arXiv:2301.12390v1 [cs.SI] 29 Jan 2023

提升并行 Louvain 算法在社区检测中的效率

Subhajit Sahu 安全、理论与算法研究中心（CSTAR）
IIITHyderabad，印度 - 500 032
subhajit.sahu@research.iiit.ac.in

摘要

社区检测是网络分析的重要方面，它能够识别网络中的群组和模式。随着网络规模不断扩大，采用快速算法高效分析网络至关重要。Louvain 算法是一种基于模块度的贪心算法，能够通过多轮迭代将网络划分为互不连通的社区。在大型、稠密网络中，它也以能够构建高质量社区而闻名。然而，它的速度至少比依赖标签传播的社区发现技术慢十倍；后者通常极快，但得到的社区质量较低。研究人员提出了多种并行化和改进 Louvain 算法的方法。为了确定哪种策略通常最合适，以及哪些参数值能够在不损害社区质量的前提下实现最高性能，评估这些现有方法的性能和准确性至关重要。本报告实现了静态 Louvain 算法的单线程和多线程版本，并仔细研究该方法的具体细节，进行必要的调整和优化，同时确定合适的参数值。每轮之间的容差可以调整，从而改变该方法的性能。初始容差设为 0.01、容差递减因子设为 10 时，算法的异步版本效果最佳。总体而言，我们的研究结果表明，该方法并不适用于共享内存并行；不过，一种可能的变通方法是将图拆分为可独立处理的适当大小的块，再将它们合并。

1 引言

来自现实世界数据源的互联数据日益增多，例如社交网络和生物网络，这使得图作为一种表示方式的应用越来越广泛 [1]。然而，这些图通常规模庞大，因此需要借助并行处理来应对数据规模。此外，许多现实世界的图都是动态的，边不断地增加和删除 [2]。因此，近年来，针对动态图分析以及更新图分析结果的并行算法研究受到了广泛关注。此类研究包括动态计算中心性分数 [3, 4, 5, 6, 7], 、维护双连通分量 [8, 9, 10], 以及计算最短路径 [11, 12, 13]。

社区检测是图分析中广受研究的课题，在电子商务、通信网络和医疗保健等领域有着实际应用。它旨在识别图中的顶点群组，即社区；这些群组内部连接密集，而与网络其余部分的连接稀疏。如果仅根据网络的拓扑结构识别这些结构，则称为内在社区。另一方面，外在社区则根据外部信息或节点属性定义，例如是否属于某个特定组织或地理位置。图中可以识别出多种类型的社区。不相交社区是指每个顶点恰好属于一个社区的情况（本文研究的类型）。而重叠社区[14] 允许每个顶点属于多个社区；层次社区则具有多层级的成员关系结构。

社区检测是一项具有挑战性的问题，因为它是 NP难，而且社区的数量及其规模分布事先并不明确。解决这一问题的方法多种多样，例如标签传播 [15, 14, 16], 随机游走 [17], 扩散 [18],自旋动力学 [19], 适应度指标优化 [20, 21], 统计推断 [22, 23], 模拟退火 [24, 19], 团渗流 [25, 26, 27], 等等。这些方法可分为两大类：分裂式方法和凝聚式方法。分裂式方法也称为自顶向下方法，它假设图中的所有顶点属于同一个社区，然后通过迭代识别并移除桥，将其拆分为连接更紧密的社区 [28, 29]。凝聚式方法或自底向上方法则将两个或更多社区合并，以最大化某个分数 [30, 2]。另一种方法是种子集扩展：从一组相关的目标种子顶点出发，逐步扩展并形成围绕它们的社区 [31]。

使用不同的社区检测方法时，可以通过质量函数评估得到的社区。一种常用的指标是模块度，由 Newman 和 Girvan 提出，用于比较社区内部边的数量与随机零模型中的预期数量。模块度的取值范围为 -0.5（非模块化聚类）到 1.0（完全模块化聚类），理论上，优化这一函数可以得到最佳分组。另一种适应度得分是导通率，用于衡量社区割或社区间边。另一种常用的质量函数是常数波茨模型（CPM），旨在克服模块度的一些局限性 [32]。

2 文献综述

目前已有多种用于检测网络中社区的技术。其中一些基于模块度优化、层次聚类、标签传播、区域密度、核心聚类 [33], 、博弈论、信息论（Infomap） [34, 35, 36, 37, 38],以及生物进化（遗传学） [39, 40, 41]。模块度得分 [20, 42, 40],、标准化互信息指数（NMI） [43, 44], 和杰卡德指数 [43] 等指标常用于比较不同方法所得社区的质量。

Louvain方法是一种基于模块度的贪心优化算法，通过对图中的顶点进行分层聚合来获得社区。该方法由鲁汶大学的Blondel等人创建 [42]，是这一领域最受欢迎的启发式算法之一。它的

平均时间复杂度为 $\Theta(n\log\ n)$，其中 n 为网络中的节点总数 [45]。执行Louvain 算法的方法可以是粗粒度的，即并行处理一组顶点；也可以是细粒度的，即并行处理所有顶点。Grappolo软件库中实现了多种针对Louvain 算法的并行化启发式算法 [46]。不过需要注意的是，依赖模块度最大化的社区检测方法（如Louvain方法）已知会受到分辨率限制问题的影响，从而无法识别某些规模的社区 [47]。对Louvain 算法的一些改进包括采用合适的启发式分区方法 [48], 处理图分区之间的幽灵顶点 [48], 限制内部搜索规则 [49], , 以及尽早剪除没有希望的候选项 [49]。其他值得关注的方法包括在BigData批处理框架中使用 MapReduce [50]。

Louvain 算法的主要优势之一是能够发现模块度较高的社区。模块度用于衡量社区内部连接的密集程度，以及不同社区之间连接的稀疏程度。不过，Louvain 算法也存在一些局限性。它可能对节点的初始顺序敏感，从而导致结果无法复现。此外，它也难以检测连接非常密集或紧密相连的社区，因为它倾向于拆分密集社区，以便找到规模更大、凝聚力更强的社区。

3 评估

本节首先介绍我们的实验设置，例如所用系统和数据集。随后，我们将考察静态 Louvain 算法，留意其中的细节并探索各种优化方法，同时实现这些方法的单线程和基于 OpenMP 的多线程版本。

3.1 实验设置

3.1.1 所用系统

在实验中，我们使用了一套系统，其中包括两颗运行频率为 2.10 GHz 的 Intel Xeon Silver 4116 64 位处理器，以及 128GB、运行频率为 2666 MHz 的 DDR4同步寄存式动态随机存取存储器。每颗处理器均配备 12 个 x86核心，每个核心具有 2 个超线程，并拥有 16.5M L3缓存。服务器运行 CentOS 7.9，使用 GCC 9.3 和 OpenMP 5.0 进行编译，优化级别设为 3（-O3）；所有实验均启用了同时多线程（SMT）。

3.1.2 数据集

我们实验中使用的图详见表1。这些图来自 SuiteSparse矩阵集合 [51]。图中的顶点总数从 7.49 万到 1200 万不等，边总数从 81.1 万到 3.04 亿不等。所有边均视为无向边，默认权重为 1；此外，还为所有图中的每个顶点添加了自环。

表1：在实验中，我们使用了 17 个图。每个图中的边都添加了方向相反的副本，使其成为无向图，并且每条边的权重均设为 1。该表列出了每个图的顶点总数（$|V|$）、将图转为无向图后的边总数（$|E|$），以及顶点的平均度（D_{avg}）。顶点数和边数均按具体情况四舍五入至最接近的千位或百万位。

图	$\|V\|$	$\|E\|$	D_{avg}
网页图			
web-Stanford	282K	3.99M	14.1
web-BerkStan	685K	13.3M	19.4
web-Google	916K	8.64M	9.43
web-NotreDame	326K	2.21M	6.78
印度支那-2004	7.41M	304M	41.0
社交网络			
社交网络-Slashdot0811	77.4K	1.02M	13.2
社交网络-Slashdot0902	82.2K	1.09M	13.3
社交网络-Epinions1	75.9K	811K	10.7
soc-LiveJournal1	4.85M	86.2M	17.8
合作网络			
coAuthorsDBLP	299K	1.96M	6.56
coAuthorsCiteseer	227K	1.63M	7.18
coPapersCiteseer	434K	32.1M	74.0
coPapersDBLP	540K	30.5M	56.5
道路网络			
意大利 OSM	6.69M	14.0M	2.09
大不列颠 OSM	7.73M	16.3M	2.11
德国 OSM	11.5M	24.7M	2.15
亚洲 OSM	12.0M	25.4M	2.12

3.2 Louvain 算法

如前所述，Louvain 算法是一种凝聚式层次社区检测方法，通过贪心方式迭代优化模块度。给定一个无向加权图，首先将所有顶点各自视为一个社区。在第一阶段（也称为局部移动阶段），每个顶点都会贪心地选择移动到某个邻居所属的社区，使模块度增幅最大。如果移动到任何邻居的社区都无法提高模块度，该顶点便留在自己的社区中。所有顶点依次执行这一过程。如果模块度的总变化超过某个阈值（容差参数），就会重复这一阶段。该阶段完成后，所有顶点便形成了第一层社区结构。接下来的阶段称为聚合阶段：将属于同一社区的所有顶点合并为一个超顶点，使社区之间的边表示为相应超顶点之间的边（合并边权重），而各社区内部的边则表示为相应超顶点上的自环（同样合并边权重）。局部移动阶段和聚合阶段合起来构成一个轮次。随后将该超顶点图作为下一轮的输入。此过程持续进行，直到模块度增幅低于某个阈值（轮次_容差参数，通常为 0，因为我们的目标是最大化模块度增益）。每轮都会得到每个顶点的社区归属层次结构，

并可将其表示为树状图 [52]。通常，我们将最高层级的结构视为社区检测过程的最终结果。

据观察，在每轮之间调整容差（称为阈值缩放）会影响算法的运行时间，但不会显著影响所得社区的模块度。我们通过实验确定每轮之间适宜的容差下降速率（tolerance_decline_factor 参数），同时也确定一个适用于大多数图的初始容差参数值。在第一项实验中，我们实现了单线程 CPU 版本的 Louvain 算法。我们将容差初始值从 1 调整至 10^{-12}，步长为 10，并将容差下降因子从 10 调整至 10^{4}。从结果中我们发现，初始容差为 0.01 时，所得社区的模块度最高，且所需计算时间最短。此外，增大容差_decline_factor 会增加计算时间（正如预期），但似乎不会影响最终的模块度。因此，容差_decline_factor 设为 10 较为合适。

原论文的作者采用了该算法的异步版本，其中每个顶点的社区成员关系可以取决于当前迭代中其邻居的社区成员关系（类似于高斯-赛德尔方法） [42]。不过，这种方法会对同一内存区域进行读写，在多线程实现中可能因缓存一致性开销而损害性能。因此，我们考虑对该算法的同步版本进行实验，其中每个顶点的社区成员关系只能取决于上一迭代中其邻居的社区成员关系（类似于雅可比方法）。我们在基于中央处理器的单线程算法实现上进行了这一比较。结果表明，同步版本和异步版本都能提供模块度质量相当的社区；对于某些图，异步版本提供的社区质量略高。然而，就总耗时以及局部移动阶段的迭代总次数（这是算法中开销最大的部分）而言，同步版本都明显慢于异步版本。因此，我们得出结论：在 Louvain 算法的并行实现中，采用异步或部分异步的顶点处理方法，相较于完全同步版本，可能会取得更好的性能。Halappanavar 等人探讨了通过图着色实现部分异步的顶点排序 [46]。

在第二个实验中，我们实现了一个基于 OpenMP 的多线程 Louvain 算法版本。与之前的算法类似，每个线程都有一个独立的哈希表，可用于选择移入模块度增量最高的社区。如果有多个社区的模块度增量并列最高，我们只选择哈希表中的第一个。与之前一样，为了提升性能，我们分别分配哈希表，而不是将它们连续存储在一个向量中。目前，我们使用 OpenMP 的“auto”调度方式和总计 12 个线程（这并非最优选择）。我们在一个双路插槽系统上，将线程数从 2 调整到 48；该系统的每个中央处理器有 12 个核心，每个核心有 2 个超线程。

从结果中可以看出，增加线程数只会稍微缩短 Louvain 算法的运行时间。这表明，当多个读线程和一个写线程访问公共内存位置时（这里指每个

顶点的社区成员关系），性能会下降（可能是由于缓存一致性压力增大），而且如果重叠访问过多，性能将趋近于顺序算法。使用全部 48 个线程进行社区检测会显著延长获取结果所需的时间，这可能是由于线程与操作系统之间的切换所致。收敛所需的迭代次数也会随着线程数增加而增加，这表明异步多线程实现的行为开始趋近于算法同步版本的行为，而同步版本的收敛速度远慢于异步版本。解决这些问题的一种方法是以某种方式将图划分为多个分区，使每个分区都可以独立运行，然后再将它们合并起来。

4 结论

Louvain 算法因能够在网络中找到高质量的社区而深受研究人员青睐。我们的研究聚焦于静态 Louvain 算法。在实现基于 OpenMP 的单线程和多线程变体时，我们仔细关注算法的细节，并进行必要的调整或优化，同时确定合适的参数值。通过调整每轮迭代之间的容差（也称为阈值缩放），可以改变该方法的性能。根据我们的研究结果，使用算法的异步版本，并将初始容差设为 0.01、容差_衰减_因子设为 10 时，能够以最少的计算时间生成模块度较高的社区。基于 OpenMP 的算法并行实现表明，该算法通常并不适合共享内存并行处理（除非输入图包含大量顶点）。一种可行的解决方案是将图划分为若干易于处理的分区，分别独立运行，然后再重新组合。

参考文献

[1] S. Sakr、A. Bonifati、H. Voigt、A. Iosup、K. Ammar、R. Angles、W. Aref、M. Arenas、M. Besta、P. A. Boncz等，“未来属于大图”，《ACM通讯》，第64卷，第9期，第62–71页，2021年9月。[2] N. Zarayeneh和A. Kalyanaraman，“Delta-Screening：一种用于更新动态图中社区的快速高效技术”，《IEEE网络科学与工程汇刊》，第8卷，第2期，第1614–1629页，2021年4月。[3] K. Lerman、R. Ghosh和J. H. Kang，“动态网络的中心性指标”，载于《第八届图挖掘与学习研讨会论文集》，2010年，第70–77页。[4] C. Li、J. Han、G. He、X. Jin、Y. Sun、Y. Yu和T. Wu，“静态和动态信息网络中SimRank的快速计算”，载于《第13届扩展数据库技术国际会议论文集》，2010年，第465–476页。[5] E. Bergamini和H. Meyerhenke，“全动态网络中介中心性的近似计算”，《互联网数学》，第12卷，第5期，第281–314页，2016年。[6] E. Nathan和D. A. Bader，“图中Katz中心性更新的动态算法”，载于《2017年IEEE/ACM社交网络分析与挖掘进展国际会议论文集》，2017年，第149–154页。

[7] S. C. Regunta、S. H. Tondomker、K. Shukla和K. Kothapalli，“动态接近中心性和中介中心性的高效并行算法”，《并发与计算：实践与经验》，第e6650页，2021年。[8] Z. Galil和G. F. Italiano，“维护动态平面图的双连通分量”，载于《自动机、语言与编程国际讨论会》。Springer，1991年，第339–350页。[9] W. Liang、R. Brent和H. Shen，“并行维护k-连通性的全动态方法”，《IEEE并行与分布式系统汇刊》，第12卷，第8期，第846–864页，2001年。[10] C. A. Haryan、G. Ramakrishna、K. Kothapalli和D. S. Banerjee，“全动态维护2-连通分量的共享内存并行算法”，载于《2022年IEEE国际并行与分布式处理研讨会（IPDPS）》。IEEE，2022年，第1195–1205页。[11] P. Narvaez、K.-Y. Siu和H.-Y. Tzeng，“最短路径树计算的新型动态算法”，《IEEE/ACM网络汇刊》，第8卷，第6期，第734–746页，2000年。[12] E. P. Chan和Y. Yang，“动态图中的最短路径树计算”，《IEEE计算机汇刊》，第58卷，第4期，第541–557页，2008年。[13] A. Khanda、S. Srinivasan、S. Bhowmick、B. Norris和S. K. Das，“用于更新大规模动态网络中单源最短路径的并行算法模板”，《IEEE并行与分布式系统汇刊》，第33卷，第4期，第929–940页，2021年。[14]S. Gregory，“通过标签传播发现网络中的重叠社区”，《New Journal of Physics》，第12卷，第103018页，2010年10月。[在线]。可访问：https://dx.doi.org/10.1088/1367-2630/12/10/103018[15] U. N. Raghavan、R. Albert和S. Kumara，“在大规模网络中检测社区结构的近线性时间算法”，《Physical Review E》，第76卷，第3期，2007年9月。[16] J. Xie、B. K. Szymanski和X. Liu，“SLPA：通过说话者—倾听者交互动态过程揭示社交网络中的重叠社区”，载于《2011年IEEE第11届国际数据挖掘研讨会》。IEEE，2011年，第344–349页。[17] M. Rosvall和C. T. Bergstrom，“复杂网络中的随机游走图揭示了社区结构”，《Proceedings of the National Academy of Sciences》，第105卷，第4期，第1118–1123页，2008年。[18] K. Kloster和D. F. Gleich，“基于热核的社区检测”，载于《第20届ACM SIGKDD国际知识发现与数据挖掘会议论文集》，2014年，第1386–1395页。[19] J. Reichardt和S. Bornholdt，“社区检测的统计力学”，《Physical Review E》，第74卷，第1期，第016110页，2006年。[20] M. E. Newman，“利用矩阵的特征向量发现网络中的社区结构”，《Physical Review E》，第74卷，第3期，第036104页，2006年。[21] S. Fortunato，“图中的社区检测”，《Physics Reports》，第486卷，第3–5期，第75–174页，2010年。[22] E. Cˆome和P. Latouche，“基于精确积分完整数据似然的随机块模型中的模型选择与聚类”，《Statistical Modelling》，第15卷，第564–589页，2015年3月，doi：10.1177/1471082X15577017。 [在线]。可访问：https://doi.org/10.1177/1471082X15577017[23] M. E. Newman和G. Reinert，“估计网络中的社区数量”，《Physical Review Letters》，第117卷，第7期，第078301页，2016年。

[24] R. Guimera 和 L. A. Nunes Amaral，“复杂代谢网络的功能制图”，《自然》，第 433 卷，第 7028 期，第 895–900 页，2005 年。[25] I. Der´enyi、G. Palla 和 T. Vicsek，“随机网络中的团渗流”，Physical Review Letters，第 94 卷，第 16 期，第 160202 页，2005 年。[26] S. Maity 和 S. K. Rath，“用于检测重叠社区结构的扩展团渗流方法”，载于 2014年计算、通信与信息学进展国际会议（ICACCI）。IEEE，2014 年，第 31–37 页。[27] S. K. Gupta、D. P. Singh 和 J. Choudhary，“基于团的重叠社区检测算法综述”，《知识与信息系统》，第 1–36 页，2022 年。[28] M. Girvan 和 M. E. Newman，“社会与生物网络中的社区结构”，Proceedings of the National Academy of Sciences，第 99 卷，第 12 期，第 7821–7826 页，2002 年。[29] S. Souravlas、A. Sifaleras、M. Tsintogianni 和 S. Katsavounis，“社交网络社区检测方法分类：综述”，《一般系统国际期刊》，第 50 卷，第 1 期，第 63–91 页，2021 年 1 月。[30] A. Zakrzewska 和 D. A. Bader，“一种用于图中局部社区检测的动态算法”，载于 2015年IEEE/ACM社交网络分析与挖掘进展国际会议（ASONAM）。IEEE，2015 年，第 559–564 页。[31] J. J. Whang、D. F. Gleich 和 I. S. Dhillon，“使用种子集扩展进行重叠社区检测”，载于第 22 届 ACM 信息与知识管理国际会议论文集，2013 年，第 2099–2108 页。[32] V. A. Traag、P. Van Dooren 和 Y. Nesterov，“无分辨率极限社区检测的适用范围狭窄”，Physical Review E，第 84 卷，第 1 期，第 016114 页，2011 年。[33] Y. Ruan、D. Fuhry、J. Liang、Y. Wang 和 S. Parthasarathy，《社区发现：简单且可扩展的方法》，人机交互系列。Springer International Publishing，2015 年，第 23–54 页。[34] J. Zeng 和 H. Yu，“大规模图社区检测的有效统一优化”，载于 2019年IEEE大数据国际会议（大数据）。IEEE，2019 年 12 月，第 475–482 页。[35] ——，“面向可扩展、高质量社区检测的分布式Infomap算法”，载于第 47 届国际并行处理会议论文集——ICPP 2018。美国纽约州纽约市：ACM Press，2018 年 8 月，第 1–11 页。[36] M. A. M. Faysal 和 S. Arifuzzaman，“采用信息论方法进行大型网络中的分布式社区检测”，载于 2019年IEEE大数据国际会议（大数据）。IEEE，2019 年 12 月，第 4773–4782 页。[37] M. Rosvall、D. Axelsson 和 C. T. Bergstrom，“地图方程”，《欧洲物理学杂志专题》，第 178 卷，第 1 期，第 13–23 页，2009 年 11 月。[38] L. Rita，“Infomap算法：一种社区发现算法”，2020 年 4 月。 [在线]。网址：https://towardsdatascience.com/infomap-algorithm-9b68b7e8b86[39] T. B. D. P. Aditama 和 A. Sn，“ 使用遗传算法确定社交网络中的社区结构和模块度 ”，《印度尼西亚计算与控制论系统期刊》，第 14 卷，第 219–230 页，2020 年。

[40] A. K. Ghoshal、N. Das、S. Bhattacharjee 和 G. Chakraborty，“一种用于大型网络社区检测的快速并行遗传算法方法”，载于 2019 年第 11 届通信系统与网络国际会议（COMSNETS）。IEEE，2019 年 1 月，第 95–101 页。[41] Y.-C. Lu 和 G. Chakraborty，“使用多目标优化遗传算法提高图聚类效率”，《应用科学与工程国际期刊》，第 17 卷，第 2 期，第 157–173 页，2020 年 6 月。[42] V. D. Blondel、J.-L. Guillaume、R. Lambiotte 和 E. Lefebvre，“大型网络中社区的快速展开”，《统计力学杂志：理论与实验》，第 2008 卷，第 10 期，第 P10008 页，2008 年 10 月。[43] A. Jain、R. Nasre 和 B. Ravindran，“DCEIL：基于CEIL分数的分布式社区检测”，载于 2017 年 IEEE 第 19 届高性能计算与通信国际会议、IEEE第15届智慧城市国际会议、IEEE第3届数据科学与系统国际会议（HPCC/SmartCity/DSS）。IEEE，2017 年 12 月，第 146–153 页。[44] P. Chopade 和 J. Zhan，“一种使用博弈论建模进行大型网络社区检测的框架”，《IEEE大数据汇刊》，第 3 卷，第 3 期，第 276–288 页，2017 年 9 月。[45] A. Lancichinetti 和 S. Fortunato，“社区检测算法：比较分析”，Physical Review E：统计、非线性与软物质物理，第 80 卷，第 5 期（第 2 部分），第 056117 页，2009 年 11 月。[46] M. Halappanavar、H. Lu、A. Kalyanaraman 和 A. Tumeo，“使用 Grappolo 实现可扩展的静态和动态社区检测”，载于 2017年IEEE高性能极限计算会议（HPEC）。IEEE，2017 年 9 月，第 1–6 页。[47] S. Ghosh、M. Halappanavar、A. Tumeo 和 A. Kalyanarainan，“图聚类模块度优化方法的扩展能力与质量”，载于 2019年IEEE高性能极限计算会议（HPEC）。IEEE，2019 年 9 月，第 1–6 页。[48] J. Zeng 和 H. Yu，“大规模图上的并行模块度社区检测”，载于 2015年IEEE集群计算国际会议。IEEE，2015 年 9 月，第 1–10 页。[49] S. Ryu 和 D. Kim，“使用改进的Louvain算法快速检测大规模图数据中的社区”，载于 2016年IEEE第18届高性能计算与通信国际会议、IEEE第14届智慧城市国际会议、IEEE第2届数据科学与系统国际会议（HPCC/SmartCity/DSS）。IEEE，2016 年 12 月，第 1442–1445 页。[50]T. Zeitz，“使用MapReduce实现分布式图聚类工程化”，2017 年 5 月。 [在线]。网址：https://i11www.iti.kit.edu/ media/teaching/theses/ma-zeitz-17.pdf[51] S. Kolodziej、M. Aznaveh、M. Bullock、J. David、T. Davis、M. Henderson、Y. Hu 和 R. Sandstrom，“SuiteSparse矩阵集合网站界面”，《开源软件杂志》，第 4 卷，第 35 期，第 1244 页，2019 年 3 月。[52]J. Leskovec，“CS224W：图机器学习 — 2021 — 第 13.3 讲 - Louvain 算法”。 [在线]。网址：https://www.youtube.com/watch?v=0zuiLBOIcsw