划 2 辆 3 吨 7 辆 5 吨，现在 8 辆 5 吨。4 辆 5 吨车跑全程，运费为 237.6 元，原计划只需要 234.25 元，反而增加了。

5. 再考虑 10，31 和 27 组合。现在只需要 1 辆 3 吨 8 辆 5 吨，而原计划需要 9 辆 5 吨。按照组合后的计划，1 辆 3 吨和 1 辆 5 吨只需要跑到 10 号就可以停下，2 辆 5 吨跑到 27 号，5 辆 5 吨到 31 号。这样运费也确实减少了，而且减少的金额和情况 1，2 相等，也是 81.27 元。

那么综合考虑，将 10 号城市的水果和 31 或 27 或与二者一同运输确实能使运费减少，都减少了 81.27 元。

参考文献

[1]. 未曾悟道的佛，迪克斯特拉算法之 MATLAB 实现，https://blog.csdn.net/u013414501/article/details/50506907，访问时间 2020/11/23
[2].K.H.Rosen，《离散数学（第七版）》，北京，机械工业出版社，601 页，2015 年版
[3].李航，《统计学习方法》，北京，清华大学出版社，351-352 页，2019 年版

附录

编写程序：环境：win10+inter i7+anaconda 3.7，python 程序
程序 1. 绘制交通网络图

```
import pandas as pd
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt
G=nx.Graph()

df=pd.read_excel("C:/Users/malia/Desktop/第三题/坐标.xlsx")
node=df.编号
xl=df.x 轴
yl=df.y 轴
length=154
df2=pd.read_excel("C:/Users/malia/Desktop/附件 1：全省交通网络数据.xlsx")

cooderates=[]
edges=[]
G.add_nodes_from(node)
for index,row in df2.iterrows():
    G.add_edge(row.start,row.end,weight=row.distance)
    edges.append((row.start,row.end))
for i in range(length):
    cooderates.append([xl[i],yl[i]])

vnode=np.array(cooderates)
npos = dict(zip(node, vnode))  # 获取节点与坐标之间的映射关系，用字典表示
nlabels = dict(zip(node, node))  # 标志字典，构建节点与标识点之间的关系

nx.draw_networkx_nodes(G, npos, node_size=50, node_color="#6CB6FF")  # 绘制节点
nx.draw_networkx_edges(G, npos, edges)  # 绘制边
nx.draw_networkx_labels(G, npos, nlabels)  # 标签
x_max,y_max = vnode.max(axis=0)  # 获取每一列最大值
x_min, y_min = vnode.min(axis=0)  # 获取每一列最小值
x_num = (x_max - x_min) / 10
y_num = (y_max - y_min) / 10
# print(x_max, y_max, x_min, y_min)
plt.xlim(x_min - x_num, x_max + x_num)
plt.ylim(y_min - y_num, y_max + y_num)
plt.show()
```

程序 2. 求最短路径

```
# -*- coding: utf-8 -*-
"""
@author: mashituo
"""
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_excel("C:/Users/malia/Desktop/附件 1：全省交通网络数据.xlsx")
keystart=[16,63,120]
keyend=[1,10,11,22,24,27,31,34,36,42,64,65,79,94,106,123,141,145]

import networkx as nx
G = nx.Graph()
for index,row in df.iterrows():
    G.add_edge(row.start,row.end,weight=row.distance)

pos = nx.spring_layout(G)
'''
path=nx.dijkstra_path(G, source=1, target=123)
print('节点 1 到 123 的路径：', path)
distance=nx.dijkstra_path_length(G, source=1, target=123)
print('节点 1 到 123 的最短距离为：', distance)
'''
for i in keystart:
    for j in keyend:
        path=nx.dijkstra_path(G, source=i, target=j)
        print(f'节点{i}到{j}的路径：', path)
        distance=nx.dijkstra_path_length(G, source=i, target=j)
        print(f'节点{i}到{j}的最短距离为：', distance)

nx.draw(G,with_labels=True)
plt.show()
```

程序 3.规划最优方案

```
# -*- coding: utf-8 -*-
"""
Created on Sun Nov 22 12:54:43 2020

@author: mashituo
"""
import pandas as pd
df=pd.read_excel("1.xlsx")
keyend=[1,10,11,22,24,27,31,34,36,42,64,65,79,94,106,123,141,145]
for i in keyend:
    for j in range(5):
        df[i][j]=(int(df[i][j]/1000)+1)

import numpy as np
import cvxpy as cp
n=2
c=np.array([1.5,2.25])
a=np.array([[3,5]])
'''
b=np.array([15])
x=cp.Variable(n,integer=True)
objective=cp.Minimize(cp.sum(c*x))
constriants=[0<=x,a*x>=b]
prob=cp.Problem(objective,constriants)
resluts=prob.solve(solver=cp.ECOS_BB)
#输入结果
print(prob.value)#目标函数的值
print(x.value)#各 x 的值
'''
way=[]
score=[]
for index,row in df.iterrows():
    place=[]
    place_s=[]
    for i in keyend:
        b=np.array([row[i]])
        x=cp.Variable(n,integer=True)
        objective=cp.Minimize(cp.sum(c@x))
        constriants=[0<=x,a*x>=b]
        prob=cp.Problem(objective,constriants)
        resluts=prob.solve(solver=cp.ECOS_BB)
        place.append(x.value)
        place_s.append(prob.value)
```

```
        way.append(place)
        score.append(place_s)

way=np.array(way)
score=np.array(score)

cost=[]
distance=np.array([30.67,108.36,179.15,168.95,128.94,135.1,114.66,119.54,151.19,110.58,7.
31,19.09,28.11,97.78,63.7,5.11,41.92,38.43])

for i in range(5):
        cost0=[]
        for j in range(18):
                cost0.append(distance[j]*score[i][j])
        cost.append(cost0)
cost=np.array(cost)
```