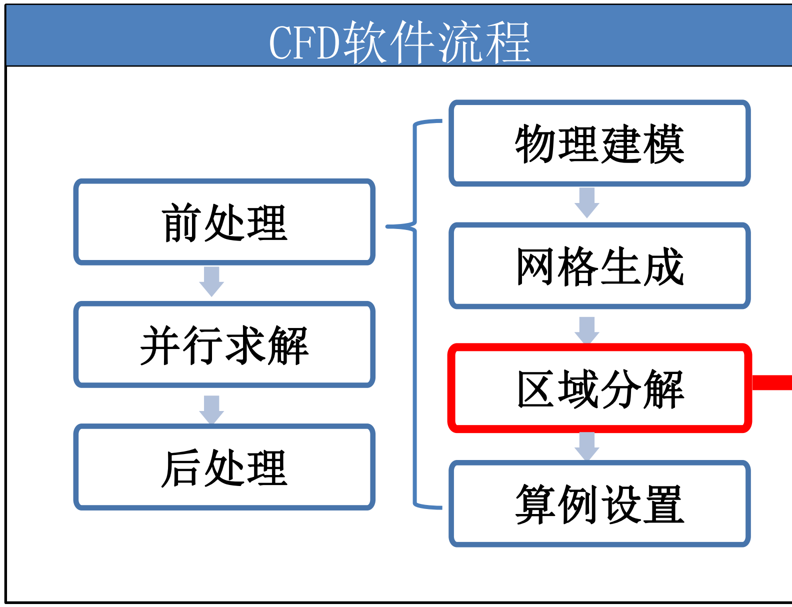
相等子集最小图分割问题

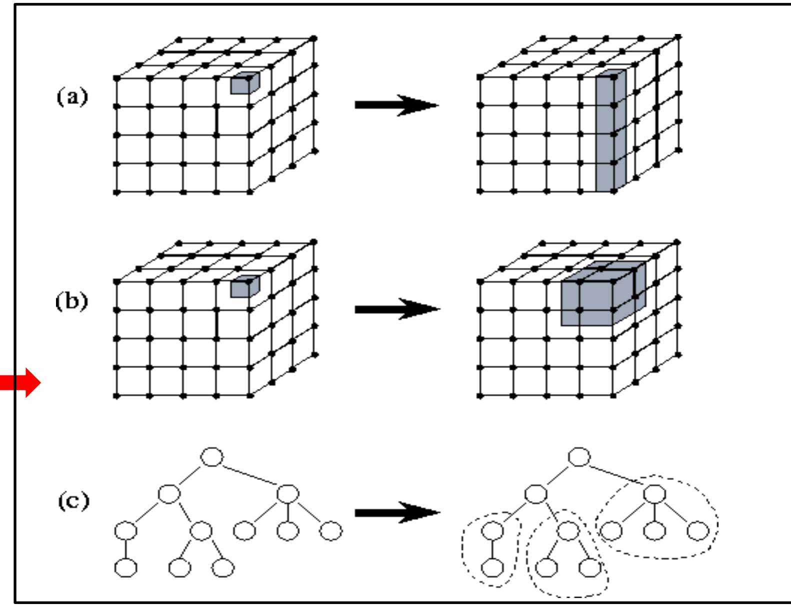
报告人：董玲玉
任课教师：段世红

目录

问题背景

区域分解



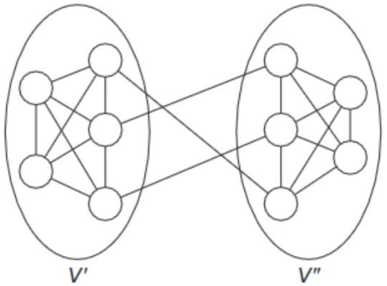


区域分解将细粒度网格聚集（划分）为粗粒度任务，**提供更小数量任务，每个任务大小更大。**

目标：通信量少 + 负载均衡

问题描述

相等子集最小割



输入： 图G=(N, A)，两个不同的节点s和t，正整数W

性质： 节点集$N = S_1 \cup S_2$且$S_1 \cap S_2 = \emptyset$，$|S_1| = |S_2|$，$s \in S_1$，$t \in S_2$，并且$|\{\{u,v\} \in A: u \in S_1, v \in S_2\}| \leqslant W$

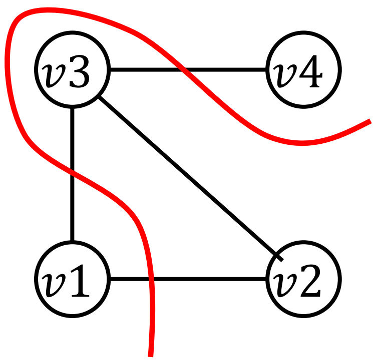
——NPC问题。 证明思路？

判断一个NP问题是不是NP-Complete的两个方法

1. 找到一个NP-Complete问题，经过证明可以reduce to 你的问题，这意味着你的方法可以解决这个NP-Complete问题，那很显然，这个解决方法也是NP-Complete的。
2. 所有的NP问题都可以reduced到你的问题

图的最大割

——NPC问题，karp[1]的21个NPC问题中证明



输入： 图$\mathrm{G} = (\mathrm{N}, \mathrm{A})$，权值函数$\mathrm{w}: \mathrm{A} \rightarrow \mathrm{Z}$（非负整数），正整数W

性质： 有子集$\mathrm{S} \subseteq \mathrm{N}$，满足$\sum_{\{u,v\} \in A, u \in S, v \in N-S} w(\{u,v\}) \geq W$

其中N是点集，A是边集。

图的最大割问题为NPC问题[1]
图的最大割$\leq_P$相等子集最小割[2]

→ 相等子集最小割为NPC问题

[1]Karp R M. Reducibility among combinatorial problems[M]//Complexity of computer computations. Springer, Boston, MA, 1972: 85-103.
[2]Johnson D S, Stockmeyer L. Some simplified NP-complete graph problems[J]. Theoretical Computer Science, 1976, 1: 237-267.

解决方法

解决思路

相等子集最小割为NPC问题 → 无法在多项式时间内找到最优解

↓

图划分算法/启发式算法

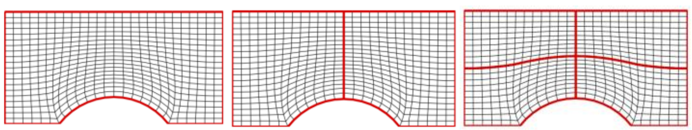
尝试使用的算法可能不是每次都能给出最优解，但至少在大多数情况下都能给出一个好的解。好的区域分解算法需要在执行时间与解决方案的质量之间权衡

贪心算法：

>过程简单，易于实现，执行速度快。

>一般用于多块结构网格的区域分解，并且无法有效控制分区间的进程通信开销。

例：



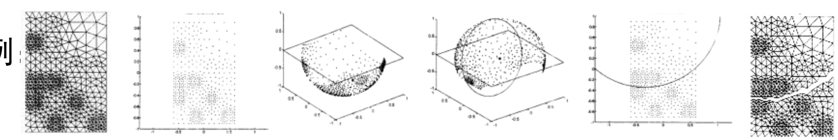
初始网格　一分为二　二分为四

几何划分算法：

>执行速度快，综合考虑负载平衡和最小通信。

>对网格单元的形状有一定的要求，单元纵横比以及顶点角度都有限制。比较适合二维问题的处理。

例：



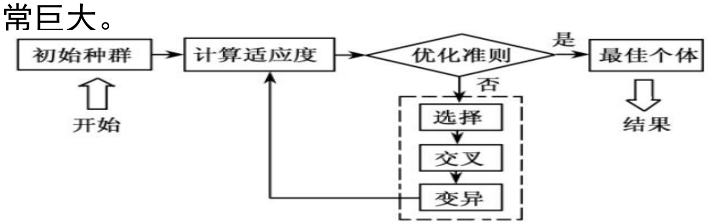
输入网格　网格坐标　坐标映射　球上圆弧　圆弧反映射　划分结果

遗传算法：

>综合考虑负载平衡和最小通信，逼近最优解。

>随着网格规模和种群数量的增加，内存要求和时间成本将非常巨大。

例：

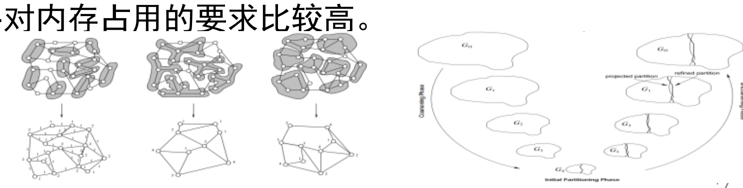


多级K-Way算法：

>执行速度快，综合考虑负载平衡和最小通信，而且有并行版本。

>对内存占用的要求比较高。

例：



图的粗化　多级K-Way划分图

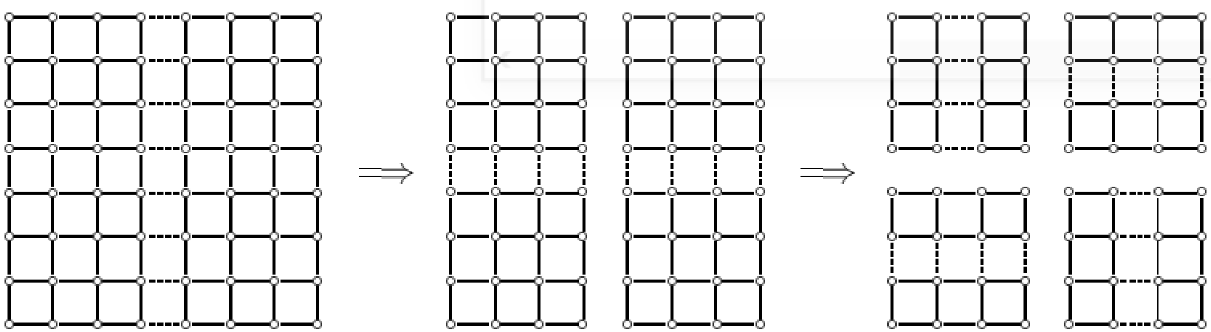
递归二分法概述

区域分解常用技术

递归二分法：

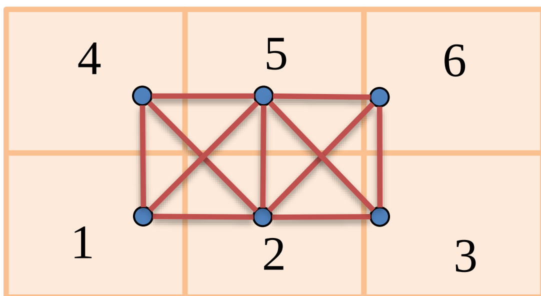
递归二分法是一类算法的总称，指通过一定的划分依据将网格转化得到的对偶图划分为两个子图，并在子图上递归的进行划分直到得到目标分区数的一系列算法。

算法	划分依据	优势	劣势
递归坐标二分法	坐标值	运行速度快，实现简单，保证负载平衡	无法控制通信成本
递归图二分法	顶点间最短路径	保证负载平衡，一定程度上考虑通信	运行时间较长，可能出现很差的分区结果
递归谱二分法	费德勒向量	保证负载平衡和减小通信	运行时间长
……	……	……	……

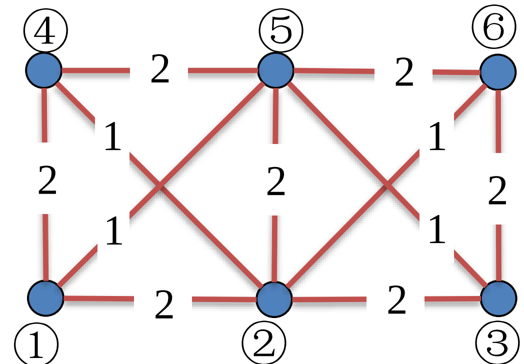


对偶图&Laplacian矩阵

生成网格对偶图的Laplacian矩阵



网格



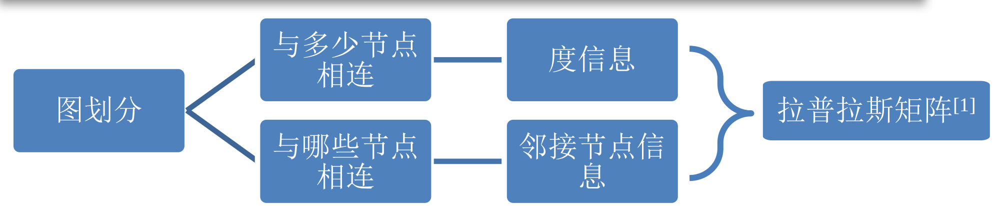
带权对偶图

- 非对角线：
  若两网格共用点：G_ij = -1
  若两网格公用边：G_ij = -2（两个点）
  若两网格共用面：G_ij = -4（四个点）
  若两网格无公共点：G_ij = 0
- 对角线：
  G_ii = -(所在列所有非对角元素和)

①节点度=2+2+1=5 ②节点邻接，权值为2

$$\begin{bmatrix} 5 & -2 & & -2 & -1 & \\ -2 & 8 & -2 & -1 & -2 & -1 \\ & -2 & 5 & & -1 & -2 \\ -2 & -1 & & 5 & -2 & \\ -1 & -2 & -1 & -2 & 8 & -2 \\ & -1 & -2 & & -2 & 5 \end{bmatrix}$$

拉普拉斯矩阵G
（节点度矩阵-节点邻接矩阵）



使用RSB划分图为P个子图

递归谱二分法

计算热点

求拉普拉斯矩阵的费德勒向量

使用Lanczos迭代方法计算与拉普拉斯矩阵相似的三对角矩阵

求该三对角矩阵的特征值&特征向量

对计算出的费德勒向量顶点排序

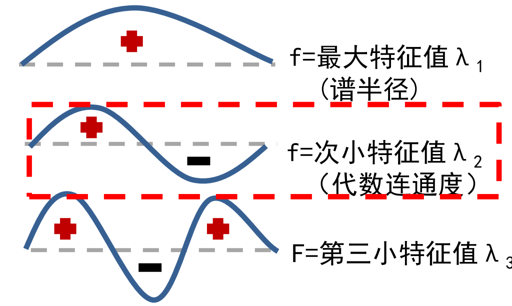
根据向量分量的符号给每个子图分配一半顶点

算法 递归谱二分法
1. 计算费德勒向量（使用lanczos算法）
2. 对计算出的费德勒向量的顶点进行排序
3. 给每个子域分配一半的顶点
4. 递归1-3步直到分配完所有顶点

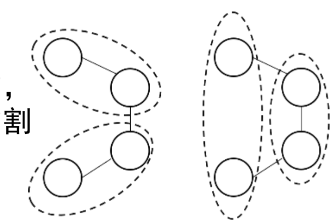
优缺点

缺点：计算量大，耗时长
（并行优化动机）

优点1：
基本负载均衡



优点2：
产生的子域是连通的，
从而可以减少划分的割

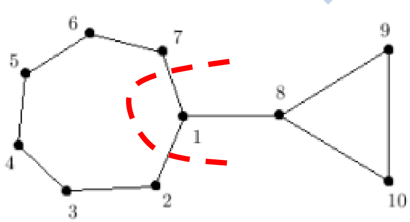


(a)子图连通
割=1

(b)子图不连通
割=2

不是最优解（问题本身为np难），但在图划分算法中已属优秀，平衡图的最小割（减少通信时间）与平均两个子图顶点个数(负载均衡)

实例（不带权对偶图）



(a) The graph A

$$
\begin{array}{c|cccccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\hline
1 & 3 & -1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\
2 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
4 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\
7 & -1 & 0 & 0 & 0 & 0 & -1 & 2 & 0 & 0 & 0 \\
8 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & -1 & -1 \\
9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\
10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 2
\end{array}
$$

(b) The Laplacian matrix $L(A)$

$$
\begin{array}{c|c}
1 & 0.0521 \\
2 & -0.1147 \\
3 & -0.2543 \\
4 & -0.3335 \\
5 & -0.3335 \\
6 & -0.2543 \\
7 & -0.1147 \\
8 & 0.3734 \\
9 & 0.4897 \\
10 & 0.4897
\end{array}
$$

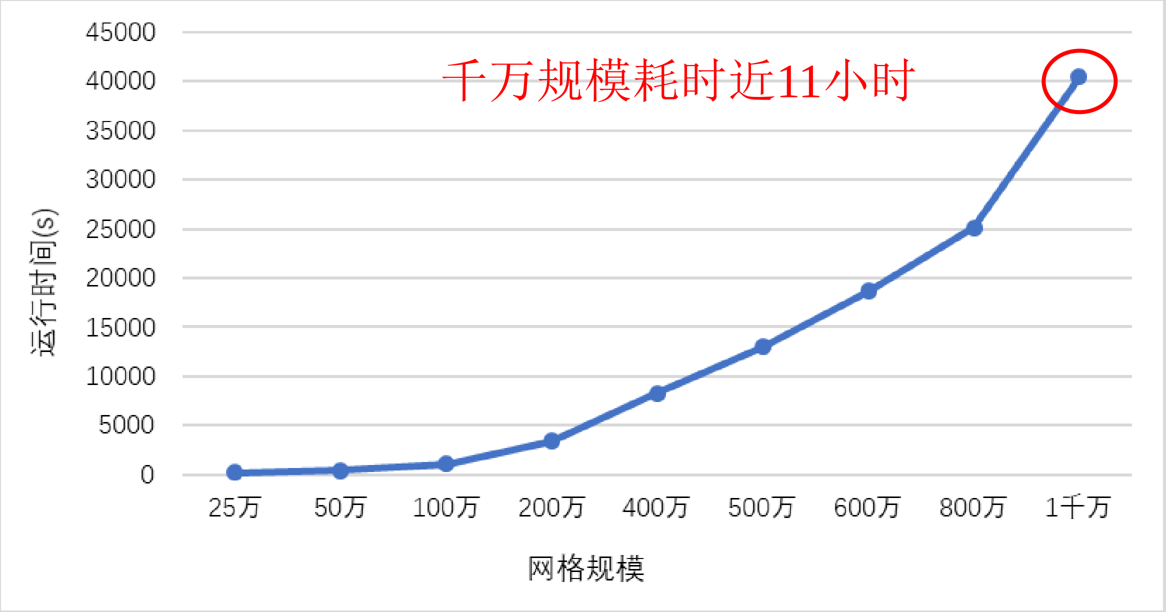
(c) The Fiedler vector $\mathbf{x}_A$

区域分解并行化需求

递归谱二分法——串行

在数值堆热工水力模拟问题中，核燃料组件的高精细建模将产生非常大的网格规模需求。随着问题规模的增大，现有串行程序在区域分解过程中的时间成本变得非常巨大，千万网格规模下的耗时甚至接近11小时，严重阻碍实验进行，引入并行化技术，解决区域分解的时间成本问题具有重要意义。

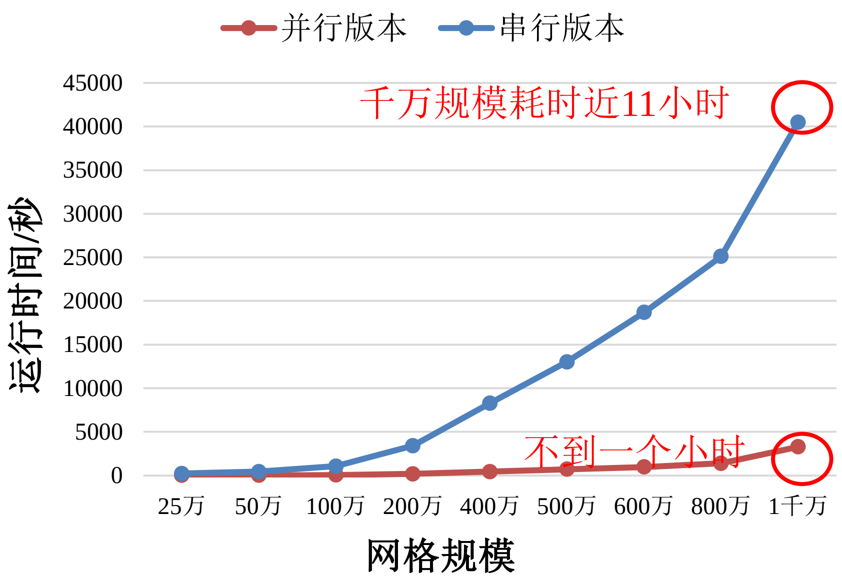
测试环境：
天河二号
测试算例：
单棒流道网格文件
测试程序：
串行区域分解程序

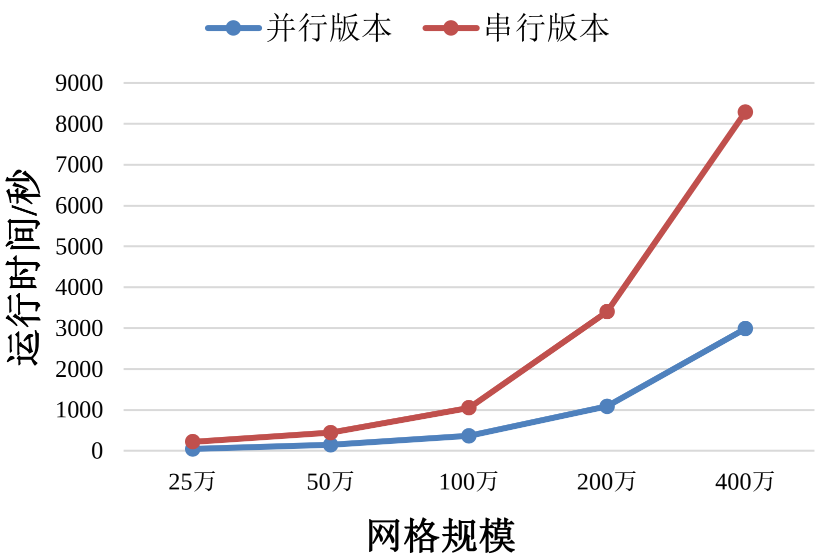


运行时间测试结果

区域分解并行化结果

递归谱二分法——并行





MPI+OpenMP方案测试结果

MPI+CUDA方案测试结果

测试环境： 天河二号
测试算例： 单棒流道网格
进程数： 8 **线程数：** 12
加速比： 15至19

测试环境： CNGrid12
测试算例： 单棒流道网格
进程数： 2 **GPU：** Tesla V100
加速比： 2.7至3.1

谢谢观看