数据理解

以大东工厂为例，我们一开始猜测是由于 X3 的高度问题会导致 X3 的大量堆积，但是经过数据的统计与分析后我们发现实际上是由于 5 系过长导致无法在一排放置 3、4 或 5 辆，从而使得 5 系库存大量积压。

由于 5 系会导致库存积压，我们在装入火车车厢的时候希望尽可能多的消耗 5 系的产能。我们发现最理想的情况下可以将 5 系与 X3 以 5：5 的比例进行装载。其中上层（A 层）装载 2 辆 5 系和 3 辆 X3，下层由于比上层稍微长一些，可以装载 3 辆 5 系和 2 辆 X3。

我们发现发往各个目的地的车型比例相差不大，但是不同目的地的需求量大有不同，其中成都最多。在我们的算法中，我们给予该差异足够的重视，具体的操作会在算法部分介绍。

我们发现铁西工厂的所有车辆都可以直接装载，不需要从大东工厂调配车辆进行“拼车”。

解题思路

我们认为首先应该将复杂的问题简单化。我们作出了以下的假设或设定来简化问题。我们将该问题根据目的地（成都、西安、郑州）拆解为 3 个半独立的问题。极大程度减少了问题的耦合性。

1. 我们设定由铁西工厂到大东工厂的单向调货运输。

2. 我们设定大东工厂的装车逻辑为 5 系与 X3 装车数量 5：5。

3. 我们设定了转运车型时仅转运 1 系，X1 和 X2。因为 3 系标轴/长轴均有可能会导致超长。

4. 我们设置货车转运时必须攒满 6 辆

我们首先考虑了不转运时的装车方案。然后加入了简单的时频转运机制（每 N 辆车下线时，从铁西往大东转运 6 辆车）。最后，我们基于实时模拟，推演 2 个小时后的库存积压情况，根据预测结果提前转运需求车辆。

算法介绍

每个工厂的每个目的地均有一条车厢队列。每读取一条数据（或者说每有一辆车下线），将该车放置到对应队列的空闲的车厢中。如果没有车厢可以放置，则创建一列新的车厢并放置其中。如果两个工厂的对于同一个目的的装满的车厢数之和为 29 时则发车。

基于上述简易算法进行模拟后，我们发现由于车型下线分布的不均匀和装配火车车厢的车型限制，大量 5 系车型会堆积。针对该缺陷，我们进行了下述改进，引入了转运机制。

我们认为大东工厂对于铁西工厂下线的车辆有“拼车需求”，因此我们对这种“拼车需求”进行了建模仿真。当大东工厂有“拼车需求”时，将铁西工厂最新下线的 1 系、X1 或 X2 装货车，凑够 6 辆发往大东。我们将这 6 辆车的发车时间加上 2 个小时视为其在大东工厂的下线时间，将其户口从铁西迁往大东。

我们认为“拼车需求”的本质是大东工厂无法自行消耗车型库存，导致库存越积越多，体现在我们的模型中为许多未填满的车厢。我们将这些车厢的空位视为大东工厂的“拼车需求”。

改进方向

我们认为未来在进行充分的建模后，可以采用强化学习的方式。

破坏性建议

题目中对于卡车的建模与实际生产环境中背离。在实际生产环境中，空车运输会造成一定的成本，因此应该将空车运输计入成本并鼓励有效率的精准双向调货。

我们目前的题目假设了过低的转运成本，尽管该目的为鼓励通过转运消耗积压库存。未来可以研究出更加复杂的机制，同时保证了转运成本的真实性并惩罚过度的库存积压。