**程序使用说明**

陈 凯

复旦大学

本程序分为CPU和GPU两个版本的实现，并且实现了一个UI界面调用CPU和GPU内核。

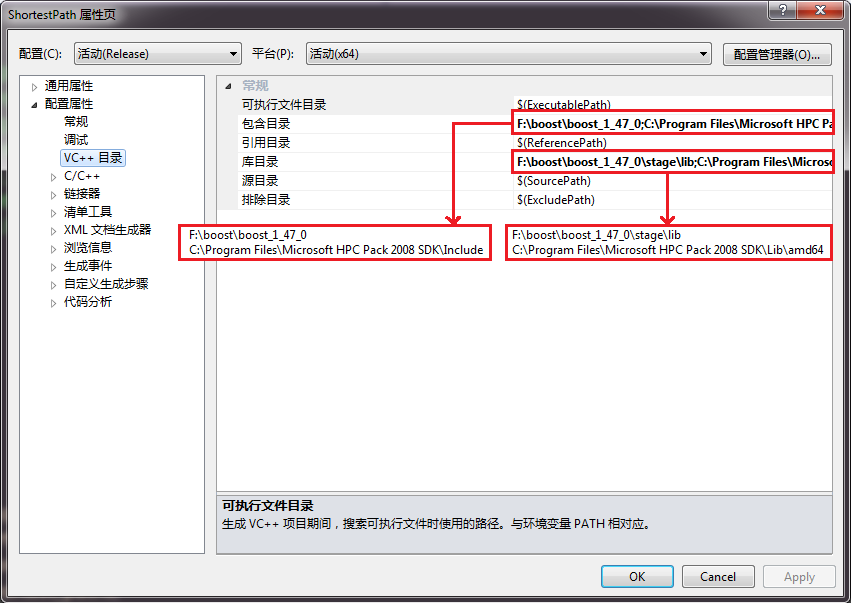
1. **CPU**

可执行文件存放在bin/CPU。

**软件环境**：Windows 7 + VS2010 + Boost 1.47 + Microsoft HPC Pack 2008 SDK（以上软件都为64位）

**HPC和Boost的编译**：请先安装Microsoft HPC Pack 2008 SDK，确保安装在默认路径。之后安装编译boost库。Boost库编译时，在boost根目录下的配置文件(project-config.jam)中添加using mpi ; 开启mpi的编译。在boost\_1\_47\_0\tools\build\v2\tools\mpi.jam中local cluster\_pack\_path\_native 参数确保是HPC的安装目录。

**VS2010配置**：VC++目录选项卡“包含目录”和“库目录”分别配置了Boost库和Microsoft HPC 2008 SDK 的对应文件夹。如测试机器的程序安装目录不同，请做相应修改。



为了便于算法的执行，我调用了Boost库中的METIS文件格式作为算法的输入格式。需先将大赛中的测试文件\*.gr转换成\*.metis文件（已在执行文件中集成）。转换文件以及读入需一定时间，请耐心等待。

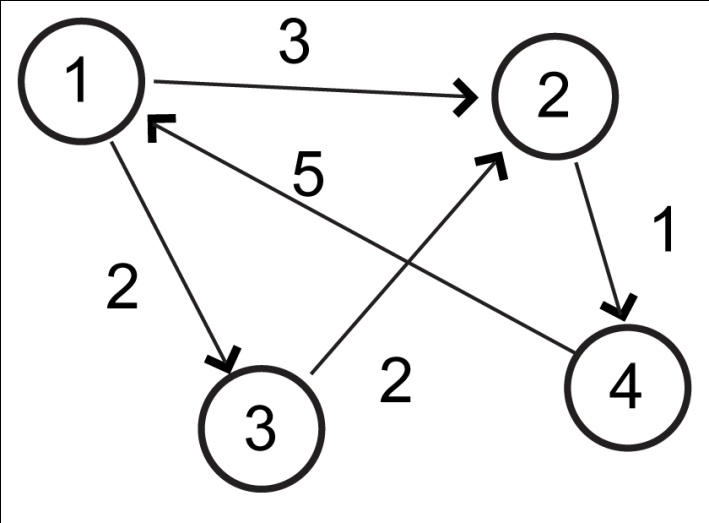
Metis格式简要说明：

第一行：n m type

其中 n为节点数，从1到n编号。m为总边数。type是图的类型，1表示链表。

以下n行，第i行代表节点i的链表。每行中有p对数，一对数中，第一个为节点号，第二个为距离。表示节点i到此节点的距离。

如图：



metis文件表示为：

4 5 1

3 2 2 3

4 1

2 2

1 5

图中有4个节点，5条边。且按照链表方式输出。

另外由于metis中的阅读器将每个节点索引分别减1，所以，节点从0开始编号。在算法中，我做了相应处理。

Metis文件的具体信息可参考[这里](http://www.boost.org/doc/libs/1_47_0/libs/graph_parallel/doc/html/metis.html)。

**程序使用方法：**

ShortestPath.exe [method] [filename] [source] [target]

ShortestPath.exe –h 得到帮助

命令事例：Shortest.exe dijkstra USA-road-d.NY.gr 或

Shortest.exe 1 USA-road.d.NY.gr

用dijkstra算法法计算USA-road-d.NY.gr中的节点1到最大节点n的距离。

其中:

method表示选用何种最短路径算法：

0) all : 所有的方法都将被测试

1) dijkstra : 串行Dijkstra算法。

2) crauser : 并行Crauser 算法。

3) delta-stepping : 并行Delta Stepping算法。

4) bellman-ford :串行Bellman Ford算法。

指定方法名称或索引号皆可。

filename: 输入的图文件。

source: 源节点（可选）。如果没有指定，默认节点为1。

target: 目标节点（可选）。如果没有指定，默认节点为n。（即图中索引号最大的节点）

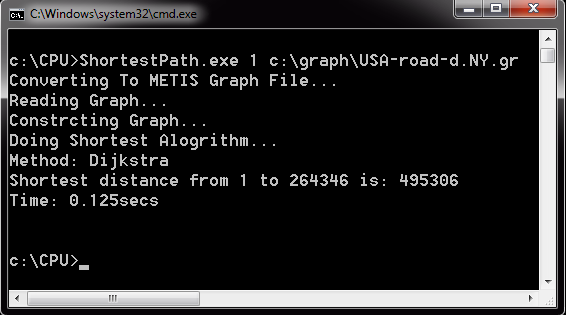
输出格式：

Method: 采用的方法

Shortest distance from [source] to [target] is: 从节点source到target的最短路径

Time: 算法耗时

以下是输出地图USA-road-d.NY.gr中从节点1到节点n的最短路径。



1. **GPU**

可执行文件存放在bin/GPU中。

**软件环境：**

Windows 7 + VS2010 + Nvidia GPU Computing SDK 4.0+ Parallel Nsight 2.0 （以上软件都为64位）

在GPU的算法中，不需要将文件转换为metis格式。

命令格式和CPU部分基本相同（多出printPath选项）：

ShortestPath.exe [method] [filename] [source] [target][printPath]

ShortestPath.exe –h 得到帮助。

命令事例：Shortest.exe cuda-delta-stepping USA-road-d.NY.gr或

Shortest.exe 3 USA-road-d.NY.gr

用CUDA Delta-Stepping算法计算USA-road-d.NY.gr中的节点1到最大节点n的距离。

其中:

method表示选用何种最短路径算法：

0) all : 所有的方法都将被测试

1) cuda-dijkstra : CUDA Dijkstra算法。

2) cuda-bellman: CUDA Bellman Ford算法。

3) cuda-delta-stepping: CUDA Delta-Stepping算法。

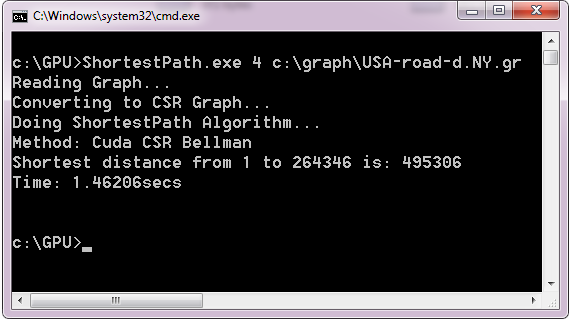
4)cuda-csr-bellman-scalar: CUDA CSR Bellman Ford (Scalar) 算法。

5)cuda-csr-bellman-vector: CUDA CSR Bellman Ford (Vector) 算法。

6)cuda-ell-bellman: CUDA ELL Bellman Ford算法。

printPath选项（可选）：0为不输出路径，1为输出路径。如果没有指定，默认为0。

其余命令参数和CPU版本相同。



1. **UI界面**

为了便于程序使用，我用C#实现了一个UI界面调用上述的cmd文件。由于结果的显示非实时，等cmd全部计算完成才能显示。所以如需了解程序执行进度，请使用命令行界面。

