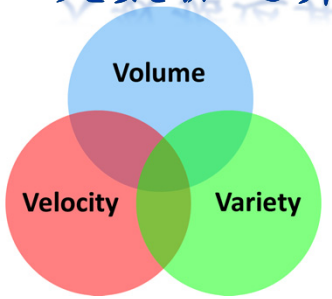
大数据系统与大规模数据分析

作业2：
大数据运算系统编程



陈世敏

中科院计算所
计算机体系结构
国家重点实验室

课程相关

- 成绩分配
  - 闭卷考试：50%
  - 作业1+作业2+作业3：30%
  - 大作业：20%
  - 课堂表现：+5%

作业2安排

- 成绩：占总成绩10%
- 时间
  - 发布：2016/4/13(Wed)
  - 上交：**2016/5/4 (Wed)，北京时间 6:59pm（共3周）**
  - 在课程系统中提交
    - 组号_学号_hw2.java 对应MapReduce程序
    - 组号_学号_hw2.cc 对应同步图运算程序
  - 晚交
    - 最晚：2016/5/11(Wed)，北京时间 6:59pm，将扣除20%成绩
    - 之后不再接收

分组（与作业1不同）

- 共分为**4个组**，每个组的作业题目有一定区别
- 分组方式如下
  - 组号 = (学号最右面6位数字) % 4
  - %是求余数
- 举例
  - 学号最右面6位数字=229032
  - 组号=229032 % 4 = 0
  - 所以是第0组

上机安排(1)

- 地点
  - 计算机学院，4层
  - 网络安全教学实验室（447室）：50台
  - 云计算教学实验室（432室）：20台
- 机器：联想PC机M6400t，Windows 7/32bit
  - 环境：每台机器安装了一个虚拟机，运行Ubuntu Linux 14.04.2, Hadoop 2.6.0, HBase 0.98等
  - 本作业只需要在单机上构成伪分布环境
- 注：可以在自己的计算机上完成作业

上机安排(2)

- 时间
  - 周五上午，8:30-11:50am
  - 周五下午，1:00-4:20pm
- 助教
  - 牛颂杰，王浩博，单鼎一
- 上机期间助教的职责
  - **管理上机秩序：**上机前找助教签到，分配机器；使用完毕，找助教签出；助教负责监督机房秩序(不得喧哗、打闹等)。
  - **解答机器使用的问题：**包括如何开机、如何登录、如何使用编辑器、如何编译和运行程序
  - **不包括：其它关于作业内容的问题**

作业内容

- 目的
  - 学习Hadoop编程
  - 学习同步图运算的编程
- 分为两个部分（共10%）
  - Hadoop编程（5%）
    - 所有组的作业内容相同
  - 同步图运算编程（5%）
    - 每个组实现不同的图运算

Hadoop编程

- 输入文件：文本文件
  - 每行格式
    - <source>␣<destination>␣<time>
    - 3个部分由空格隔开
    - 其中source和destination为两个字符串，内部没有空格
    - time为一个浮点数，代表时间（秒为单位）
    - 涵义：可以表示一次电话通话，或表示一次网站访问等
  - 输入可能有噪音
    - 如果一行不符合上述格式，应该被丢弃，程序需要正确执行
- MapReduce计算：统计每对source-destination的信息
- 输出
  - <source>␣<destination>␣<count>␣<average time>
  - 每一个source-destination组合输出一行（注意：顺序相反按不同处理）
  - 每行输出通话次数和通话平均时间(保留3位小数，例如2.300)

同步图运算

- Group 0: SSSP
- Group 1: KCore
- Group 2: Graph Coloring
- Group 3: Directed Triangle Counting

- https://github.com/schencoding/GraphLite

SSSP

- Single Source Shortest Path
  - 给定一个顶点V0
  - 求其它每个顶点到V0的最短路
- 计算方法
  - 每个顶点Vertex Value记录当前已知的最短路长度
  - 初始化：V0: 0; 其它顶点: 无穷大
  - 迭代
    - 发送的消息：当前顶点的最短路长度+出边长度
    - 收到消息后，更新当前最短路长度值
- 输入：图，V0 (命令行参数)
- 输出：顶点ID，最短路长度

  顶点ID: 最短路长度
  顶点ID: 最短路长度
  …

KCore

- KCore
  - 一个图G的 KCore 是G的子图
  - 这个子图的每个顶点的度≥K
- 计算方法
  - 每个顶点记录: is_deleted, current_degree
  - 如果顶点的度小于k，从图中删除该顶点，然后给邻居发送消息
  - 顶点收到消息后，得知被删掉的邻居顶点，更新自己的度
- 输入：无向图（有成对的有向边）
- 输出: KCore 子图中的所有顶点

  顶点
  顶点
  …

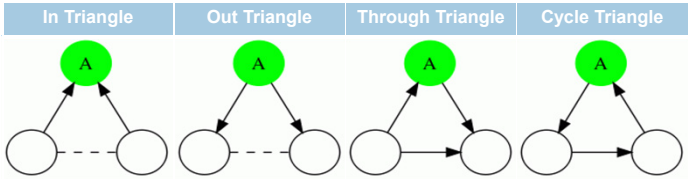
Graph Coloring

- Graph Coloring
  - 对图的顶点着色，相邻顶点不同颜色，给出一种着色方案。
  - 假设可用color数比实际需要的最小数大很多
- 计算方法
  - 每个顶点记录自己的color，初始为-1
  - Superstep = 0，顶点V0着色color=0，向邻居发送颜色编号
  - 接下来的 superstep 中，顶点收到消息后，统计邻居顶点的颜色，随机选择一个与之不冲突的颜色号着色
- 输入：无向图（有成对的有向边），命令行：V0，总color数
- 输出：

  顶点id： 颜色号
  顶点id： 颜色号
  …

Directed Triangle Counting (1)

- 有向图三角形计数：
  - 三角形类型（虚线表示箭头方向任意）



  - 从单个顶点角度，计算各类型三角形数
  - 累计上述计数，求一个无向图中各类型三角形的总数（会有多次计数）

Directed Triangle Counting (2)

- 思路
  - A知道每个邻居的所有邻居，就可以计算上述的in/out/through/cycle triangle个数
  - 如何获得邻居的邻居？
    - 每个顶点可以知道自己的out-neighbor
    - 经过一次超步通信，每个顶点可以知道自己的in-neighbor
    - 那么每个顶点都可以把in-neighbor和out-neighbor，发给邻居
  - 发送消息
    - 消息是定长的，可以发多条消息
  - 使用aggregate统计最终的triangle个数

Directed Triangle Counting (2)

- 输入：有向图

- 输出：in/out/through/cycle triangle个数

  in: 个数
  out: 个数
  through: 个数
  cycle: 个数