**如何使用？**

1. 本程序基于Python编写，首先需要安装所需的库文件：

$ pip install pyymal

$ pip install gurobipy

注：Gurobi是数学规划优化的**商业系统**，您需要购买商业许可或使用学术许可来解锁其全部功能（如求解大规模优化问题）。

1. 库文件就绪后，可在Main.py文件中设置程序运行的参数，各参数意义如下：

|  |  |
| --- | --- |
| **参数** | **意义** |
| verbose | 是否输出详细日志 |
| pathData | 输入数据的路径 |
| SetInitCurrency() | 设定起始货币种类 |
| SetTermCurrency() | 设定目标货币种类 |
| SetInitCurrencyQuantity() | 设定起始货币数量 |
| SetFeeLimit() | 设定手续费限制 |

1. 完成设定后，运行Main.py文件即可开始优化并在控制台输出模型信息与优化结果。

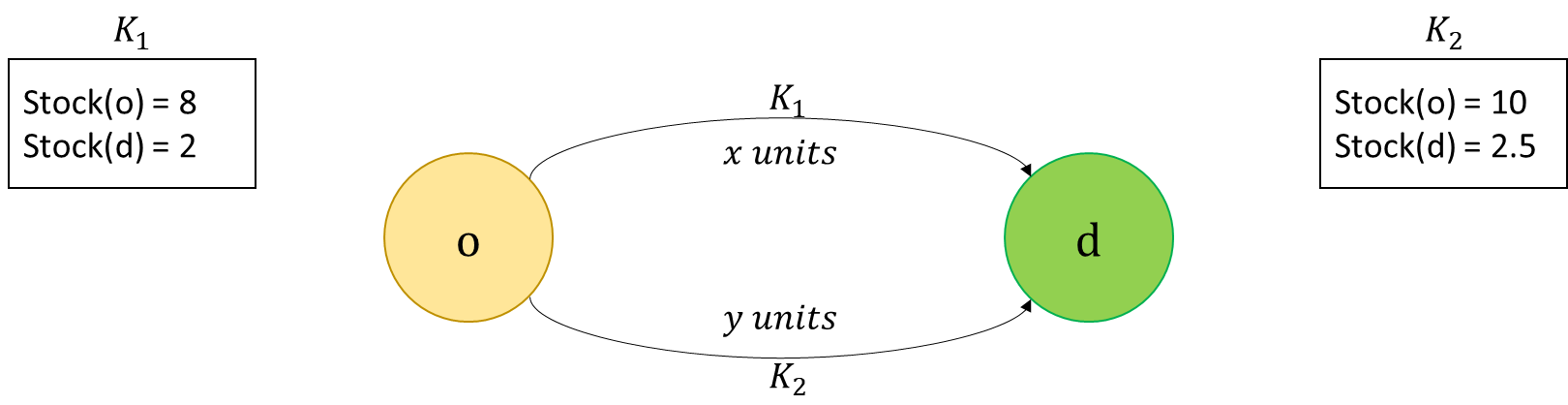
**数据结构解析**

Text

Description automatically generated

数据以yaml格式储存，使用Python读取时会被解析为字典的列表。以右图所示的数据为例，列表中共有两个字典，分别存储两个渠道/交易所的信息。所含信息有交易所/渠道名称(nameExchange)，货币存量(stocks)，交易所/渠道相关手续费(B1)，交易量相关手续费(B2)。手续费的类型为字典的字典，上级字典的键为起始货币种类，下级字典的键为目标货币的种类，值为将起始货币兑换位目标货币所需要的手续费。

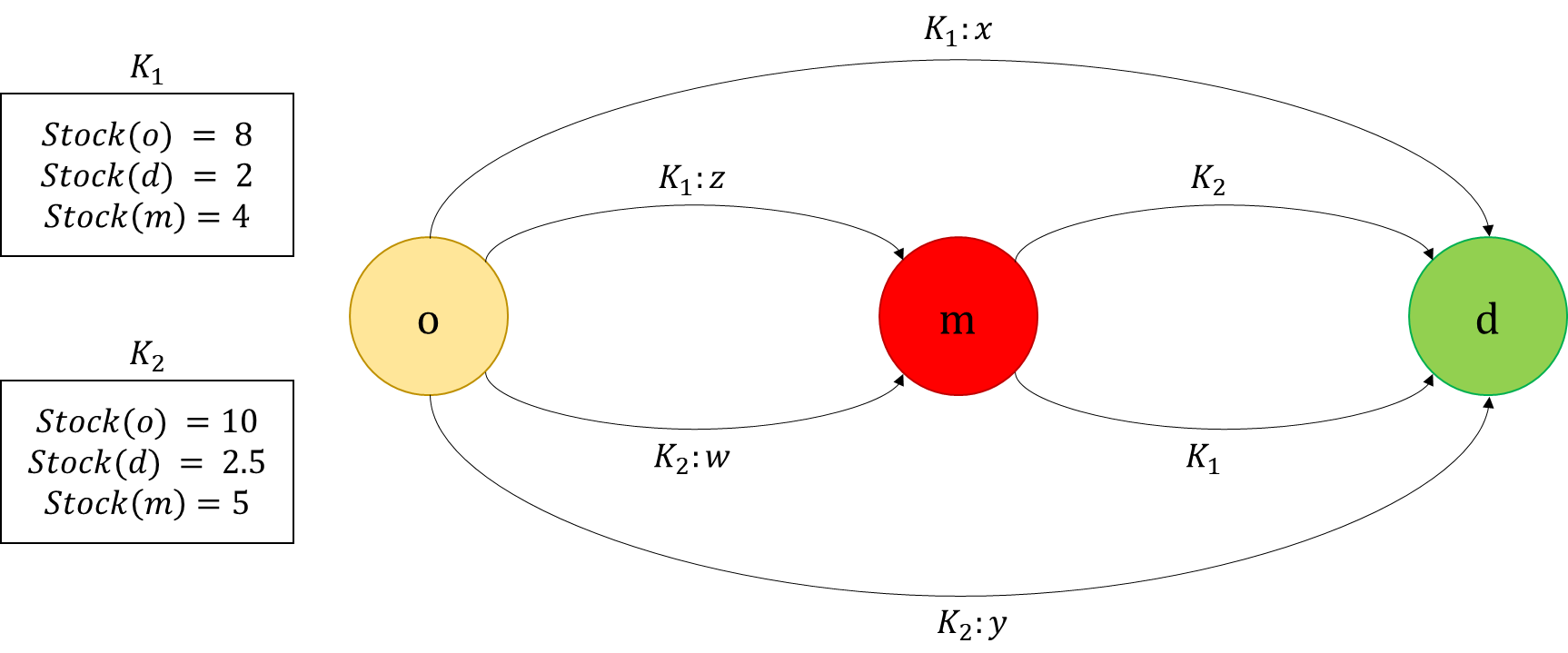
**测试案例1**：仅考虑两个渠道，两种货币，其中的货币存量为，的货币存量为，兑换目标为：将1单位货币全部兑换为货币并最大化所得的货币数量，示意图如下：

**

经程序计算，最优兑换策略为：将0.4477单位和0.5522单位的货币分别经由渠道兑换为货币，总计获得0.2368单位的货币。

**数学推导：**

**测试案例2**：仅考虑两个渠道，三种货币，其中的货币存量为，的货币存量为，兑换目标为：将1单位货币全部兑换为货币并最大化所得的货币数量，示意图如下：

****

经程序计算，使用最优兑换策略总计获得0.2410单位的货币。最优兑换策略如下：

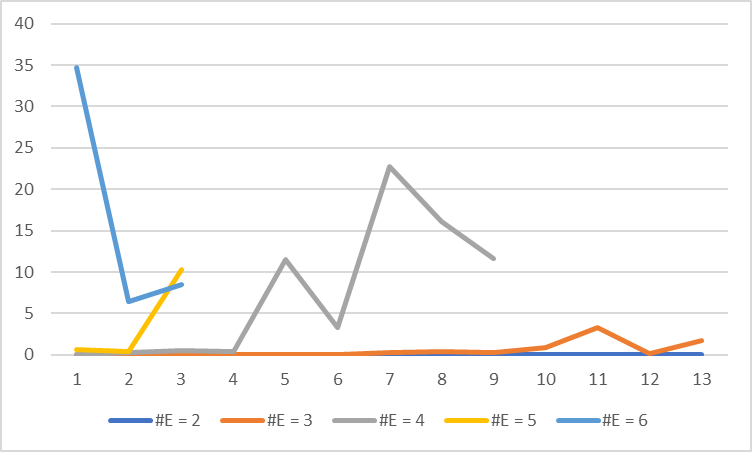
**合理性检查：**本案例相比于案例1引入额外新币种，兑换路径数量增加，且两种货币量保持不变，因而最终收益的货币数量大于案例1

**算法效率分析**

模型的规模主要由货币种类数量和交易所数量决定，下表所示的是两种因素对于求解时间的影响。理论上，货币种类数量对与模型规模的影响更大，因为决策变量的数量和约束的数量与货币种类数量和货币种类数量和交易所数量分别成二次关系和线性关系。



若固定交易所数量，则优化耗时与货币种类数量的关系如下图所示：



设备型号：Dell XPS 15 - 9570

中央处理器：Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz

内存：16 GB

操作系统：Windows 10专业版

解释器版本：Python 3.8.1