# 1 Common

全局变量定义和枚举变量

定义了数据内容分布类型和数据访问类型

|  |
| --- |
| #define DATA\_TYPE unsigned char  // 数据内部分布  enum data\_content  **{**  random**,** // 随机分布  standard\_normal**,** // 标准正态分布  poisson**,** // 泊松分布  uniform**,** // 均匀分布  geometric**,** // 几何分布  exponential // 指数分布  **};**  enum access\_mode  **{**  sequential**,** // 顺序访问  step**,** // step 访问  random**,** // 随机访问  standard\_normal**,** // 标准正态分布访问  poisson**,** // 泊松分布访问  geometric**,** // 几何分布访问  exponential // 指数分布访问  **};** |

# 2 Tree类

树结构

|  |
| --- |
| **typedef** struct Node\_st**{**  DATA\_TYPE data**;**  Node**\*** left**;**  Node**\*** right**;**  **}**Node**;**  class Tree  **{**  public**:**  Tree**(**int \_num**);**  **~**Tree**();**  // 根据数据内部分布类型初始化数据  void initData**(**init\_data\_type idt**);**  Node **\***nodes**;**  int num**;** // 节点数量  **};** |

# 3 Case类

一维模拟二维，辅助类

|  |
| --- |
| **typedef** struct data2D\_st  **{**  size\_t rows**;**  size\_t cols**;**  DATA\_TYPE **\***data**;**  size\_t pitchBytes**;**  **}**Data2D**;** |

核心组织类 Case.h

|  |
| --- |
| // 核心组织类  // 在该类中，定义了不同的数据结构形式，不同的访问方式，不同的内存以及不同的线程数目  // void global\_run();  // void shared\_run();  // void constant\_run();  // 为核心函数，以得出在不同内存下的访存性能  class Case  **{**  public**:**  Case**();**  **~**Case**();**  // 数据组织形式  // 一维数组，树等  data\_form df**;**  // 数据大小  int size**;**  int r**,**c**;** // 二维数组所需数据  // 数据内容形式  // 随机、各种分布  data\_content dc**;**  // 根据数据组织形式、数据大小和数据内容形式初始化数据  void initData**();**  // 线程数目  int thread\_num**;**  int block\_size**;** // 线程块大小  // 不同的访问方式  access\_mode am**;**  int am\_num**;** // 访问数据的数量  // 三个运行函数，即核心函数  // 分别得出在不同内存下的访问性能  // 每个函数内执行流程如下：  // (1)（申请空间，初始化数据需提前完成）  // (2)数据拷贝，根据各个参数执行第五节中定义的各个核函数  // (3)释放数据  void global\_run**();**  void shared\_run**();**  void constant\_run**();**  private**:**  // 数据组织形式  DATA\_TYPE **\***data1D**;**  Data2D data2D**;**  Tree tree**;**  **};** |

# 4 Distribution函数

根据分布类型和所给出的下标生成各种分布数据

|  |
| --- |
| \_\_host\_\_ \_\_device\_\_ DATA\_TYPE sequential**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE step**(**int index**,** int step**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE random**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE standard\_normal**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE poisson**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE uniform**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE geometric**(**int index**);**  \_\_host\_\_ \_\_device\_\_ DATA\_TYPE exponential**(**int index**);** |

# 5 核函数

每种数据类型共六个函数，命名方式为：\_数据类型+数据位置+访问类型+Ker（step访问方式与其他访问方式相比多一个参数，故访问类型，step为step访问；common为其他访问方式）。

**核函数内流程：**

（1）计算线程索引

shared需要根据计算将数据拷贝到shared memory中

（2）循环访问数据数量（am\_num）次

（3）根据访问方式，生成访问下标

（4）访问数据

|  |
| --- |
| //  static \_\_global\_\_ void \_d1DGloalCommonKer**(**DATA\_TYPE **\***data1D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d1DGloalStepKer**(**DATA\_TYPE **\***data1D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_d1DSharedCommonKer**(**DATA\_TYPE **\***data1D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d1DSharedStepKer**(**DATA\_TYPE **\***data1D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_d1DConstantCommonKer**(**DATA\_TYPE **\***data1D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d1DConstantStepKer**(**DATA\_TYPE **\***data1D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_d2DGloalCommonKer**(**Data2D **\***data2D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d2DGloalStepKer**(**Data2D **\***data2D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_d2DSharedCommonKer**(**Data2D **\***data2D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d2DSharedStepKer**(**Data2D **\***data2D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_d2DConstantCommonKer**(**Data2D **\***data2D**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_d2DConstantStepKer**(**Data2D **\***data2D**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_treeGloalCommonKer**(**Tree **\***tree**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_treeGloalStepKer**(**Tree **\***tree**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_treeSharedCommonKer**(**Tree **\***tree**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_treeSharedStepKer**(**Tree **\***tree**,** int step**,** int am\_num**);**  static \_\_global\_\_ void \_treeConstantCommonKer**(**Tree **\***tree**,** access\_mode am**,** int am\_num**);**  static \_\_global\_\_ void \_treeConstantStepKer**(**Tree **\***tree**,** int step**,** int am\_num**);** |