**音锋密集库入库库位分配需求调研**

编制人：黄秋波

补充人：成伟

# 1. 目前存在的问题：

目前针对每一个项目，单独定义存储结构，分为队列式、堆栈式等众多存储结构，且每一个项目的布局各不相同，因此，各个项目开发的代码不相同，具有比较大的差异性，代码复用度较低，维护工作量大。因此，需要设计一个通用的布局和一个通用的算法，能在此基础上定制具体项目的布局，并根据布局自动调整库位分配算法。

# 2. 研究思路

## 2.1 通用布局

由于各项目的布局情况具有以下共性：

1）母车轨道与货道垂直

2）所有货道等宽、母车轨道与货道宽度大致相同

3）如果某些地方被柱子或其它障碍物占住，则占掉的地方为整数个货位。

因此，可将整个仓库看成一个三维矩阵，比如：storage[5][20][30]，表示一个具有5层的仓库，每一层的布局是20\*30，表示有20个货道，每个货道具有30个库位。

具体实现时，以上的布局里还包括轨道、电梯位置、出库点、入库点等，因此，如果需要表示20个货道\*30个库位，并考虑有两根轨道、2个电梯、一个出库点、一个入库点，则可能需要定义的布局为22\*32，表示有22个货道的位置，其中20行用于货道、一行用于电梯、一行用于出库点、入库点，32个库位里，30个是真正的库位，2个是母车的轨道。

在定义时，可以将其中一些列定义成轨道，将其中一些点定义成障碍物、电梯、出库点、入库点等。

对于其中的货位，可定义分区，比如定义成品区、半成品区、回料区、过道区（用于子车穿越，以便将货物送到另一根轨道再运输）、备货区等。对于同一个区，其中的货物可相邻，也可不相邻，甚至在不同层。

需要考虑到的情况包括：如下图，入库点A仅能够分配至库区一、二、三，仅有库区三、四的托盘能够出库至出库点B：



## 2.2 布局可视化

以上描述的通用布局，为方便用户使用，需要在定义时做到可视化：

1）首先，通过定义三维数组的维度，即可在界面上显示一个三维的立体仓库图。

2）可以在任何一个维度上增加或删除一个单位，比如，删除一层、删除一行、删除一列等。

3）可以选择某一层查看平面图，并可以放大缩小进行查看。

4）可以在平面图上定义一个单元格的属性，比如定义为障碍物、轨道或者货道。

5）可以对货道进行分区，并对不同的分区以不同的颜色显示。

6）可以定义货道、轨道、通道上RGV、AGV的运行方向（即是无向图还是有向图）

7）可以定义入库点、出库点、电梯等，并使用有向或无向边连接它们及轨道。

8）可以对有向或无向边定义权重，即距离。

## 2.3 库位分配算法

入库时，需要根据请求分配库位。库位分配分成两类：

1）在入库点提出请求入库一托货物；

2）事先知道一批订单，根据订单先锁定库位，然后在入库点提出请求入库某个订单中的一托货物。

入库时还有一些约束条件：

1）如果占用超过一个货道，则优先分层摆放，如果是同一层，则优先分轨道摆放（使用两个不同的轨道入库）

2）如果分配的是中间的一个货道（两端都连接有轨道），则使用两个轨道各入库其中一半。

3）补满一个货道时，可以让用户设置考虑品规批次（不能混放）或者不考虑品规批次（可以混放）。

4）考虑楼层数与子母车数不配套，并且楼层数和子母车数量均大于1，如9层4套子母车的场景。这时，要优先分配至有子母车的楼层，如果有子母车的楼层没有合适的储位，则优先分配至离子母车近的楼层。

5）综合考虑开辟新货道和填充已开辟货道两种场景。开辟新货道时存在楼层间均分、单层平铺等原则，填充已开辟货道存在货道补满原则。

# 3. 接口设计

根据以上需求，可设计三个接口：

## 3.1 在入库点提出请求入库一托货物

**接口原型：**

StorageLocation GetLocation(string model, string batch, int inPoint, bool canMix, List<int> destArea, List<int> alternativeArea)

**输入参数**：

model：品规

batch：批次

inPoint：入库点

canMix：同一个货道是否可以混合放多品规批次的货物

destArea：指定目标区（重载函数无这个参数，即不指定）

alternativeArea：如果这个list不存在，则只能入到目标区，否则，优先到目标区，无空位则入到备选区。

**输出参数：**

StorageLocation对象包含三个成员：轨道号、货道号、深度

如果返回null，表示无法找到合适的库位

尚存在的问题：

入库任务产生时，如果直接指定巷道深度，则可能导致深度更大的货物后到，从而被另外的货物挡住了道路。

## 3.2 事先知道入库订单：根据订单组锁定库位

**接口原型：**

List<LockLocation> GetLockLocation(List<Order> orders, bool canMix（）, List<int> destArea, List<int> alternativeArea)

**输入参数：**

orders: 订单组，其类型Order包含成员：订单ID（返回参数中包含订单ID，暂时不用）、model（品规）、batch（批次）、amount（托数）

canMix：同一个货道是否可以混合放多品规批次的货物

destArea：指定目标区（重载函数无这个参数，即不指定）；默认为0

alternativeArea：如果这个list不存在，则只能入到目标区，否则，优先到目标区，无空位则入到备选区。

**返回参数：**

LockLocation对象包含成员：lockID、订单ID、入库点列表、轨道号、货道号、深度范围（如1-4，范围深度对用户不可见，对数据库可见）

**接口功能：**

根据传入的订单组和约束条件，搜索出对于每一个订单需要摆放在几个货道，每个货道对应一个lockID。锁定货位同时对数据库更新，标记这些货位已被锁定。对于一个订单可能存在多个lockID，因此，这个订单的货物将被摆放在多个地方。如果是摆放在一个货道，但是从左右两侧分别入库的，也将得到两个lockID。在LockLocation中包含入库点列表，因此，这个订单的货物必须从这些入库点中的一个入库，否则在入库时不能分配到货位。

（解释：这里的入库点列表限定是考虑到物理上的连通，也就是到这个货道只能从这些入库点进，别的入库点不能连通到这个货道）。

## 3.3根据lockID申请库位

**接口原型：**

StorageLocation GetLockLocation(int lockId, int inPoint)

**输入参数：**

lockId：第2个接口中锁定的货位的lockID

inPoint：入库点。

**返回参数：（返回一个深度，目前步骤为到达货道再申请货位深度）**

StorageLocation对象包含成员：轨道号、货道号、深度

如果返回null，表示无法找到合适的库位，比如入库点不正确或者锁定的货位已被占满。

参数补充说明：

物料属性除品规、批次外，还会有货主等属性（即根据货主、品规、批次等唯一确定一种物料），参数要能够扩展。

为了保证接口参数的扩展性，后续可以视需要将所有的输入、输出参数整合到一个类中，作为类的属性，即仅通过一个对象实例作为输入参数。

# 4 仿真模拟

由于库位分配算法需要考虑的因素众多，一时无法将所有因素考虑完整，因此需要一个仿真系统进行模拟，分析算法的性能。

仿真针对第3节定义的三个接口，可以批量自动生成请求，记录入库货物分配情况，并在界面上实时展示货物的分布。也可以由用户自己创建请求，同样记录入库货物分配情况，并在界面上实时展示货物的分布。

可以根据记录的日志分析货位分配算法的性能，为下一步改进算法提供依据。

# 5. 软件要求

1）东华大学提供一个独立的功能模块，软件上与已有系统无耦合度，植入到已有的Solution中，就是添加引用；调用入口则是文档中的接口。

2）数据存储：

共用一套数据库，双方均能够访问对方的数据表，双方均不能够修改对方的数据表。如果各方的表结构不能满足对方的需要，可以采用视图或者临时表以提供数据。

3）开发平台

VS.NET平台，编程语言为C#，ORM为EntityFrameWork，数据查询使用Linq，框架为.Net Framework4.5及以上。

4）音锋现有的WMS系统软件，仅需要对“立体库入库库位分配动态库” 添加引用。WMS系统不需要为了满足或适应“立体库入库库位分配软件”，而做诸如架构性、代码性的调整和整改。

5）软件实现采用通用算法完成。而不是采用接口、实现类、简单工厂模式，不能采用不同的物流规划和入库需求通过不同实现类完成的方式。

# 6. 应达到的技术指标和参数

1）响应时间：分配算法响应时间不超过1秒

2）需要支持和适应不同物流规划设计的入库分配需求，能够满足以上所述约束条件下不同仓储项目、不同物流规划、不同业务流程、不同物流场景下的库位分配。包括且不限于子母车库、堆垛机库（一个货道一至二个储位）、四向穿梭车库等。