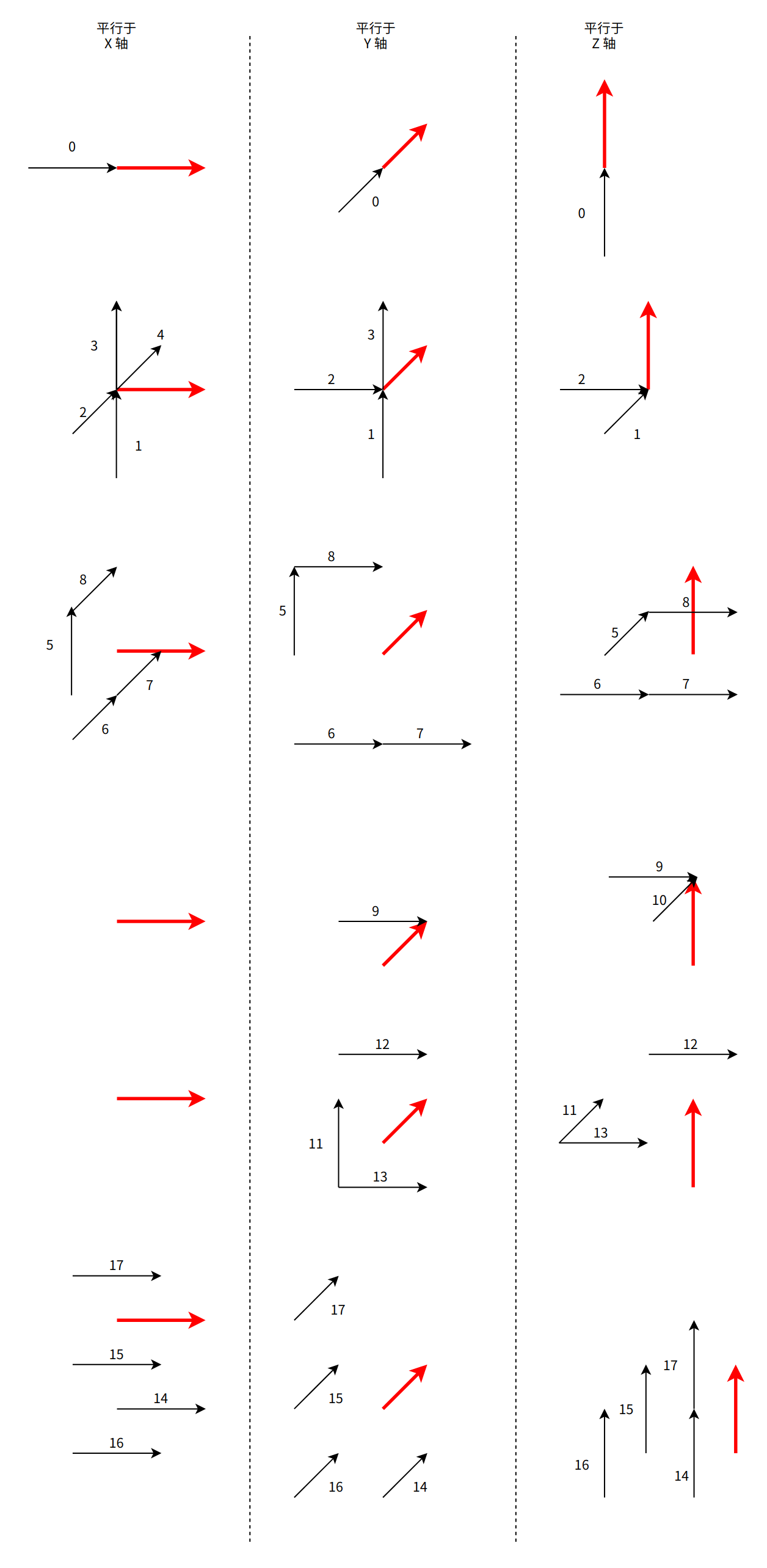
**测18条边熵**



**数据统计**

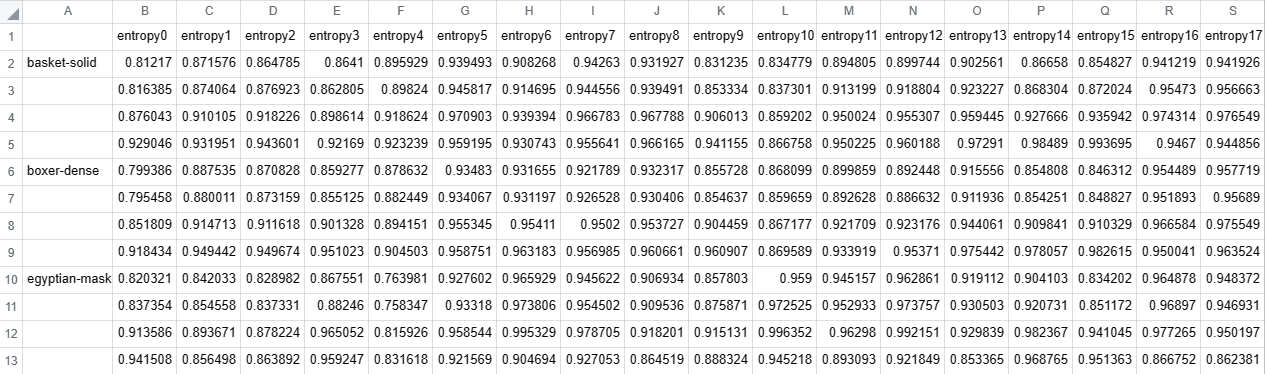
**未分方向测试标识信息熵值**

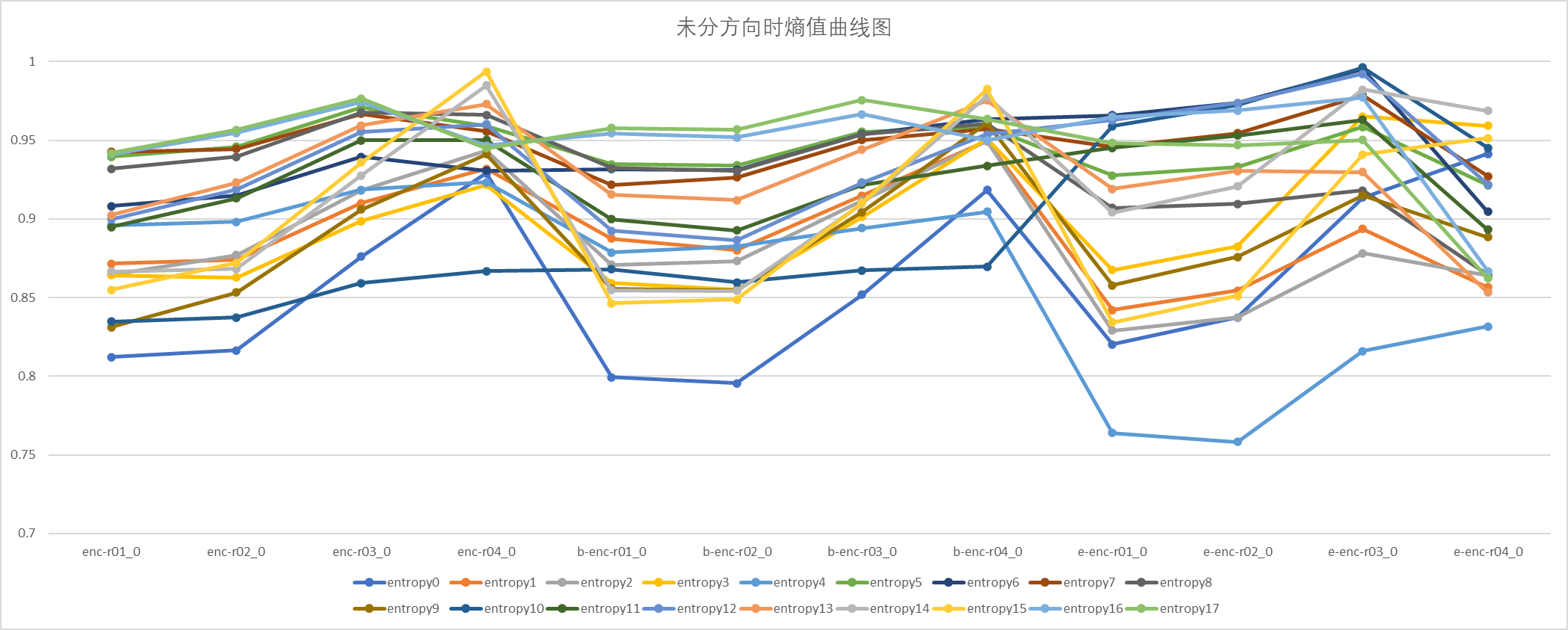
|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 18条边中某条边未被占据 | 18条边中某条边被占据 |
| 当前待编码边未被占据 | 0 | P00 | P10 |
| 当前待编码边被占据 | 1 | P01 | P11 |
|  | entropy | E1 | E2 |

按公式：

其中表示18条边中某条边被占据和未被占据的概率。

每个序列每个码率点求得一个平均值

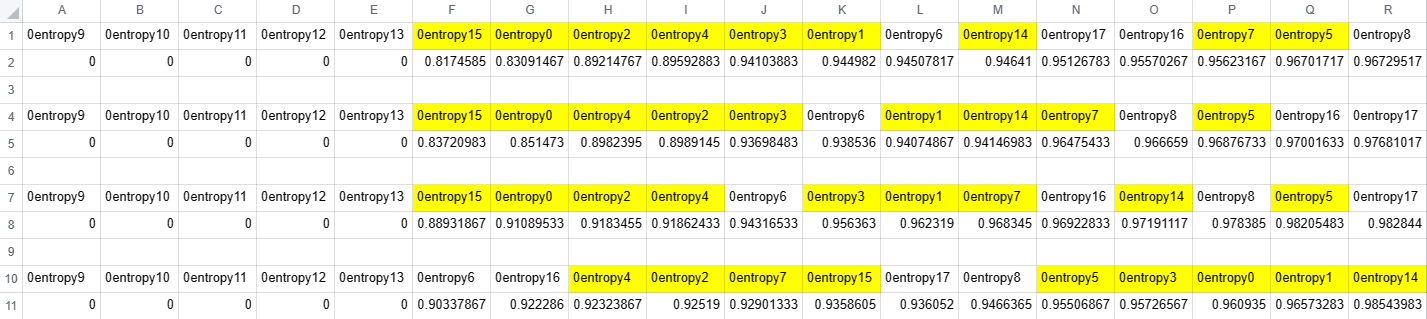




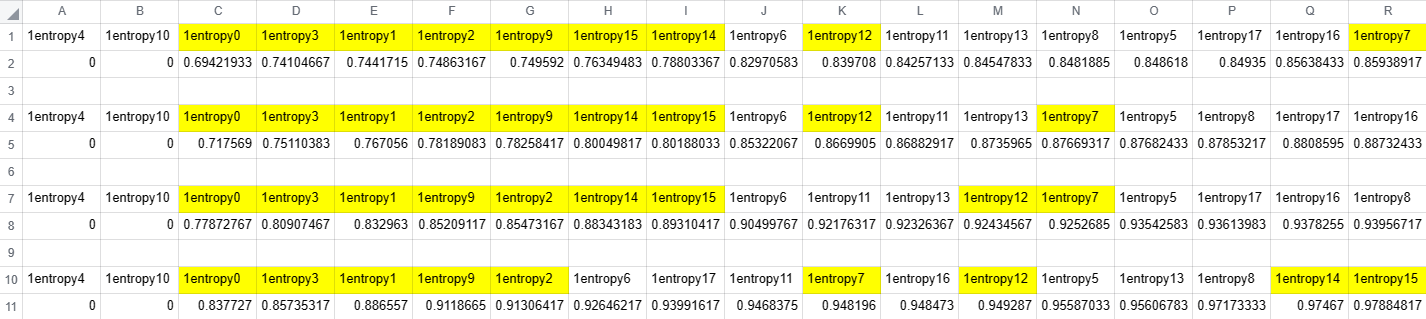
**分方向测试标识信息熵值**

* 标黄表示现有使用的9条边
* 每一行按熵增顺序排列

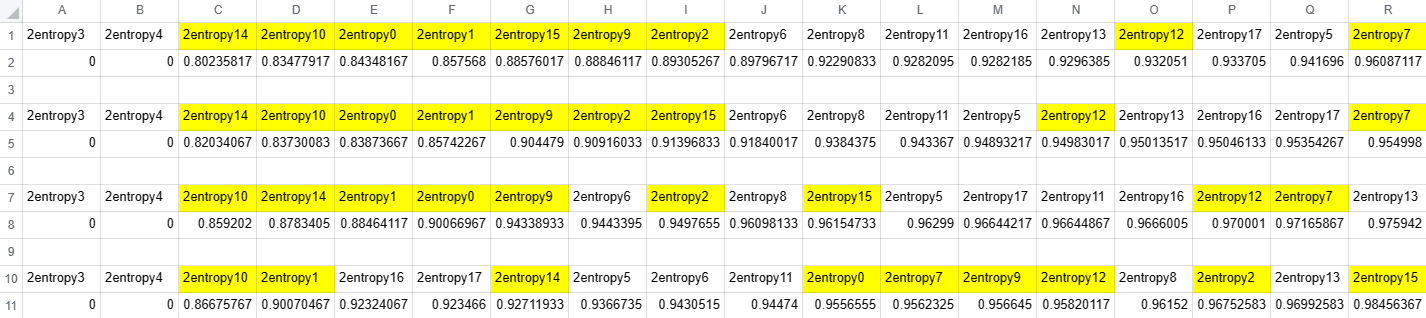
direction=0



direction=1



direction=2



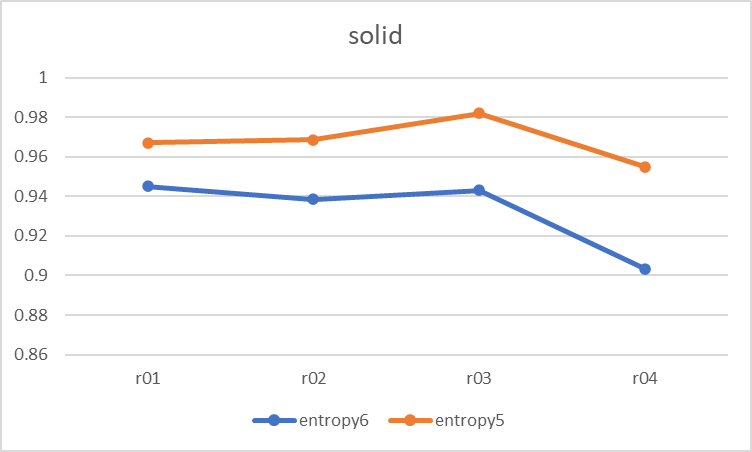


* 基本上是使用熵值较小的边，但最后使用的不相邻的垂直与平行边熵值不是最合适的：

|  |  |  |
| --- | --- | --- |
| direction | 不合适的边 | 合适的边 |
| 0 | 5 | 6 |
| 1 | 7、12 | 11、6 |
| 2 | 7、12 | 6、11 |

* 依据上述数据，我打算修改各个方向使用的边，用熵值较小的替换熵值较大的边，以寻求性能增益。

1. direction=0时——用entropy6替换entropy5



1. direction=1时——用entropy11替换entropy7、用entropy6替换entropy12

|  |  |
| --- | --- |
|  |  |

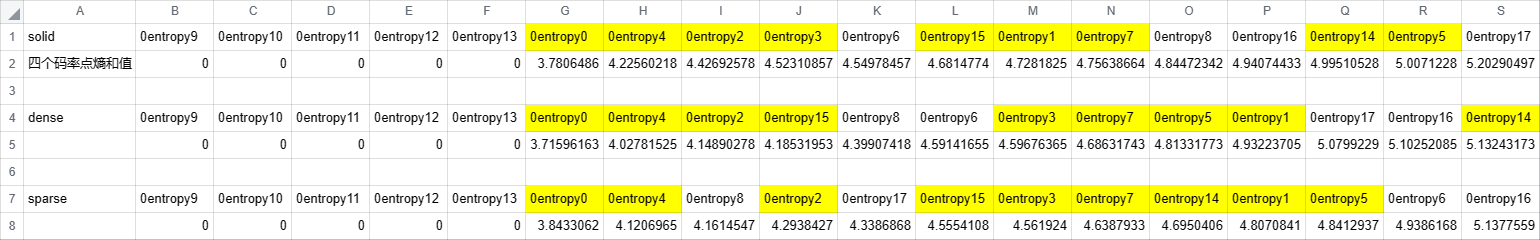
1. direction=2时——用entropy6替换entropy7、用entropy11替换entropy12

|  |  |
| --- | --- |
|  |  |

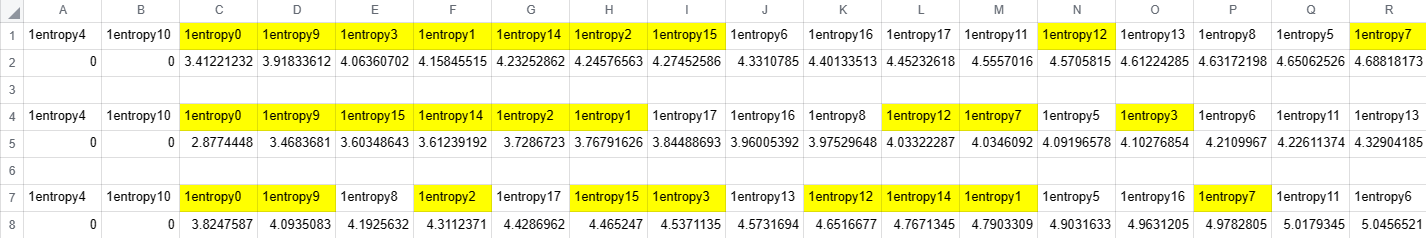
**分方向测试位置信息熵值**

* 与优化标识信息所使用的边流程相同，对位置信息进行熵值计算与分析

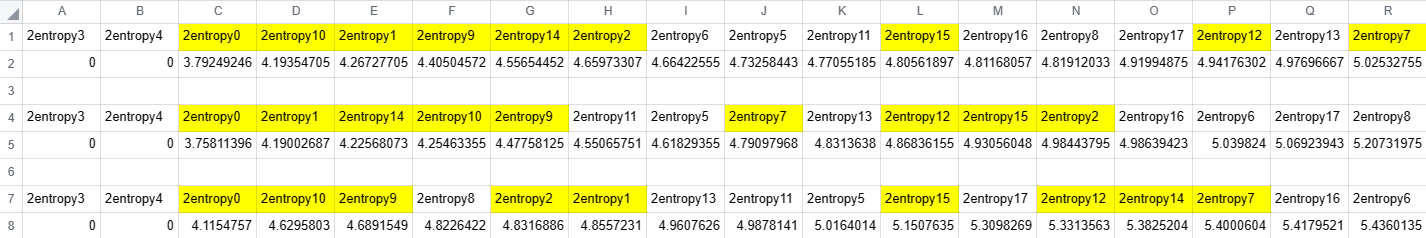
1. direction=0



1. direction=1



1. direction=2



* 分析数据得到以下待修改边与修改边的熵值对比折线图：

1. direction=0——用entropy8替换entropy5

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. direction=1——用entropy17替换entropy7

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. direction=2——用entropy11替换entropy7

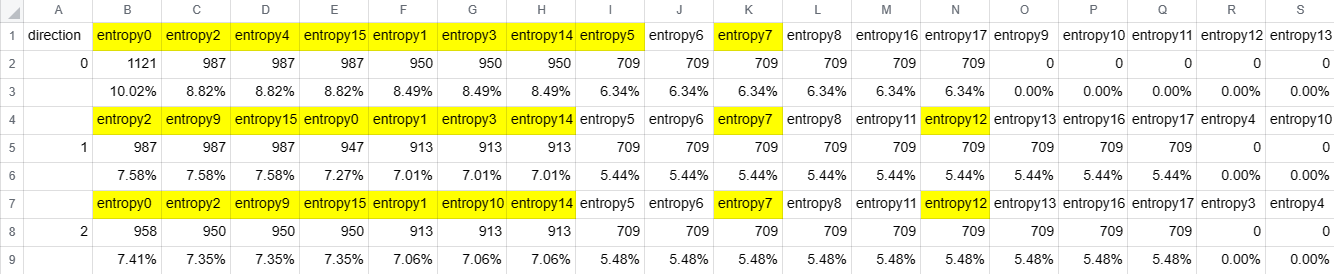
|  |  |  |
| --- | --- | --- |
|  |  |  |

**不同方向上18条边分别存在数量在总存在边数中的占比**

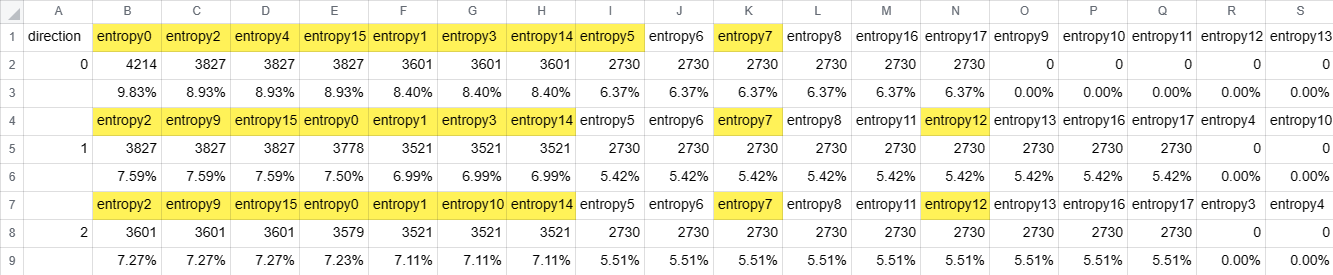
为了考虑各个边在进行上下文构建时的所占权重，我对18条边的存在数量占比进行了统计，替换前熵值较大的边与替换后熵值较小的边，两者在18条边中的存在数量占比是一致的，因此不会因为占比问题影响熵编码效果。

* solid类型basketballplayer序列下测试：

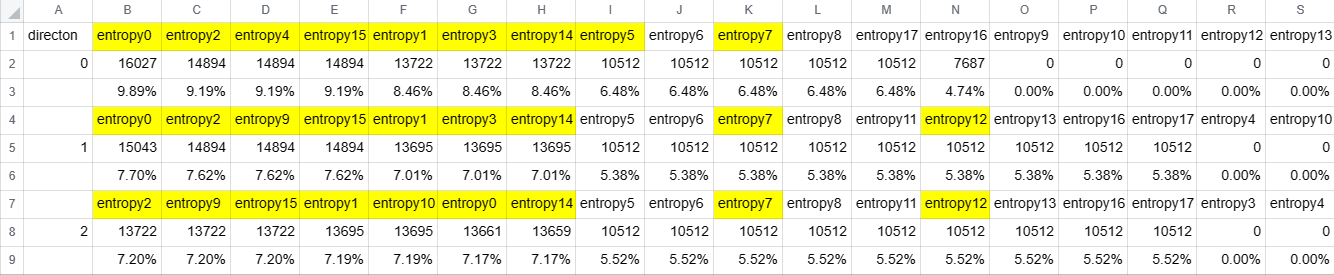
R01码率点



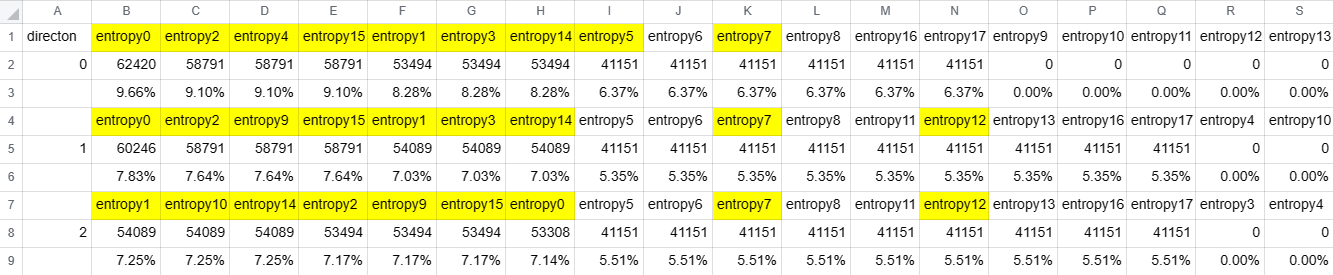
R02码率点



R03码率点



R04码率点



**性能测试**

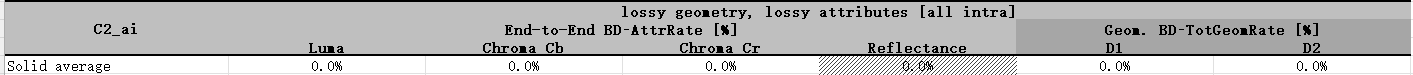
**修改方案**

|  |  |
| --- | --- |
| 标识信息部分：  direction=0——5->6  direction=1——7->6  direction=2——7->6 | 位置信息部分：  5->8  7->17  7->11 |

**测试结果**

solid类型：

|  |  |
| --- | --- |
|  |  |



问题：为什么将使用的边换成熵值较小的边以后，熵编码效率没有得到提高？