3DPlacement Documentation

计 53 唐适之 2015011308
计 53 张耀楠 2015011295

目录

Introduction

3DPlacment is a program to place several 3D cubes (or blocks) in an as small as possible volume, using the T-tree algorithm from *Temporal Floorplanning Using the T-tree Formulation [iccad04]*. We used it to experimentally generate a placement that keeps that wasted volume rate less than 10% for maximized number of cubes.

Program Overview

Build and Run

To build the program, go to directory *src*, and execute *make*. It will generate the binary executable *3DPlacement_release*. This program can also be built in debug mode via executing *make debug*, so as to activate correctness verification in every step of the algorithm. Enabling debug mode will not interfere the original build and it will generate *3DPlacement_debug*. One can also build a test program via executing *make test* to get an executable *test*. *test* will do several random modifications on the T-tree to check the correctness.

Run *3DPlacement* with 3 arguments, i.e.

3DPlacement_release <inputFile> <configFile> <outputFile>

to input the 3D cubes from *inputFile* to calculate the minimized placement and output to *outputFile*, using the configurations from *configFile*. For example,

3DPlacement_release blocks.in blocks.conf blocks.out

The format of the *inputFile* is as below: The first line contains the number of the cubes. Each line below contains three numbers indicating the length (on x axis), width (on y axis) and height (on z axis) of that cube. A helper program *gen* should be built at the same time, and can be used to generate *n* random cubes with lengths, widths and heights in range [1,10]. Variable *n* is input from the command line.

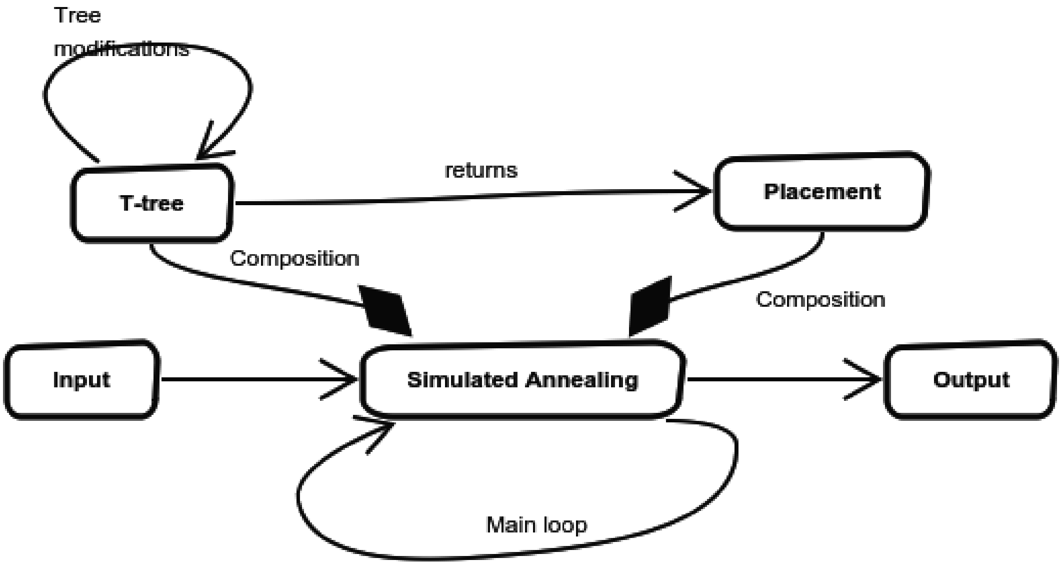
The format of *configFile* can be referred to *testcase/blocks.conf*. This file is able to contain comments do describe the details.

The *outputFile* is of the format below: Each line contains 7 number which describe one cube in the placement. The first number is *id*, which indicates which one it is in the *inputFile*. It counts from zero. 3 following number indicate the coordinates of the nearest lower-left corner of that cube. The last 3 number indicate the current length, width and height of that cube, for cubes can rotate in a placement.

When running *3DPlacement*, it will report some parameters in each 100 steps of the simulated annealing, which make manual optimization easier.

Main workflow

The main workflow can be described in the figure below:



Each loop of the simulated annealing calls a tree modification, and then generated a placement from the tree. The placement is evaluated and decided whether to accept. If the placement is rejected by simulated annealing, the tree modification will be undone.

Detailed reference

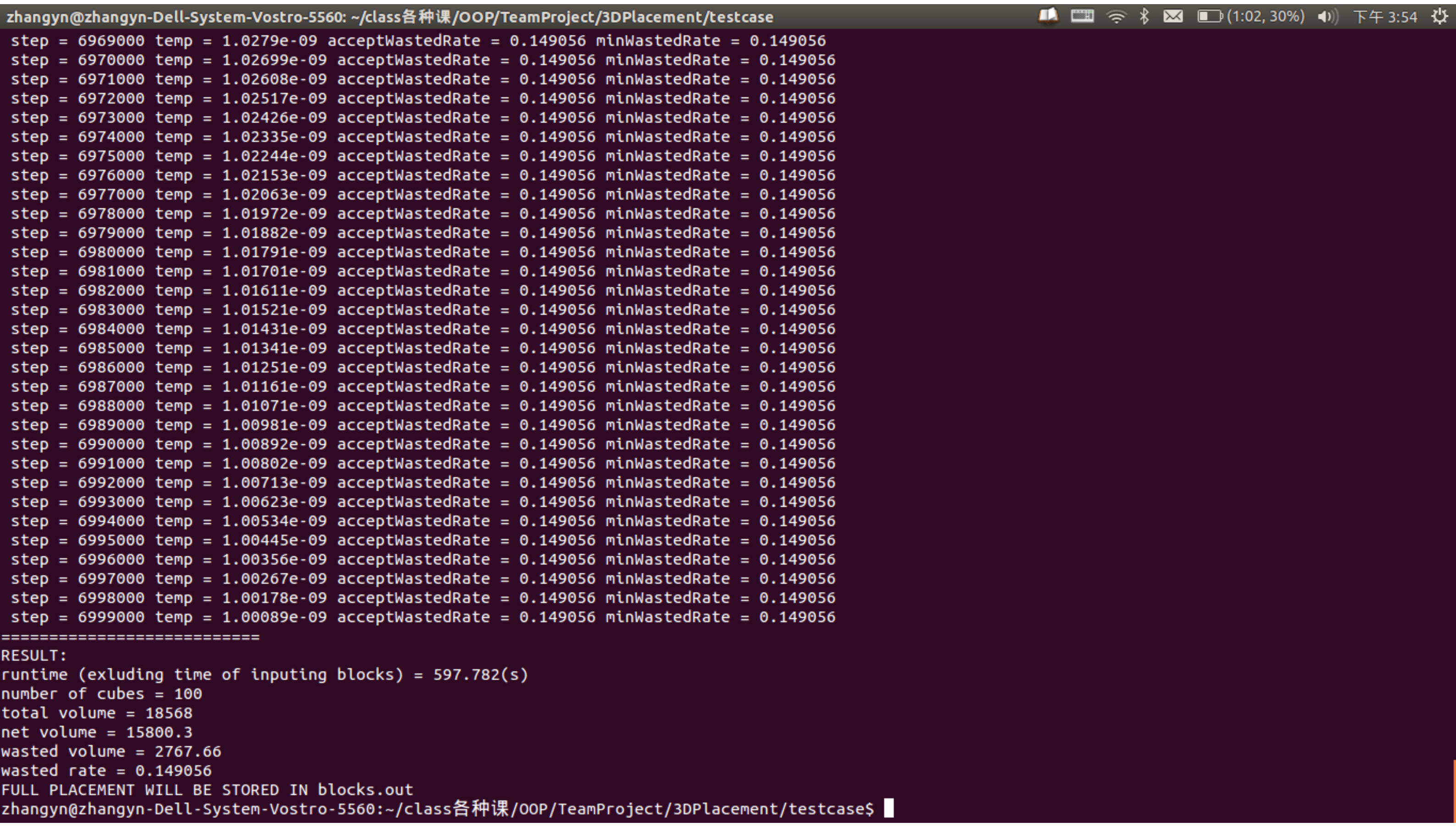
Detailed reference for each classes and functions can be referred to *doc/reference/html/index.html*, which is generated by *Doxygen*.

Experiment Result

In *<configFile>* , five parameters are needed. They are starting temperature, ending temperature, probability to do randomMove, probability to do randomSwap , and the number of how many steps. You can use “#” to write some tag in it.

In ./testcase, You can find three files, blocks.in, blocks.conf and blocks.out , which are *<inputFile> <configFile> <outputFile>* in this experiment.

In this experiment showed int testcase , with starting temperature 0.0000005 and ending temperature 0.000000001 , probability to do randomMove 0.4,probability to do randomSwap , we get the min waste rate 14.9% in 7000000 steps, 597s, with 100 blocks. That's the best result we have got in 10 minutes.



zhangyn@zhangyn-Dell-System-Vostro-5560: ~/class各种课/OOP/TeamProject/3DPlacement/testcase
(1:02, 30%)
下午 3:54

```
 step = 6969000 temp = 1.0279e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6970000 temp = 1.02699e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6971000 temp = 1.02608e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6972000 temp = 1.02517e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6973000 temp = 1.02426e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6974000 temp = 1.02335e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6975000 temp = 1.02244e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6976000 temp = 1.02153e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6977000 temp = 1.02063e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6978000 temp = 1.01972e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6979000 temp = 1.01882e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6980000 temp = 1.01791e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6981000 temp = 1.01701e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6982000 temp = 1.01611e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6983000 temp = 1.01521e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6984000 temp = 1.01431e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6985000 temp = 1.01341e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6986000 temp = 1.01251e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6987000 temp = 1.01161e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6988000 temp = 1.01071e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6989000 temp = 1.00981e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6990000 temp = 1.00892e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6991000 temp = 1.00802e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6992000 temp = 1.00713e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6993000 temp = 1.00623e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6994000 temp = 1.00534e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6995000 temp = 1.00445e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6996000 temp = 1.00356e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6997000 temp = 1.00267e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6998000 temp = 1.00178e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
 step = 6999000 temp = 1.00089e-09 acceptWastedRate = 0.149056 minWastedRate = 0.149056
===========================
RESULT:
runtime (exluding time of inputing blocks) = 597.782(s)
number of cubes = 100
total volume = 18568
net volume = 15800.3
wasted volume = 2767.66
wasted rate = 0.149056
FULL PLACEMENT WILL BE STORED IN blocks.out
zhangyn@zhangyn-Dell-System-Vostro-5560:~/class各种课/OOP/TeamProject/3DPlacement/testcase$ █
```

Followed are some other data when we try other parameters. (next page)

Size	Start T	End T	Steps	Pmove	Pswap	ans1	Time1	ans2	Time2	ans3	Time3	Avr
100	0.005	0.001	10000000	0.2	0.4	0.278	975	0.302	886	0.308	1027	0.296
100	0.005	0.0001	10000000	0.2	0.4	0.188	712	0.201	658	0.213	695	0.201
100	0.005	0.00001	10000000	0.2	0.4	0.225	1018	0.273	1088	0.322	1075	0.273
100	0.05	0.001	10000000	0.2	0.4	0.295	706	0.33	850	0.346	817	0.324
100	0.05	0.0001	10000000	0.2	0.4	0.192	697	0.198	736	0.226	930	0.205
100	0.05	0.00001	10000000	0.2	0.4	0.203	709	0.225	890	0.223	790	0.217
100	0.05	0.000001	10000000	0.2	0.4	0.187	735	0.192	752	0.282	909	0.220
100	0.5	0.001	10000000	0.2	0.4	0.352	712	0.318	857	0.352	712	0.341
100	0.5	0.0001	10000000	0.2	0.4	0.211	818	0.238	941	0.196	624	0.215
100	0.5	0.00001	10000000	0.2	0.4	0.287	855	0.207	717	0.216	831	0.237
100	0.0005	0.0001	10000000	0.2	0.4	0.196	1134	0.177	1161	0.167	1013	0.180
100	0.0005	0.000001	10000000	0.2	0.4	0.178	1140	0.166	927	0.178	1116	0.174
100	0.0005	0.0000001	10000000	0.2	0.4	0.167	1136	0.156	1062	0.176	1140	0.166
100	0.000005	0.000001	10000000	0.2	0.4	0.19	1152	0.18	986	0.175	1087	0.182
100	0.000005	0.0000001	10000000	0.2	0.4	0.17	1136	0.177	1182	0.183	1142	0.177
100	0.000005	0.00000001	10000000	0.2	0.4	0.174	1094	0.167	1131	0.176	1085	0.172
100	5.00E-008	1.00E-010	10000000	0.2	0.4	0.179	1130	0.156	1113	0.19	1080	0.175
100	5.00E-010	1.00E-012	10000000	0.2	0.4	0.194		0.2		0.167		0.187
100	5.00E-003	1.00E-010	10000000	0.2	0.4	0.156		0.18		0.175		0.170
100	5.00E-003	1.00E-008	10000000	0.2	0.4	0.196		0.203		0.17		0.190
100	5.00E-003	1.00E-009	10000000	0.2	0.4	0.17		0.189		0.184		0.181
100	5.00E-003	1.00E-009	10000000	0.4	0.4	0.192		0.256		0.175		0.208
100	5.00E-003	1.00E-009	10000000	0.33	0.33	0.322		0.269				0.197
100	5.00E-004	1.00E-009	10000000	0.2	0.4	0.185		0.186		0.168		0.180
100	5.00E-005	1.00E-009	10000000	0.2	0.4	0.18		0.16		0.17		0.170
100	5.00E-006	1.00E-009	10000000	0.2	0.4	0.186		0.144		0.192		0.174
100	5.00E-006	1.00E-007	10000000	0.2	0.4	0.195		0.183		0.167		0.182
80	5.00E-006	1.00E-007	10000000	0.2	0.4	0.211		0.178		0.194		0.194
80	5.00E-006	1.00E-009	15000000	0.2	0.4	0.161		0.157		0.151		0.156
100	5.00E-005	1.00E-009	10000000	0.4	0.4	0.168		0.173		0.184		0.175
100	5.00E-005	1.00E-009	10000000	0.2	0.2	0.183		0.175		0.166		0.175
100	5.00E-005	1.00E-009	10000000	0.2	0.4	0.198		0.168		0.174		0.180
100	5.00E-006	1.00E-009	10000000	0.2	0.4	0.15		0.165		0.165		0.160
100	5.00E-007	1.00E-009	10000000	0.2	0.4	0.168		0.16		0.182		0.170
100	5.00E-006	1.00E-010	10000000	0.2	0.4	0.182		0.18		0.155		0.172
100	5.00E-006	1.00E-009	10000000	0.2	0.4	0.145	822					
						0.162	823					
						0.164	810					
			8000000			0.168	690					
			7000000			0.163	602					
	5.00E-007	1.00E-009	7000000			0.149	597					
	5.00E-007	1.00E-008	7000000			0.192	606					
	5.00E-007	1.00E-009	7000000			0.16	533					
			8000000			0.166	588					
			10000000			0.158	741					