## 近似计算——课程作业 陈剑峰 23020161153315

1. **参考论文**

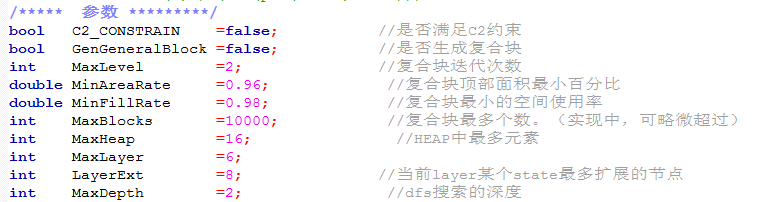
1、求解三维装箱问题的混合模拟退火算法

2、求解三维装箱问题的多层启发式搜索算法

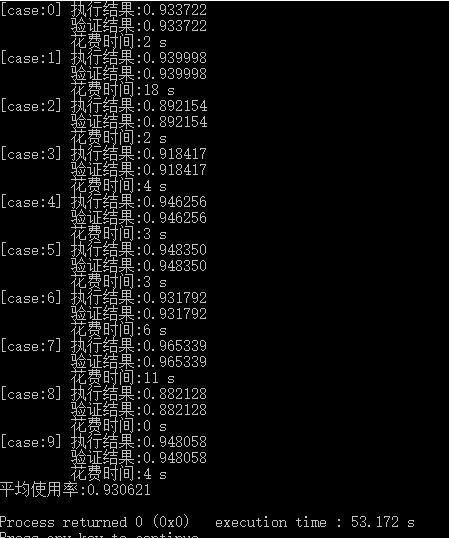
3、a tree search algorithm for the three-dimensional container loading problem

按照以上论文的思路，实现了一下三维装修问题(见代码3d-container.cpp,3d-container.h)，使用方法如下：参数一带上数据的文件名即可。

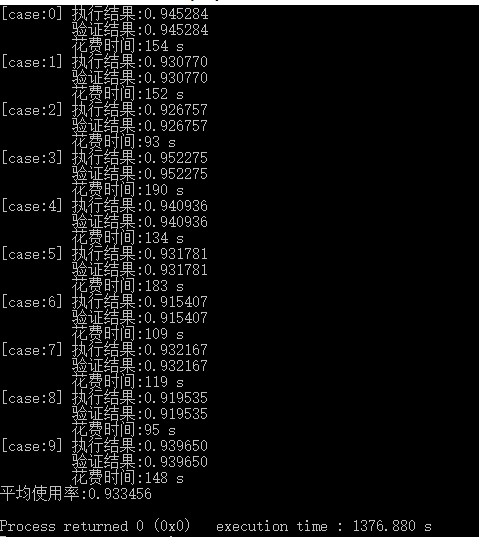
1. **结果(只记录thipack1,7的前10组数据)**

参数为：

1、thpack1



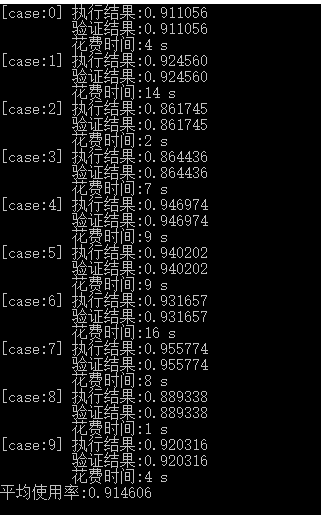
2、thpack7结果



1. 总结

1、不生成复合块时的装载率普遍比生成复合块的装载率更优。

如对于thpack1，生成复合块后的平均装载率从0.930621下降到0.914606(下图)

(thpack1 生成复合块的结果图）

我猜测原因如下：(1)复合块的生成导致block总数非常大，也就是说有相当一部分分支是没有搜索到的。

(2)复合块由不同box组成，搜索的前期也许结果会更优，但是中期，可能会导致剩下的不同box零零碎碎(即同种box没有足够的个数生成一个较大的block，不同的box间又无法生成满足98%空间占有率的block)，然后出现较多的间隙。

2、对于thpack1，即使增加每一层保存的最优解数目/每一个节点扩展到节点数/搜索深度，都很难增加装载率了。

3、我试了一些方法，比如每次取出剩余空间体积最大的那个，比如将block按底面积排序(之前是按照体积排序)，效果都不好。

4、其他还有一些散乱的想法，我暑假有时间再看看能不能优化下。