生产驱动型的多周期产品配送问题

问题描述

某生产加工企业目前有多个生产基地，生产多种 SKU，不同生产基地可能会生产相同的 SKU，全国有多家经销商负责商品的销售。经销商一周内每天均可以向总公司下订单，总公司在一周的特定时间做结单（比如：周一和周四两次结单）。结单以后，总公司根据所有经销商下单情况，委托第三方物流公司装车发货，第三方物流公司按照车辆类型和车辆数量收取费用。从节省运输成本角度，装车发运目标为所用货车总成本最小。同时每日单辆车的装车发运结果尽量满足最小起发量。

每个生产基地库存分为两部分：在仓库存和计划生产库存。在仓库存就是当前仓库存在的 SKU 库存，随时可以装车。在产库存是要根据排产计划，等到产品生产下线以后才可转为在仓库存。

制定配送计划的时候需要考虑如下因素：（a）最小起运量约束。每种类型的车型均有一个最小起运量，相同类型的车型最小起运量相同，如果实际装车量小于该类型车的最小起运量，产生惩罚；（b）在相邻的结单之间的时间周期内，所有经销商的订单均需要满足；（c）车辆容量上限；（d）供求数量约束。

为了抽象问题的本质，做如下假设：（a）每个基地的可用第三方运输货车数量无上限；（b）运货车辆的容量限制采用一维度量方式；（c）在相邻的结单之间的周期内，基地可以提供的每种 SKU 的数量（在仓库存+计划生产库存总量），均可以满足该周期内，各个经销商的需求。

其他变形版本（这两个条件可以形成新的问题，暂时不考虑，交给后面的学弟学妹做）：

1. 基地库存容量限制
2. 不同日期承运车辆的数量波动尽量小
3. 相同 SKU 尽量装在同一辆车中，不要太分散

数学模型

符号名称	含义说明
集合和下标	
$i \in I$	i 为生产基地编号， I 为生产基地集合
$j \in J$	j 为经销商编号， J 为经销商集合
$k \in K$	k 为 SKU 编号， K 为 SKU 集合
$f \in F$	f 为车辆类型， F 是第三方物流公司提供的车辆类型集合
$t = 0,1,\ldots,T$	t 为当前规划周期， $t = 0$ 表示期初， T 为最大规划周期
问题参数	
v_k	SKU k 的体积量化数值
p_{ikt}	在周期 t 内，生产基地 i 的 SKU k 的生产数量，其中 $t = 1,\ldots,T$
d_{jk}	在所有周期内，经销商 j 对 SKU k 的总需求数量
c_f	车辆类型 f 对应车辆的使用成本
u_f	车辆类型 f 对应车辆的最大装载量
e_f	车辆类型 f 对应车辆的最小起运量
Q_i	生产基地 i 的库存容量上限
决策变量	
s_{ikt}	生产基地 i 中 SKU k 的库存转移到周期 t 的数量，其中 $t = 0,1,\ldots,T$ 。特别地， s_{ik0} 表示生产基地 i 中 SKU k 的期初库存。
x_{ijft}	在周期 t 内，从生产基地 i 到经销商 j 的车辆类型 f 对应车辆的装载方案的使用数量，其中 $t = 1,\ldots,T$ 。
a_{ijkft}	在周期 t 内，从生产基地 i 到经销商 j 的车辆类型 f 对应车辆中装载 SKU k 的商品数量。
p_f	如果车辆类型 f 中装载的 SKU 数量不满足最小起运量，则 $p_f = 1$； 否则 $p_f = 0$

分解思路：采用下料问题的列生成算法思路。

主模型

$$\min \sum_{t=1}^{T}\sum_{i\in I}\sum_{j\in J}\sum_{f\in F} c_f x_{ijft} \tag{1.1}$$

约束条件

$$\sum_{t=1}^{T}\sum_{i\in I}\sum_{f\in F} a_{ijkft} x_{ijft} \ge d_{jk}, \forall j\in J, k\in K \tag{1.2}$$

$$\sum_{f\in F}\sum_{j\in J} a_{ijkft} x_{ijft} = p_{ikt} + s_{ikt-1} - s_{ikt}, \forall i\in I, k\in K, t=1,\dots,T \tag{1.3}$$

$$\sum_{k\in K} s_{ikt} \le Q_i, \forall i\in I, t=1,\dots,T \tag{1.4}$$

约束(1.2)表示，在所有周期内，从所有生产基地运往经销商 j 的 SKU k 的数量，大于等于该经销商的需求。约束(1.3)表示，在周期 t 内，从生产基地 i 运出去的 SKU k 的数量，等于该周期内生产基地 i 生产这种 SKU 的数量，加上生产基地 i 中 SKU k 的库存转移到周期 $t-1$ 的数量，再减去生产基地 i 中 SKU k 的库存转移到周期 t 的数量。约束(1.4)表示，在任意周期 t 内，生产基地 i 中剩余的所有 SKU 数量不得超过该基地的库存容量上限。

令 π^1_{jk} 为约束(1.2)的对偶变量，π^2_{ikt} 为约束(1.3)的对偶变量。在线性规划中，对偶变量表示约束条件的 RHS 每变动一个单位，目标函数值跟着发生变化的数量。假设约束(1.3)的 RHS 增加一个极小的数量 $\delta > 0$，那么要想使得等式成立，约束(1.3)的 LHS 也必须增加 δ，即在周期 t 内，从生产基地 i 运出去的 SKU k 的数量增加了。而在 p_{ikt} 和 s_{ikt-1} 已知的情况下，这会导致 s_{ikt} 的取值减少。

$$\sum_{f\in F}\sum_{j\in J} a_{ijkft} x_{ijft} = p_{ikt} + s_{ikt-1} - s_{ikt} + \delta, \forall i\in I, k\in K, t=1,\dots,T \tag{1.5}$$

采用这种方式感觉有问题的是，约束(1.3)直接放在主模型中，体现不出来对 stair-case 结构的充分利用，并且由于该约束过于复杂，可能会影响求解效率。（目前的办法）

使用大 M 法构造的初始解生成模型如下：

$$\min \sum_{t=1}^{T}\sum_{i\in I}\sum_{j\in J}\sum_{f\in F} c_f x_{ijft} + M_1 \sum_{j\in J}\sum_{k\in K} \bar{\alpha}_{jk} + M_2 \sum_{t=1}^{T}\sum_{i\in I}\sum_{k\in K} \bar{\beta}_{ikt} + M_3 \sum_{t=1}^{T}\sum_{i\in I}\sum_{k\in K} \bar{\gamma}_{ikt} \tag{1.6}$$

约束条件

$$\sum_{t=1}^{T}\sum_{i\in I}\sum_{f\in F} a_{ijkft} x_{ijft} + \bar{\alpha}_{jk} \ge d_{jk}, \forall j\in J, k\in K \tag{1.7}$$

$$\sum_{f\in F}\sum_{j\in J} a_{ijkft}x_{ijft} + \overline{\beta}_{ikt} - \overline{\gamma}_{ikt} = p_{ikt} + s_{ikt-1} - s_{ikt}, \forall i\in I, k\in K, t=1,...,T \tag{1.8}$$

$$\sum_{k\in K} s_{ikt} \le Q_i, \forall i\in I, t=1,...,T \tag{1.9}$$

$$\overline{\alpha}_{jk} \ge 0, \forall j\in J, k\in K \tag{1.10}$$

$$\overline{\beta}_{ikt} \ge 0, \forall i\in I, k\in K, t=1,...,T \tag{1.11}$$

$$\overline{\gamma}_{ikt} \ge 0, \forall i\in I, k\in K, t=1,...,T \tag{1.12}$$

$$x_{ijft} \ge 0, \forall i\in I, j\in J, f\in F, t=1,...,T \tag{1.13}$$

子模型

子模型中，锁定变量中的下标i、j、f和t，子模型的个数为$|I|\times|J|\times|F|\times|T|$个。

目标函数

$$\max \sum_{k\in K} \pi^1_{jk} a_{ijkft} + \sum_{k\in K} \pi^2_{ikt} a_{ijkft} - c_f - p_f \tag{1.14}$$

约束条件

$$\sum_{k\in K} v_k a_{ijkft} \le u_f \tag{1.15}$$

$$\sum_{k\in K} v_k a_{ijkft} + p_f M \ge e_f \tag{1.16}$$

$$a_{ijkft} \le \min(d_{jk}, p_{ikt} + s_{ikt-1}), \forall k\in K \tag{1.17}$$

CG 实现细节

和单周期的模型不同的是，多周期的列生成算法模型要考虑库存的动态性。但是在构建模型时，又不能将每个周期割裂开来，因为当前周期的决策必定会影响到下个周期。如果当前周期所求得的不是最优解，那么后续的所有周期都必定不是最优解。所以，不能通过固定当前周期s_{ikt}取值，来求解下一个周期s_{ikt+1}取值的方法解决此问题。

算法设计

对该问题，拟采用自适应大规模领域搜索算法（Adaptive Large Neighborhood Search, ALNS）求解。算法对比方案如下：

（1）Monolithic Model：使用 Gurobi 求解性能较差的大模型。