

Team Project

3D Placement

Report

计 54 李泽龙 2015011321
计 54 贾越凯 2015011335
计 54 陈宇 2015011343

2016 年 6 月

目 录

1 问题描述

给出 n 个长方体的箱子(长、宽、高)，箱子可以随意放置或旋转(棱与坐标轴平行)，但不能有重叠，求能包围所有箱子的长方体的最小体积。

2 算法简介

使用论文 [1] 中的算法，把箱子的放置方案与一棵 T-tree 对应，然后使用模拟退火算法对 T-tree 进行调整，求出最小包围长方体的体积。

2.1 T-tree

2.2 模拟退火

现在，一棵 T-tree 对应了唯一一种放置方案。下面使用模拟退火算法，对 T-tree 进行调整，以求出最小包围长方体的体积。

首先随机生成一棵 T-tree，然后在模拟退火中调整该树，以产生相邻解。调整的方法主要有 3 种：

- Move: 把一个箱子移到另一处；
- Swap: 交换两个箱子；
- Rotate: 旋转一个箱子。

这些调整方法在 T-tree 中对应的操作为：

- Move: 删掉一个结点，插入一个新地方；
- Swap: 交换两个节点所代表的箱子；
- Rotate: 交换该节点箱子的长、宽、高。

由于在模拟退火中是随机产生相邻解，所以要删的点、插入的位置、要交换的点都是随机的。

在模拟退火中，一个可行解的价值

$$\Phi = \alpha V$$

其中 V 是该解对应的放置方案的最小包围长方体的体积。对于相邻解，设其价值为 Φ' ，当前温度为 T ，则接受该解的概率为

$$\begin{cases} 1, & \Phi' < \Phi \\ e^{-\frac{\Phi' - \Phi}{T}}, & \Phi' \geq \Phi \end{cases}$$

3 系统构成

下面是程序中所用的类之间的关系：

3.1 Box

箱子类，包含成员变量 l, w, h (长、宽、高)，以及计算体积的成员函数 `Volume()`，还有用于打开、保存箱子数据的静态成员函数。箱子的列表 `vector<Box>` 定义为 `BoxList` 类型。

3.2 PlacedBox

放好的箱子类，继承于 `Box`，比基类多了 $x, y, z, x2, y2, z2$ 成员变量，分别表示放好的箱子所构成的区域 xyz 坐标的最小值与最大值。放好的箱子的列表 `vector<Placement3D>` 定义为 `PlacedBoxList` 类型。

3.3 Solution

问题的解构成的类，包含一个 `PlacedBoxList` 类型成员变量 `boxes`，还有用于检验解打得合法性的成员函数 `Check()`，求包围所有箱子长方体的体积的成员函数 `GetBoundingBoxVolume()`，计算浪费的空间所占的比例的成员函数 `GetWastedVolumeRatio()` 等。

3.4 Placement3D

类 Placement3D 为求解该类问题提供一个接口, 使用 BoxList 构造, 其中 solve() 函数使用模拟退火调整 T-tree, GetSolution() 函数返回求得的解。剩下一些成员函数用于设置在模拟退火中的参数。

3.5 TTree

类 TTree 实现了在参考文献中描述的 T-tree 的一个框架, 用于实现模拟退火调整过程中对 T-tree 的各种操作, 即 Move(), Swap() 和 Rotate()。其中也实现了若干对 T-tree 插入、删除的函数。

TTree 主要使用一个 BoxList 构造, 构造的结果是随机产生一棵合法的 T-tree。此外还有一个复制构造函数, 实现了深层复制, 用于在模拟退火中初始化相邻解。

TTree 还包含一个纯虚函数 GetSolution(), 用于求解该 T-tree 表示的放置方案。由于论文 [3] 中提到对于一种放置方案, 可以构造多种含义不同的 T-tree, 因此该函数设计为纯虚函数, 由 TTree 类派生的不同含义的 T-tree 类去实现。

3.6 BTTree

该类是 TTree 的一种实现, 即实现了成员函数 GetSolution()。由于计算方案时要把原来三叉的 T-tree 转化为若干棵二叉树, 所以称其为“Binary T-tree”, 也就是 BTTree 类。

在 GetSolution() 中, 会调用该类的成员函数 treePacking() 去对该 T-tree 进行放置, 放置过程中调用 treeDecomposition() 把 T-tree 分解成二叉树与若干和这个二叉树相邻的节点, 然后调用 binaryTreePacking() 对二叉树的每个节点进行放置。placeModule() 则是对单独的一个点进行放置。

4 测试

下面是测试过程, 环境为 Linux(Ubuntu 16.04)。

5 演示程序

为了让数据可视化，我们还用 OpenGL 写了一个 demo，能够实现对问题解的三维显示。

例如，对于论文中的样例，有六个箱子，相应数据存储为 `box.txt`：

```
6
2 3 4
2 3 2
2 2 4
2 2 2
6 3 5
5 1 6
```

在终端输入

```
make
make run DEMO_ARGS="-p box.txt -o sol.txt"
```

等待几秒后，便会给出求得的放置方案的三维显示，并可随意拖动、缩放：

该解会被保存到 `sol.txt` 中，并且终端会输出如下信息：

```
Checking validity: OK
Total number of boxes: 6
Total volume of boxes: 180
Volume of bounding box: 180
Wasted volume: 0
Wasted volume ratio: 0.000%
```

6 参考文献

- [1] Yuh, P.-H., Yang, C.-L., and Chang, Y.-W. 2004. Temporal floorplanning using the T-tree formulation. In Proceedings of International Conference on Computer-Aided Design. 300 – 305.
- [2] Yuh, P.-H., Yang, C.-L., and Chang, Y.-W. 2007. Placement of Defect-Tolerant Digital Microfluidic Biochips Using the T-tree Formulation. ACM Journal on Emerging Technologies in Computing Systems, Vol. 3, No. 3, Article 13.
- [3] Yuh, P.-H., Yang, C.-L., and Chang, Y.-W. 2009. T-Trees: A Tree-Based Representation for Temporal and Three-Dimensional Floorplanning. ACM

Transactions on Design Automation of Electronic Systems, Vol. 14, No. 4,
Article 51.