近似最优的分布式博弈论介数中心度算法

王烨飞 华强胜 高文杰 金海

(大数据技术与系统国家地方联合工程研究中心，武汉 430074)
(服务计算技术与系统教育部重点实验室，武汉 430074)
(集群与网格计算湖北省重点实验室，武汉 430074)
(华中科技大学计算机科学与技术学院，武汉 430074）

摘 要 传统的介数中心度仅依赖通过节点的最短路径数量度量节点重要程度，但在多节点失效场景下，可能无法准确反映节点对网络整体功能的影响。相比之下，基于博弈论的介数中心度则能够考虑节点在不同节点组中的贡献，从而为多节点失效场景下最小化失效影响提供参考。其中常用的一类中心度为基于半值的介数中心度。然而，基于半值的介数中心度的集中式算法在无权图中的复杂度可达 $O(n^4)$，其中n为网络的节点数，这使得传统的单机计算方法在大规模网络中面临挑战。本文提出了一种基于半值的介数中心度分布式算法（MR-SMBC算法），轮数复杂度为$O(n)$。该算法通过解决中心度计算过程中边际贡献导致的顶点重复遍历问题，相较于基础算法实现了线性级别别的轮数复杂度降低。MR-SMBC算法基于CONGEST模型设计，该模型限制每轮网络节点传输的消息大小为$O(logn)$比特。此外，本文通过通信复杂度问题的归约，证明了基于半值的介数中心度问题的轮数复杂度下界为$\Omega(n/logn+D)$，其中D为网络的直径长度。这一结果表明MR-SMBC 算法在CONGEST模型下实现了近似最优的计算效率。本文通过在多个真实世界图和人工生成图上的实验，测试了MR-SMBC 算法的可扩展性，并评估了通信与计算时间比率等性能指标。实验结果表明，与基础算法相比，MR-SMBC 算法能够实现线性级别的时间加速。

关键词 图算法；分布式算法；CONGEST模型；图中心度；网络分析

中图法分类号 TP309 DOI 号: * 投稿时不提供 DOI 号

Nearly Optimal Distributed Algorithm for Computing Game-Theoretic Betweenness Centralities

WANG Yefei HUA Qiang-Sheng GAO Wenjie JIN Hai

(National Engineering Research Center for Big Data Technology and System, Wuhan 430074)
(Service Computing Technology and System Lab, Wuhan 430074)
(Cluster and Grid Computing Lab, Wuhan 430074)
(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract

Conventional betweenness centrality measures the importance of a node in a network by counting the number of shortest paths that pass through it. While this approach is widely used in various fields, it has limitations, especially in scenarios involving multi-node failures. In such cases, traditional betweenness centrality may not accurately capture the true impact of a node on the network's overall functionality.

收稿日期： －－；最终修改稿收到日期： －－. 本课题得到新一代人工智能国家科技重大专项（2022ZD0115301）和国家自然科学基金项目（62372202）资助. 王烨飞, 博士研究生, 主要研究方向为分布式算法和图算法.E-mail: wangyefei@hust.edu.cn. 华强胜(通信作者), 博士, 教授, 中国计算机学会（CCF）杰出会员 (25955D), 主要研究方向为并行与分布式计算理论与算法.E-mail: qshua@hust.edu.cn. 高文杰, 硕士研究生, 主要研究方向为分布式算法和图算法. 金海, 博士, 教授, 中国计算机学会（CCF）会士(05193F). 主要研究领域为计算机体系结构、计算系统虚拟化、集群计算和云计算、网络安全、对等计算、网络存储与并行 I/O 等. 第 1 作者手机号码: 15038180360.

This issue becomes particularly evident when multiple high-centrality nodes are located along the same shortest paths, leading to redundant protective measures and inefficient resource allocation. Essentially, relying solely on individual node contributions may result in suboptimal strategies for mitigating the impact of failures in large, complex networks. To address this limitation, game-theoretic betweenness centralities were introduced, which take into account the contributions of nodes within different groups. This approach allows for a more comprehensive understanding of the network, as it considers the collective influence of nodes in various groups, rather than just the individual importance of each node. One well-known type of game-theoretic betweenness centrality is the semi-value-based betweenness centrality. However, calculating this centrality using centralized algorithms is computationally expensive, with a time complexity of $O(n^4)$ in unweighted graphs, where n is the number of nodes in the network. This complexity poses significant challenges for applying traditional single-machine computation methods to large-scale networks, as the time and memory requirements grow quickly with the network size. In this paper, we present several contributions to overcome these challenges and improve the efficiency of betweenness centrality computations. First, we build on the existing centralized Shapley value-based betweenness centrality algorithm and propose a version of this algorithm tailored for the CONGEST model, which we call the SPBC algorithm. The CONGEST model, which is commonly used in distributed computing, allows for more efficient communication between nodes in a network. Building on the SPBC algorithm, we then design a distributed algorithm for semi-value-based betweenness centrality, named MR-SMBC. The MR-SMBC algorithm reduces the round complexity of the traditional approach, reducing it from quadratic to linear, with a round complexity of $O(n)$. This reduction is achieved by addressing the issue of repeated node traversal during the centrality computation process, which occurs due to the marginal contributions of nodes in the centralized version. Second, we derive lower bounds on the round complexity for solving Shapley value-based and semi-value-based betweenness centrality problems within the CONGEST model. Through the reduction of the communication complexity problem, we establish that the round complexity lower bounds for these problems are $\Omega(n/logn + D)$, where D is the network diameter. This theoretical result demonstrates that our MR-SMBC algorithm achieves near-optimal distributed efficiency in the CONGEST model. Third, we conduct extensive experiments to evaluate the performance of the MR-SMBC algorithm. We use a variety of real-world and synthetic graph datasets to assess the algorithm's performance. The experimental tests include the scalability of the algorithm, the impact of source batch sizes, and the ratio of communication time to computation time. The experimental results indicate that the MR-SMBC algorithm achieves linear-level time acceleration compared to the trivial semi-value-based betweenness centrality algorithm.

Keywords Graph algorithms; Distributed algorithms; The CONGEST model; Graph centrality; Network analysis.

1 引言

中心度是图论和网络分析中的一个重要概念，广泛用于衡量网络中单个节点的重要性。通过计算节点的中心度，可以识别网络中的关键节点，这些节点在信息传播、资源分配和网络结构优化等方面起着至关重要的作用。节点的中心度高低反映了其在网络中的连接程度、影响力和核心地位，是分析社交网络、交通网络以及各种复杂系统的重要工具。

除了单个节点的重要性外，集合的整体重要性也是一个关键指标。例如，在金融领域，中心度有助于识别关键市场参与者，为投资策略和风险管理提供参考。某些市场参与者可能隶属于较大的金融机构，因此，在监控和保护这些机构时，考虑整个机构的重要性具有重要意义。然而，分析一个节点集合的重要性时，仅仅将该集合内各个节点的中心度相加，通常无法准确反映该集合在网络中的整体重要性。为了解决这一问题，群体中心度的概念被提出。群体中心度在汇总集合内节点影响力的同时，避免了对集合内多个节点影响的重复计算。

在一些可能存在多节点失效场景中，如金融网络攻防、流行病传播控制和网络脆弱性分析等，失效节点往往具有不可预测性。在应对这种不可预测的失效时，仅仅将传统中心度较高的节点作为保护对象，不一定能最大程度上减少损失。而

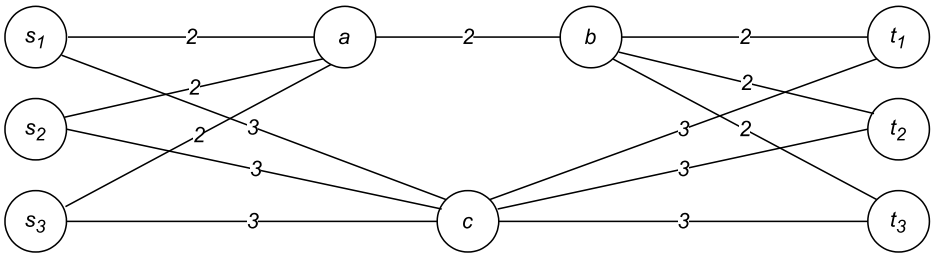

图 1 受限介数中心度实例

结合合作博弈论和群体中心度的博弈论中心度，已经被证明在多种真实网络和模拟网络中能更好地保护网络免受节点故障的影响[1]。在博弈论中心度中，图中的每个节点被视为一个玩家，而节点集合则被视为一个联盟。博弈论中心度通过将联盟的群体中心度作为其价值评价函数，并结合合作博弈论中的分配规则，量化节点在整个网络中的贡献。其中，$Shapley$ 值和半值是较为常见的分配规则。本文讨论的对象为博弈论中心度中的两类中心度：基于 $Shapley$ 值的介数中心度和基于半值的介数中心度。

博弈论中心度综合每个节点在不同节点集合中的贡献，能够考虑多节点失效中可能出现各种情形。相比传统的中心度，博弈论中心度能够为多节点失效场景下保护重要节点提供参考，减少节点失效造成的期望损失。为了更清晰地展示中心度、群体中心度以及基于半值的中心度之间的异同，下面通过一个简单示例进行说明。首先定义一种介数中心度的简单变体——受限介数中心度（Restricted Betweenness Centrality）。节点 v 的受限介数中心度被定义为：

$$\psi^*(v,S,T)=\sum_{s\in S,t\in T}\frac{\sigma_{st}(v)}{\sigma_{st}}$$

其中 S 为源点集合，T 为终点集合，σ_{st} 表示从源点 s 到终点 t 的最短路径数量，而 $\sigma_{st}(v)$ 表示从 s 到 t 的最短路径中经过节点 v 的数量。与介数中心度不同，受限介数中心度仅考虑以 s 为起点且 t 为终点的最短路径。在图 1 中，源点集合 S 包含 s_i，终点集合 T 包含 t_j，其中 $i,j\in\{1,2,3\}$，图中边的权重则表示相邻节点之间的距离。图中一共有9对 s_it_j，且对于每对 s_it_j，他们之间的最短路径有两条，$[s_i,a,b,t_j]$ 和 $[s_i,c,t_j]$，a,b,c 三点都会被其中一条最短路径经过。因此根据受限介数中心度的定义，节点 a,b,c 的中心度是相等的，且均为9/2。一个集合 U 的群体受限介数中心度定义如下：

$$\psi^*(U,S,T)=\sum_{\substack{s\in S\\ t\in T}}\frac{\sigma_{st}(U)}{\sigma_{st}}$$

在这里，$\sigma_{st}(U)$ 表示从 s 到 t 的所有最短路径中，至少经过集合 U 中一个节点的路径数量。

假如直接将集合中顶点的受限介数中心度求和作为集合重要程度评估的标准，那么图 1 中集合 $\{a,b\}$ 和集合 $\{a,c\}$ 的重要性相同。但事实上集合 $\{a,c\}$ 相比于 $\{a,b\}$ 对网络连接更加关键：如果 a 和 c 失效，集合 S 和 T 之间的通信将完全中断；而如果 a 和 b 失效，这两个集合仍然可以通过 c 进行通信。该情况通过群体受限介数中心度表现，$\psi^*(\{a,c\},S,T)>\psi^*(\{a,b\},S,T)$，其能够更准确反映点集在网络中的重要程度。

$$\psi^*_{SM}(v,S,T)=\sum_{U\subset V\setminus\{v\}}p_{|U|}\big(\psi^*(U\cup\{v\},S,T)-\psi^*(U,S,T)\big)$$

其中 $p_{|U|}$ 表示集合 U 对应的权重，由集合大小 $|U|$ 决定。权重根据实际要求调整，但需要满足 $\sum_{0\le k\le n-1}p_k\binom{n-1}{k}=1$，$n$ 为图中节点的数量。直观来说，$\psi^*(U\cup\{v\},S,T)-\psi^*(U,S,T)$衡量节点 v 在受限介数中心度方面对集合 $U\cup\{v\}$ 的贡献。通过将节点 v 对不同集合贡献加权求和，$\psi^*(v,S,T)$ 能够为多节点失效场景下的节点保护提供依据。

例如考虑一种特殊情况，为大小为1的集合设置权重 $p_1=\frac{1}{n-1}$，而对于其他大小的集合，$p_k=0$。在此假设下，敌方均匀攻击网络中的两个节点使得他们失效，$\psi^*_{SM}(v,S,T)$ 反映了通过保护 v 避免其在攻击中失效，相比于不保护 v 的情况，所能保留的受限介数中心度的期望值。在图 1 中，根据上述定义，$\psi^*_{SM}(c,S,T)$ 大于 $\psi^*_{SM}(b,S,T)$、$\psi^*_{SM}(a,S,T)$。这表明即使它们的中心度相等，在面对可能到来的攻击，保护节点 c 存活相比于节点 a 或者 b 存活能够更大程度上保证网络通信效率。

基于博弈论中心度的网络防御优化方案具有广泛的应用潜力。例如金融网络的多节点攻防场景下，攻击者通过恶意软件或黑客手段，可能对金融网络中的多个机构进行攻击，进而造成关键节点的功能受损或失效，这将引发连锁反应，甚至导致整个网络瘫痪。平台需要在部分关键机构预先布置防御措施，以保证受到攻击后关键节点的存活，减少攻击对网络带来的影响。

传统的介数中心度是通过分析单个机构在网络中的关键程度来评估其重要性，但其缺陷在于：多

个具有高介数中心度的机构可能处于相同的最短路径上，导致保护措施过度集中，进而产生资源浪费。群体介数中心度通过度量多个机构的集合在网络中的关键性可以弥补传统方法的不足。然而，面对不可预测的攻击，群体介数中心度的效果仍然无法得到充分保证。博弈论中心度方法则结合合作博弈理论，进一步优化防御策略。在此方法中，群体介数中心度被用作合作博弈的度量标准，通过计算在各种网络攻击情况下，保护某个机构所保留的介数中心度的期望值，从而为平台提供更为高效的资源分配策略。

虽然博弈论中心度在评估风险、规避攻击以及制定策略中具有重要作用，但是在计算部分博弈论中心度的过程中，例如基于半值的介数中心度，由于需要遍历所有可能集合的群体中心度，传统算法的计算复杂度呈指数级增长。

为解决这一问题，Szczepanski等人[1]提出了集中式算法，适用于单机计算基于半值的介数中心度。与传统方法不同，该算法不需遍历所有可能的集合，而是将中心度计算转化为每个节点边际贡献的计算。在无权图中，该算法的计算复杂度为 $O(n^4)$，其中 n 为网络中的节点数量。

尽管集中式算法能有效将计算复杂度降至多项式时间，但在大规模图中，$O(n^4)$ 的复杂度依然会导致单机计算资源消耗巨大，效率低下。与此相比，分布式系统将计算任务分配至多个处理单元，能够并行处理大量数据和复杂计算，从而显著提高计算效率，减轻单机计算压力。采用分布式算法处理基于半值的介数中心度计算问题，能更好地满足实际应用需求，提高计算效率与可行性。本文旨在设计并实现一个针对无权图的分布式基于半值的介数中心度低复杂度算法。本文采用 CONGEST 模型作为分布式计算模型，该模型是经典的分布式通信模型，每条边在每轮只能传输大小为 $O(logn)$ 比特的消息。

尽管分布式系统在处理大规模图的博弈论介数中心度计算问题上具有显著优势，但在设计和分析低复杂度分布式算法时仍面临诸多挑战。上界方面，现有最先进的分布式介数中心度算法在计算博弈论介数中心度时，由于涉及额外的期望计算，需要平方级的轮数复杂度。下界方面，对博弈论介数中心度的计算除了依赖于最短路径数量，还依赖于最短路径长度，这与传统介数中心度不同。这导致传统介数中心度的复杂度下界证明并不适用于博弈论介数中心度，因此需要构造新的证明以获得博弈论介数中心度问题的轮数复杂度下界。

本文的主要贡献包括：

1.基于[1]中描述的基于 $Shapley$ 值的介数中心度集中式算法，本文提出了其在CONGEST模型下的版本，称为SPBC算法。以此为基础，本文设计了一种轮数复杂度为 $O(n)$ 的分布式算法MR-SMBC，用于计算基于半值的介数中心度，相比于集中式算法的简单分布式化将平方级的轮数复杂度降低至线性级。

2.通过通信复杂度问题的归约，我们得到CONGEST模型下基于半值的介数中心度求解问题的轮数复杂度下界，证明MR-SMBC算法达到了近似最优。

3.在多个真实世界图和人工生成图上对MR-SMBC算法的可扩展性进行了测试，并评估了其在通信和计算时间比率等性能指标上的表现。实验结果表明，与现有的基于半值的介数中心度算法相比，MR-SMBC算法在执行时间上实现了线性级别的加速。

第 2 节概述相关工作；第 3 节介绍预备知识，涵盖与图论和博弈论相关的定义；第 4 节提出SPBC算法和MR-SMBC算法，并进行复杂度分析与正确性证明；第 5 节给出下界证明；第 6 节展示实验结果并进行分析；第 7 节总结论文的主要成果，并讨论未来的研究方向。

2 相关工作

Brandes[2]首先提出了介数中心度的高效算法，其核心思想是利用广度优先搜索求解单源最短路径，并结合动态规划的方法来更新每个节点的介数中心度。Hua等人[3]提出了在CONGEST模型下无向无权图的介数中心度计算的下界，并设计了一种轮数复杂度为 $O(n)$ 的分布式介数中心度算法，该算法依赖于全源最短路径（APSP）的 $O(n)$ 轮分布式算法 。在这些上下界的基础上，该算法能够达到近似最优的效果。Hoang等人[4]基于分布式全源最短路径算法[5]，提出了新的CONGEST模型下的介数中心度算法，相较于Hua等人的算法，减少了常数因子的轮数，同时该算法在现实世界图中测试也达成良好的效果。

分布式中心度计算不仅限于CONGEST模型。Behera等人[6]提出了一种基于MapReduce 框架的分布式算法，用于计算社交网络数据中的节点中心度，包括度中心度、接近中心度和特征向量中心度。Crescenzi等人[7]提出了一种基于 Bellman-Ford 算法的分布式介数中心度计算方法，该方法基于距离矢量协议拓展 Bellman-Ford 算法，使得每个节点在一定次数的迭代后获得介数中心度。

此外，许多研究进一步在不同拓扑结构上，采用多种方法对分布式介数中心度问题算法进行了深入探讨。Wang等人[8][9]针对不同类型的网络拓扑（如有向无环图（DAG）和树）提出了特定的介数中心度算法。这些算法的设计考虑到了网络拓扑的特殊性质，从而提高了计算效率。You等人[10]基于线性规划方法提出了计算接近中心度和有向树中的

介数中心度的分布式算法。还有一些研究致力于利用代数工具来进行分布式介数中心度计算，例如Solomonik等人[11]提出的基于稀疏矩阵乘法的通信高效介数中心度算法。Hua 等人[12]综述了多种分布式图性质算法，探讨如何在分布式环境中设计低复杂度算法计算图性质，介绍如何通过通信复杂性归约得到证明分布式图性质问题的轮数复杂度下界。

动态图的分布式介数中心度计算则能够为网络节点或链接失效后网络的关键节点评估提供依据。iCentral[13]是一种创新的增量算法，它将图分解为双连通分量，并在本地处理以实现动态更新。MR-IBC[14]是一个基于MapReduce框架的增量并行介数中心度算法，该算法能够高效处理动态网络中因节点或边的插入与删除而引发的网络结构剧烈变化。Yang等人[15]引入了局部介数中心度的概念并设计了通过动态剪枝更新局部介数中心度的动态维护算法。

Grofman等人[16]首次将合作博弈论应用于网络中节点中心度的分析。Szczepanski等人[17]正式引入了基于$Shapley$值的介数中心度的定义，指出每个联盟的价值源自其集合的介数中心度，而每个节点的中心度度量则基于$Shapley$值的定义获得。他们的另一项工作扩展了基于$Shapley$值和半值的介数中心度的研究，涵盖了加权和无权图，并为其计算设计了多项式时间算法[1]。同时，Tarkowski等人[18]将这一研究扩展到更多类别的博弈论网络中心度，如度中心度和接近中心度的$Banzhaf$指数。尽管已有不少关于博弈论网络中心度的研究，但截至目前，仍未见有研究关注在分布式环境中计算博弈论网络中心度的问题。

3 预备知识

本节将展示系统模型、问题定义及相关符号，以便为后续的分析和方法提供必要的基础。本文聚焦于分布式环境中的拥塞模型（CONGEST模型）。在该模型中，图作为网络的抽象模型，其中每个顶点代表一个网络节点、计算单元或处理器，而图的边则表示这些节点之间的通信链路。CONGEST模型的核心特性是：每个顶点在单个同步轮内只能向其邻居广播大小为$O(logn)$的消息，其中n为图中顶点的数量。

与局部模型（LOCAL模型）相比，CONGEST模型对顶点之间的带宽施加了限制，这导致每次通信只有有限的消息交换，使得低复杂度算法设计更具挑战性。本文轮数复杂度被视为评估算法效率的关键指标。此外，本文也关注消息复杂度，其关注算法在通信过程中交换的信息量。

首先介绍与图和介数中心度相关的定义。考虑一个无权图$G(V,E)$，其中顶点集为V，边集为E，图中顶点和边的数量分别为$n=|V|$和$m=|E|$。本文主要讨论有向无权图和无向无权图中的算法设计。在有向图中，顶点v的入邻居和出邻居分别记作$N_{in}(v)$和$N_{out}(v)$；而在无向图中，这两个符号均表示顶点v的邻居。

定义1.（介数中心度）对于给定的顶点$v\in V$，如果$bc(v)$是v的介数中心度，则满足以下公式：

$$bc(v)=\sum_{s,t\in V\setminus\{v\}}\frac{\sigma_{st}(v)}{\sigma_{st}}$$

其中，σ_{st}表示从顶点s到顶点t的最短路径数量，而$\sigma_{st}(v)$是从顶点s到顶点t的所有最短路径中经过v的路径数量。

介数中心度能够衡量一个顶点在整个网络中的重要性。而群体介数中心度则能够评估一组顶点在网络中的整体影响力，它不仅关注单个顶点的作用，还考察了这一组顶点对网络中其他顶点的潜在影响。

定义2.（群体介数中心度）对于给定的顶点集$U\subset V$，如果$bc(U)$是U的群体介数中心度，则满足以下公式：

$$bc(U)=\sum_{s,t\in V\setminus U}\frac{\sigma_{st}(U)}{\sigma_{st}}$$

其中，$\sigma_{st}(U)$是从顶点s到顶点t的所有最短路径中，至少经过顶点集U中一个顶点的路径数量。

接下来介绍与博弈论相关的定义。$Shapley$值是合作博弈中常用的价值分配方法，其广泛应用于经济学、人工智能等领域。它通过考虑每个玩家在所有可能联盟中的边际贡献，公平地计算合作博弈中各个参与者的贡献。然而，在一些实际问题中，$Shapley$值可能无法充分反映参与者之间的复杂关系或特定的约束条件。为了解决这一问题，提出了半值（semi-value）概念。半值作为$Shapley$值的扩展，在计算玩家贡献时引入了特定的权重或约束条件，使其能够适应更加多样化的合作博弈场景。

定义3.（半值）给定一个博弈$(A,value)$，其中A为玩家集合，$value$为玩家集合的价值函数。对于玩家$i\in A$，如果满足以下条件，则称$sv(i)$为i的半值：

$$sv(i)=\sum_{U\subset A\setminus\{i\}}p_{|U|}[value(U\cup\{i\})-value(U)]$$

其中，$p_{|U|}$表示大小为$|U|$的子集的权重，且满足$\sum_{0\le k\le|A|-1}p_k\binom{|A|-1}{k}=1$。直观上看，半值能够衡量博弈中每个玩家的贡献，并能通过设置不同的权重，在各种复杂情况下将总价值分配给各个参与者。当集合U的权重函数设置为$(|A|\binom{|A|-1}{|U|})^{-1}$时，所有集合的权重相同，半值退化为$Shapley$值。

定义4.（基于半值的介数中心度）对于给定的顶点$v\in V$，$bc_{SM}(v)$表示顶点v的基于半值的介数中心度，其定义为：

$$bc_{SM}(v)=\sum_{U\subset V\setminus\{v\}} p_{|U|}(bc(U\cup\{v\})$$

类似可以得到顶点 v 的基于 $Shapley$ 值的介数中心度 $bc_{SP}(v)$，但由于篇幅限制，此处省略相关公式。由于集合 U 对于顶点 v 的权重大小完全取决于该集合大小，因此 $bc_{SM}(v)$ 也可以表示为下面的形式：

$$bc_{SM}(v)=\sum_{k\in[0,n-1]}\frac{P_k}{\binom{n-1}{k}}\Big(\sum_{\substack{U\subset V\setminus\{v\}\\|U|=k}}(bc(U\cup\{v\})-bc(U))\Big)$$

其中，$P_k=p_k\binom{n-1}{k}$。如果直接根据上述公式计算 $bc_{SM}(v)$，由于需要计算所有 U 的介数中心度，这将至少需要指数级的时间复杂度。为了在多项式时间内计算博弈论中的介数中心度，Szczepanski等人[1]从顶点 v 的边际贡献角度进行了分析，并给出了 $bc_{SM}(v)$和 $bc_{SP}(v)$的另一种公式。

$$bc_{SM}(v)=\sum_{k\in[0,n-1]}P_k\Big(\sum_{\substack{s,t\in V\setminus\{v\}\\n-d_s(t)\le k}}\frac{\sigma_{st}(v)}{\sigma_{st}}f_{SM}(d_s(t),k)+\sum_{\substack{s\in V\setminus\{v\}\\n-d_s(v)\le k}}g_{SM}(d_s(v),k)\Big)\qquad(1)$$

$$bc_{SP}[v]=\sum_{s,t\in V\setminus\{v\}}\frac{\sigma_{st}[v]}{\sigma_{st}}f_{SP}(d_s[t])+\sum_{s\in V\setminus\{v\}}g_{SP}(d_s[v])$$

其中 $f_{SP}(d)=\frac{1}{d}$，$g_{SP}(d)=\frac{2-d}{2d}$，

$$f_{SM}(d,k)=\frac{(n-d)!\,(n-k-1)!}{(n-d-k)!\,(n-1)!}$$

$$g_{SM}(d,k)=f_{SM}(d,k)+\frac{k-n+1}{n-1}$$

表1列出了本文在后续算法伪代码、描述和证明中使用的符号说明。

表1 符号说明

符号	定义
n	图 G 中的顶点数量
m	图 G 中的边数量
D	图 G 的直径
$N_{in}(v)/N_{out}(v)$	顶点 v 的入/出邻居
$d_s(v)$	s 到 v 的距离
σ_{sv}	s 到 v 最短路径数量
$f_{SM}(d,k),g_{SM}(d,k)$	$bc_{SM}(v)$ 的辅助函数
$f_{SP}(d,k),g_{SP}(d,k)$	$bc_{SP}(v)$ 的辅助函数

4 算法与分析

本节介绍基于 $Shapley$ 值的介数中心度分布式算法 SPBC 和基于半值介数中心度的分布式算法 MR-SMBC，并给出它们的正确性分析、轮数复杂度和消息复杂度分析。

4.1 基于 $Shapley$ 值的分布式BC算法SPBC

本小节提出基于 $Shapley$ 值的介数中心度分布式算法SPBC并简要介绍算法。

算法 1 计算基于 $Shapley$ 值的介数中心度。该算法分为两个阶段：第一阶段为分散阶段，计算全源最短路径（APSP），使得每个顶点都能够知道其他顶点到自己的最短距离，即算法 1 第 1 行的 Distributed-APSP子程序；第二阶段为汇聚阶段，根据算法1的结果，累积计算基于 $Shapley$ 值的介数中心度，即算法 1 的第 $2-17$ 行。算法 1 的第 $18-20$ 行为每个顶点 v 的本地计算，不涉及顶点间通信。

算法 1． SPBC

输入：$G(V,E),f_{SP},g_{SP}$

输出：对于 $v\in G$，顶点 v 知道 $bc_{SP}(v)$

1 $(d_s(v),\sigma_{sv},T_s(v))_{s\in V}\leftarrow$*Distributed-APSP* (G)

2 计算图 G 的顶点数 n、直径 D，以及 最大轮数 $T_{max}\leftarrow\max(T_p(q))_{(p,q)\in V^2}$

3 向每个顶点广播 n,D,T_{max}

4 设置全局时钟 $tp\leftarrow T_{max}$

5 初始化 $bc_{SP}(v),\phi_s(v),\delta_s(v)\leftarrow 0$，对于每个$s\in V\setminus v$

6 FOR $tp\leftarrow T_{max}$ 到 0 DO

7 IF $\exists T_s(v)=tp$

THEN

8 $\phi_s(v)\leftarrow 1/\sigma_{sv}(\delta_s(v)+f_{SP}(d_s(v)))$

9 v 向 $N_{in}(v)$ 发送消息$msg(d_s(v),s,\phi_s(v))$

10 ENDIF

11 WHILE v 从 u 接收 $msg(d_s(u),s,\phi_s(u))$ DO

12 IF $d_s(u)+1=d_s(v)$

THEN

13 $\phi_s(v)\leftarrow\phi_s(v)+\phi_s(u)$

14 $\delta_s(v)\leftarrow\delta_s(v)+\sigma_{sv}(\phi_s(u))$

15 ENDIF

16 ENDWHILE

17 ENDFOR

18 For $\forall s\in V\setminus\{v\}$ DO

19 $bc_{SP}(v)\leftarrow bc_{SP}(v)+\delta_s(v)+g_{SP}(d_s(v))$

20 ENDFOR

算法 2 展示算法 1 第一阶段的Distributed-APSP子程序，调用分布式Bellman-Ford算法[4]来解决所有对的最短路径问题。其工作原理如下：

每个顶点 v 维护一个优先队列 L_v，该队列包含的元素为 $(d_s(v),s)$，其中 s 表示源点，$d_s(v)$表示从源点 s 到顶点 v 的临时最短路径长度。$p_v^r(d_s(v),s)$ 表示在第 r 轮时，元素 $(d_s(v),s)$ 在优先队列 L_v 中的位置。

算法 2 的第 $5-8$ 行为顶点 v 作为消息发送者时的子程序：算法初始化时钟 r 后，一旦存在某个 w，使得 $d_w(v)+p_v^r(d_s(v),s)=r$，顶点 v 发送 msg 到所有出邻居并记录当前轮数 $T_w(v)$。

算法 2 的第 $9-21$ 行为顶点 v 作为消息接收

者时的子程序：当顶点 v 接收到其他顶点的消息 msg 时，若能够降低 $d_w(v)$，顶点 v 会相应地更新 L_v。此外，为了介数中心度的计算，算法还需要维护 σ_{wv}。

当算法结束时，L_v记录所有其他源顶点到顶点 v 的最短路径长度。此外，顶点 v 在算法 1 结束时，还将记录从 s 到 v 的最短路径数量 σ_{sv} 和消息发送轮数 $T_s(v)$，这些参数用于算法 2 第二阶段介数中心度的计算。

需要强调的是，算法 2 不需要传输或存储每个顶点在最短路径上的前驱节点。这避免了在稠密图中存储消耗过大的问题。

算法 2. Distributed-APSP

输入：$G(V,E)$

输出：对于 $v \in G$，顶点 v 知道$d_s(v),\sigma_{sv},T_s(v)$

1 $L_v \leftarrow \{(0,v)\}$

2 初始化 $d_s(v),\sigma_{sv},T_s(v) \leftarrow 0$，对每个$s \in V \setminus \{v\}$

3 初始化全局时钟 $r \leftarrow 0$

4 FOR $r \leftarrow 0$ 到 $2n$ DO

5 WHILE $\exists w'$使$d_w(v) + p_v^r(d_w(v),w) = r$ DO

6 v 发送 $msg(d_w(v),w,\sigma_{wv})$ 到 $N_{out}(v)$

7 v 记录发送轮数 r 为 $Tw(v)$

8 ENDWHILE

9 WHILE v 接收 u 的 $msg(d_w(u),w,\sigma_{wu})$ DO

10 IF $\exists(d_w(v),w) \in L_v$ 使 $d_w(u)+1 = d_w(v)$ THEN

11 $\sigma_{wv} = \sigma_{wv} + \sigma_{wu}$

12 ELSEIF $\exists(d_w(v),w) \in L_v$ 使$d_w(u)+1 < d_w(v)$ THEN

13 $L_v \leftarrow L_v \setminus \{(d_w(v),w)\}$

14 $L_v \leftarrow L_v \cup \{(d_w(u)+1,w)\}$

15 $\sigma_{wv} \leftarrow \sigma_{wu}$

16 ELSEIF $(d_w(v),w) \notin L_v$ THEN

17 $L_v \leftarrow L_v \cup \{(d_w(u)+1,w)\}$

18 $\sigma_{wv} \leftarrow \sigma_{wu}$

19 ENDIF

20 ENDWHILE

21 ENDFOR

算法 1 的第 2 行到第 17 行描述汇聚阶段。首先任选图中的一个顶点作为主顶点执行第 2 行到第 4 行：

算法 2 中每个顶点 v 记录其发送包含源点 s 的消息 msg 时的轮数 $T_s(v)$。主顶点从这些轮数中提取最大值，记作 $T_{\max}$。同时主顶点建立一个以轮数为单位的全局时钟 tp，从 $T_{\max}$开始倒计时至0。

每当顶点 v 检测到时钟 tp 等于自己保存的某个 $T_s(v)$时，计算中间变量 $\phi_s(v)$并将 $\phi_s(v)$ 加入消息 msg 并将其发送给 v 的入邻居（参考算法 1 的第 $7-10$ 行）。

直观理解 $T_s(v)$的作用：在分散阶段，也就是算法 2，每个顶点 v 记录消息发送时间 $T_s(v)$，构成时间表；在汇聚阶段，每个顶点参照翻转的时间表决定何时发送消息。这一操作确保算法的正确性，同时避免网络拥塞。具体细节将在定理 1 中进行证明。

当顶点 v 从其出邻居 u 接收到 msg 时，它根据 $d_s(v)$ 和 $d_s(u)$检查 u 是否是源点 s 到顶点 v 的最短路径前驱。如果是，顶点 v 计算依赖 $\phi_s(v)$ 的值，并更新中间变量 $\phi_s(v)$（参考算法 1 的第 $11-16$ 行）。通过本地计算，最终得到基于 $Shapley$ 值的介数中心度 $bc_{SP}(v)$（参考算法 1 的第 $18-20$ 行）。算法展示有向图的情况，对于无向图需要在第 19 行计算 $bc_{SP}(v)$ 时对 $\delta_s(v)$ 值减半以避免重复计算同一路径的介数。部分情况下顶点之间的最短路径数量可能达到指数量级，CONGEST 模型的信道限制将导致消息 msg 的发送无法在单轮内完成。采用浮点表示形式来近似最短路径的数量[3]可以通过引入适度的相对误差，确保消息大小维持 $O(logn)$。

定理1. 在算法 1 的最终输出中，变量 $bc_{SP}(v)$ 等于图 G 中顶点 v 的基于 $Shapley$ 值的介数中心度。

证明. Hoang等人[4]已经证明了Distributed-APSP的正确性，并证明了对于每个源点 s，顶点 v 的最短路径前驱 u 记录的发送时间小于顶点 v 的记录的发送时间，即 $T_s(u) < T_s(v)$。

这里只需要证明算法 1 的第二阶段可以正确利用APSP问题的结果获得基于 $Shapley$ 值的介数中心度。根据基于 $Shapley$ 值的介数中心度的定义，源点 s 基于 $Shapley$ 值的介数中心度可以表示为公式(2)的形式。

在公式(2)中，其中顶点 v 的累积 $\delta_s(v)$的表示具体展示在公式(3)。其中集合 $Suc_s(v)$ 的定义如下：$Suc_s(v)$ 包含顶点 u，如果 v 存在于 s 和 u 之间的某条最短路径上，并且 u 是 v 的出邻居。直观上，$Suc_s(v)$ 是顶点 v 的以 s 为源点的最短路径的后继集合。参考Brandes算法[2]，可以将公式(3)展

$$bc_{SP}(v) = \sum_{s\in V\setminus\{v\}}\Big(\sum_{t\in V\setminus\{s,v\}}\frac{\sigma_{st}(v)}{\sigma_{st}}f_{SP}(d_s(t)) + g_{SP}(d_s(v))\Big) = \sum_{s\in V\setminus\{v\}}(\delta_s(v) + g_{SP}(d_s(v))) \quad (2)$$

$$\delta_s(v) = \sum_{t\in V\setminus\{s,v\}}\frac{\sigma_{st}(v)}{\sigma_{st}}f_{SP}\big(d_s(t)\big) \quad (3)$$

$$= \sum_{u\in Suc_s(v)}\Big(\frac{\sigma_{su}(v)}{\sigma_{su}}f_{SP}(d_s(u)) + \frac{\sigma_{sv}}{\sigma_{su}}\sum_{w\in V\setminus\{s,v,u\}}\frac{\sigma_{sw}(u)}{\sigma_{sw}}f_{SP}(d_s(w))\Big) \quad (4)$$

$$= \sum_{u\in Suc_s(v)}\frac{\sigma_{sv}}{\sigma_{su}}\big(f_{SP}(d_s(u)) + \delta_s(u)\big) \quad (5)$$

开为公式(4)，进一步转化为$\delta_s(v)$的递归形式，在公式(5)。

这里现在采用归纳法证明，当 $t=T_s(v)+1$ 时，$\delta_s(v)$满足公式(5)。首先证明基本情况成立。假设某个源顶点 s 到任意其他顶点的最大距离为 D_s。对于到源顶点 s 距离为 D_s 的顶点 w_{D_s}，由于顶点 w_{D_s}其是距离源顶点 s 最远的顶点，根据算法1的第11-16行，w_{D_s}不会发送任何 msg 也不会对 $\delta_s(v)$进行更新。因此在算法的执行的整个过程中，$\delta_s(w_{D_s})=0$。另一方面没有 s 到其他点的最短路径经过 w_{D_s}，因此 $Suc_s(v)$是空集，根据公式(5)， $\delta_s(w_{D_s})=0$。

基本情况已被证明，接下来的归纳步骤考虑与源顶点 s 距离为 $h+1$ 的顶点 w_{h+1}，其中 $h\in[0,D_s)$，假设当 $t=T_s(w_{h+1})+1$，$\delta_s(w_{h+1})$ 已经满足公式(5)。这里需要证明在 $t=T_s(w_h)+1$时，对于所有与源顶点 s 距离为h的前驱 w_h，$\delta_s(w_h)$也满足公式(5)。对于 $w_{h+1}\in Suc_s(w_h)$，w_{h+1}在 $t=T_s(w_{h+1})+1$ 时，$\delta_s(w_{h+1})$ 满足公式(5)，在 $t=T_s(w_{h+1})$ 时发送消息。由于算法 2 中的发送时间的设置，$T_s(w_h)<T_s(w_{h+1})$。因此在算法 2 的倒计时中，当 $t=T_s(w_h)+1$，顶点 w_h 已经接收到所有对于源点 s 的后继发送的消息。并且根据归纳假设，这些后继在发送消息之前就已经满足公式(5)，因此算法 1 的第 8 行，第 $13-14$ 行，$\delta_s(w_h)$ 能被正确计算。也就是说，在$t=T_s(w_h)+1$ 时，$\delta_s(w_h)$ 也满足公式(5)，完成归纳证明。顶点 v 的累积 $\delta_s(v)$ 正确计算后，算法 1 的第 $18-20$ 行的本地计算参考公式(2)计算 $bc_{SP}(v)$。

证毕.

定理2. 算法 1 的轮数复杂度为 $O(n)$，消息复杂度为$O(nm)$ 。

证明. 算法 2 求解全源最短路径问题需要 $O(n)$ 轮，因此有 $T_{\max}=O(n)$。算法 1 的第 $2-3$ 行的轮数为 $O(D)$，而第 $6-17$ 行的循环执行次数为 $T_{\max}=O(n)$，且每次循环所需轮数为 $O(1)$。最后，第 $18-21$ 行的本地计算不涉及基于轮数的通信。因此，算法 1 可以在 $O(n)$ 轮内完成对 $bc_{SP}(v)$ 的计算。

关于消息复杂度，算法 2 每个轮数中最多需要发送 $O(m)$ 条消息，因此消息复杂度为 $O(nm)$。由于全局时钟的调度，算法 1 中的每个消息实际上也对应算法 2 中的一个消息。因此，算法 1 的消息复杂度为 $O(nm)$。

证毕.

4.2 基于半值的分布式BC算法SMBC

本节首先介绍一种用于计算基于半值的介数中心度的基本算法，接着提出改进的MR-SMBC算法，并给出其正确性证明及复杂度分析。

算法 3. SMBC

输入：$G(V,E),P_i,f_{SM},g_{SM}$

输出：对于 $v\in G$，顶点 v 知道 $bc_{SM}(v)$

1 计算图 G 的顶点数 n

2 FOR $k\leftarrow 0$ 到 $n-1$ DO

3 将$SPBC$算法中的 $f_{SP}(d)$ 和$g_{SP}(d)$ 替换为 $f_{SM}(d,k)$ 和 $g_{SM}(d,k)$

4 $SPBC(G)$

5 ENDFOR

基于半值的介数中心度与基于$Shapley$值的介数中心度之间的关键区别在于辅助函数 f和辅助函数 g 的输入参数。基于 $Shapley$ 值的介数中心度的辅助函数中，输入参数仅包含距离 d，而基于半值的介数中心度的辅助函数则包含了额外的参数 k。为了得到正确的结果，参数 k 最多可能从0遍历到 $n-1$。因此，为了实现基于半值的介数中心度的分布式计算，可以直接采用一种基础方法，将算法 1 重复执行 n 次，每次迭代对应于辅助函数 f 和 g 的不同 k 值，该算法在算法 3 中展示。需要说明的是，根据公式(1)，算法 3 的替换需要还验证 $n-d_s(t)\le k$ 和 $n-d_s(t)\le k$，空间限制没有在伪代码中表现。然而，这种方法在理论上需要 $O(n^2)$ 轮才能完成计算。

为了降低过高的轮数复杂度，我们希望改进的分布式算法能够进行更多低时间成本的本地计算，从而减少高时间成本的消息传递，即减少轮数复杂度。基于这一想法，我们对公式(1)进行了变形，得到如公式(7)所示的结果。累积 $\delta_s(v,n)$ 的定义如公式(8)所示，公式(9)展示其递归形态。

上述公式变换的直接目的是公式(1)第一项中两个求和循环的顺序，分别是源点 s 的 s 循环和半值辅助函数 f 计算中的 k 循环。最终的目的是让每个顶点的辅助函数的计算本地执行，不需要消耗额外的轮数复杂度。基于公式(7)，本文提出MR-SMBC（算法 4 ）。算法 4 保留了算法 1 的框架，假设已知辅助函数 f 和 g 以及 G 的大小。一旦到达特定的时间点，顶点 v 将本地首先完成 k 循环计算（参

$$bc_{SM}(v)=\sum_{k\in[0,n-1]}P_k\Big(\sum_{\substack{s,t\in V\setminus\{v\}\\ n-d_s(t)\le k}}\frac{\sigma_{st}(v)}{\sigma_{st}}f_{SM}(d_s(t),k)+\sum_{\substack{s\in V\setminus\{v\}\\ n-d_s(v)\le k}}g_{SM}(d_s(v),k)\Big) \tag{6}$$

$$=\sum_{s\in V\setminus\{v\}}\delta_s(v,n)+\sum_{k\in[0,n-1]}\sum_{\substack{s\in V\setminus\{v\}\\ n-d_s(v)\le k}}P_k g_{SM}(d_s(v),k) \tag{7}$$

$$\delta_s(v,n)=\sum_{k\in[0,n-1]}P_k\Big(\sum_{\substack{t\in V\setminus\{s,v\}\\ n-d_s(t)\le k}}\frac{\sigma_{st}(v)}{\sigma_{st}}f_{SM}(d_s(t),k)\Big)=\sum_{t\in V\setminus\{s,v\}}\frac{\sigma_{st}(v)}{\sigma_{st}}\Big(\sum_{\substack{k\in[0,n-1]\\ n-d_s(t)\le k}}P_k f_{SM}(d_s(t),k)\Big) \tag{8}$$

$$=\sum_{u\in Suc_s(v)}\frac{\sigma_{sv}}{\sigma_{su}}\Big(\sum_{\substack{k\in[0,n-1]\\ n-d_s(u)\le k}}P_k f_{SM}(d_s(u),k)+\delta_s(u,n)\Big) \tag{9}$$

考算法 4 的第 $5-9$ 行），再继续传递消息继续完成算法1的汇聚阶段。这项改进降低轮数复杂度，消除了算法 3 中的 n 次冗余遍历。

算法 4. MR-SMBC

输入：$G(V,E),P_i,f_{SM},g_{SM}$

输出：对于$v\in G$，顶点 v 知道$bc_{SM}(v)$

1 重复 *SPBC* 算法的第$1-4$ 行

2 初始化$bc_{SM}(v),\phi_s(v,n),\delta_s(v)\leftarrow 0$，对每个 $s\in V\setminus\{v\}$

3 FOR $tp\leftarrow T_{max}$ 到 0 DO

4 IF $\exists T_s(v)=tp$ THEN

5 FOR $k\leftarrow 0$ 到 $n-1$ DO

6 IF $n-d_s(v)\le k$ THEN

7 $tem_s(v)\mathrel{+}=P_k f_{SM}(d_s(v),k)$

8 ENDIF

9 ENDFOR

10 $\phi_s(v,n)\leftarrow 1\backslash\sigma_{sv}(\delta_s(v)+tem_s(v))$

11 v 向 $N_{in}(v)$发送 $msg(d_s(v),s,\phi_s(v,n))$

12 ENDIF

13 WHILE v 接收 $msg(d_s(u),s,\phi_s(u,n))$ DO

14 IF $d_s(u)+1=d_s(v)$

15 THEN

16 $\phi_s(v,n)\leftarrow\phi_s(v,n)+\phi_s(u,n)$

17 $\delta_s(v)\leftarrow\delta_s(v)+\sigma_{sv}(\phi_s(u,n))$

18 ENDIF

19 ENDWHILE

20 ENDFOR

21 FOR $\forall s\in V\setminus\{v\}$ DO

22 FOR $k\leftarrow 0$到 $n-1$ DO

23 IF $n-d_s(v)\le k-1$ THEN

24 $bc_{SM}(v)\mathrel{+}=\delta_s(v)+P_k g_{SM}(d_s(v),k)$

25 ENDIF

26 ENDFOR

27 ENDFOR

根据下面给出的定理，可以确定算法 4 在获得正确结果的同时，仅需 $O(n)$ 的轮数复杂度即可完成计算。

定理3. 在算法 4 的最终输出中，变量 $bc_{SM}(v)$ 等于图 G 中顶点 v 的基于半值的介数中心度。

证明. 算法 4 的基本过程类似算法 1，两者的主要区别在于为了减少原先辅助函数计算所需的通信轮数，算法 4 要求每个顶点在特定时间进行额外的本地计算，参考算法 4 的第 $5-9$ 行。这一子程序使得顶点 u 能够正确统计公式(9)中的 $\sum_{\substack{k\in[0,n-1]\\ n-d_s(u)\le k}}P_k f_{SM}(d_s(u),k)$ 项，并将

$$1/\sigma_{sv}\Big(\sum_{\substack{k\in[0,n-1]\\ n-d_s(u)\le k}}P_k f_{SM}(d_s(u),k)+\delta_s(u,n)\Big)$$

这项发给其对应源点 s 的前驱 v，具体过程与算法 1 中顶点 u 发送公式(5)中的 $f_{SP}(d_s(u))+\delta_s(u)$项一致。因此根据定理 1，算法 1 可以正确计算累积 $\delta_s(v)$，算法 4 也能够正确计算累积$\delta_s(v,n)$，进而得到正确的$bc_{SM}(v)$。

证毕.

定理4. 算法 4 的轮数复杂度和消息复杂度分别为 $O(n)$和 $O(nm)$ 。

证明. 算法 4 根据定理 2，算法 1 的轮数复杂度和消息复杂度分别为 $O(n)$ 和 $O(nm)$。而算法 4 中对算法 1 的修改仅限于本地计算，并未影响通信，因算法 4 的轮数复杂度和消息复杂度与算法 1 一致。

证毕.

5 下界证明

本节将借助两方通信复杂度，通过构造图中集合不相交问题的通信复杂度下界，推导出基于 $Shapley$值的介数中心度问题以及基于半值的介数中心度问题在CONGEST模型下的通信复杂度下界。

定义5[19]. （两方通信复杂度）给定两名参与者Alice和Bob，其中Alice拥有任意输入 a，Bob拥有任意输入 b 。假设存在一个函数h，其算法集记为 $\mathcal{A}$ ，对于 $\mathcal{A}$ 中的任意算法 $alg\in\mathcal{A}$，Alice和Bob的输入分别为 a 和 b 。在使用算法 alg 时，Alice和Bob交换的比数 $Bit(alg(a,b))$，则定义计算函数 h所需交换的最小比特数为：

$$Bit(h)=\min_{alg\in\mathcal{A}}Bit(alg(a,b))$$

假设存在一个图 $G(V,E)$ 并将其分割为两个点不相交的子图，分别称为图 $G_A(V_A,E_A)$和图 $G_B(V_B,E_B)$。图 G 的顶点集合 V 是 V_A 和 V_B 顶点集合的并集，而图 G 的边集合 E 则由图 G_A 的边集合 E_A、图 G_B 的边集合 E_B 和两图之间的边集合 E_C 组成。具体来说，集合 E_A 包含所有两个端点均属于 V_A 点集的边，集合 E_B 的定义类似。集合 E_C 则包含所有两个端点分别位于 V_A 和 V_B 的边。Alice和Bob分别持有 $\{G_A,E_C\}$ 和 $\{G_B,E_C\}$。Alice和Bob可以交换各自持有的信息并解决 $func'$，进而解决图性质函数 $func$，基于这一关系，可以建立 $func$ 的轮数复杂度和 $func'$的两方通信复杂度之间的联系。

定理5[3]. 令 $func$为图 G 上的任意函数，且 $func'$为能够解决$func$的另一函数。则有：

$$\frac{Bit(func')}{2c\cdot B}\le Round(func)$$

其中，$Bit(func')$表示函数 $func'$的两方通信复杂度，$func'$ 通常采用集合不相交问题；$Round(func)$表示函数 $func$ 的轮数复杂度；c 为图中 E_C 的边数量；B 为每条边每轮传递的比特数，在CONGEST模型中，B 为 $O(\log n)$。

本节剩下的部分将描述如何构造一个特定图，并通过该图中两方集合不相交问题通信复杂度，归约出基于 $Shapley$ 值的介数中心度问题在 CONGEST 模型下的轮数复杂度下界。因为半值为 $Shapley$ 值

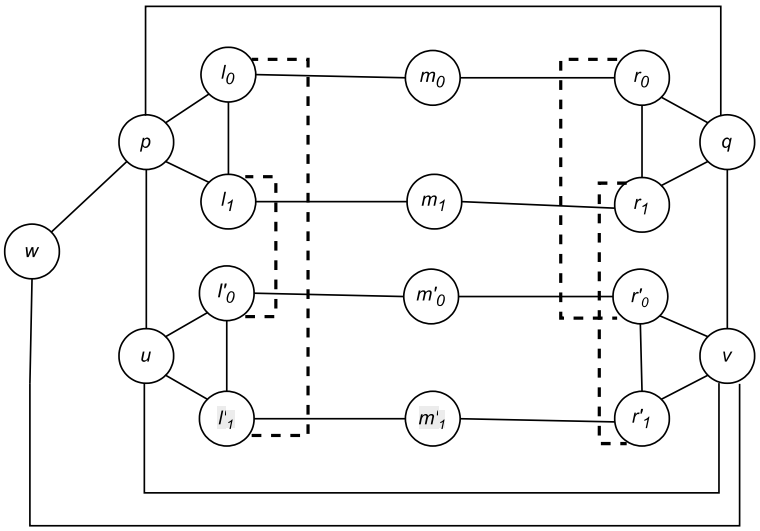

图 2 下界构造图

的一般情况，该下界可以推广到基于半值的介数中心度问题。

构造的特定图中包含六个点集 L、L'、R、R'、M、M'，每个点集包含 N 个点。其中点集L由点 l_0、$l_1 \ldots l_{N-1}$构成，点集L'由点 l'_0、$l'_1 \ldots l'_{N-1}$构成，点集 R 由点r_0、$r_1 \ldots r_{N-1}$构成，点集 R' 由点 r'_0、$r'_1 \ldots r'_{N-1}$构成，点集 M 由点 m_0、$m_1 \ldots m_{N-1}$构成，点集 M'由点m'_0、$m'_1 \ldots m'_{N-1}$构成。此外图中还包含五个特殊点 p、q、u、v、w。

接下来介绍图中固定的边如何连接，在图中通过实线边表示。点 w 与点 p、v之间存在边连接，而点 p、q、v、u依次构成一个环。

点集 L 和点 p 构成一个团，即其中任意两点之间都存在一条边连接。同理，以下的集合也都各自构成团：点集 R 和点 q，点集L'和点 u，点集 R'和点 v。

点集 L 和 M 中两个序号相同的点之间存在边连接，该连接规也适用于以下集合：点集 M 和 R，点集 L'和 M'，点集 M'和 R'。

此外图中存在非固定的边，此类边是否存在取决于两个集合，X 和 Y，在图 2 中通过虚线边表示。集合 X 和 Y 均是集合 $\{0,1 \ldots . N^2-1\}$ 的子集。如果元素i不属于集合 X，那么点 l_{imodN} 与点 $l'_{\lfloor i/N \rfloor}$ 存在一条边连接。同理如果元素 i 属于不集合 Y，那么点 r_{imodN} 与点 $r'_{\lfloor i/N \rfloor}$存在一条边连接。

为了方便理解，图 2 展示 $N=2$ 且集合 X 为 $\{0,3\}$，集合 Y 为 $\{1,2\}$ 的构造图例。

为了利用定理5，我们将该构造图分割为G_A和 G_B：图 G_A 的点集 V_A 包含点集合 L、L' 和点 p、u、w；而图G_B的点集 V_B 包含点集合 R、R'、M、M'和点q、v。因此在该构造图中边集合 E_C 包含点集 L 和M之间的所有边，点集 L' 和 M'之间的所有边以及边 $<w,q>$、$<u,v>$、$<w,v>$，边集合大小为 $O(N)$。

证毕.

引理1. 构造图中点 w 可以根据集合 X 和 Y 是否相交，区分其基于$Shapley$值的介数中心度$bc_{SP}(w)$与一常数值的大小关系：

$$\begin{cases} bc_{SP}(w) = \frac{1}{6} + \frac{2}{9}N + c_g & X \cap Y \neq \emptyset \\ bc_{SP}(w) > \frac{1}{6} + \frac{2}{9}N + c_g & Otherwise \end{cases}$$

其中c_g为一个特定常数。

证明. 回顾基于Shapley值的介数中心度的变换公式：

$$bc_{SP}(w) = \sum_{s,t \in V \setminus \{w\}} \left(\frac{\sigma_{st}(w)}{\sigma_{st}} f_{SP}(d_s(t)) + g_{SP}(d_s(v)) \right)$$

其中 $f_{SP}(d) = \frac{1}{d}, g_{SP}(d) = \frac{2-d}{2d}$。下面分别讨论不同情况下 $bc_{SP}(w)$各项的取值。

首先是讨论 $f_{SP}(d_s(t)) \frac{\sigma_{st}(w)}{\sigma_{st}}$，简称 f 项。观察经过点 w 的最短路径，只有点 p 到点 v、点集 L 到点 v、点集 R'到点 p、点集 L 到点集 R'之间的最短路径可能经过 w；而其他点对之间的最短路径不会经过点 w。

其中前三类最短路径不会受到集合X和Y的影响，必然经过点 w。点 p 到点 v 之间的最短路径有三条，其中一条经过 w，长度为2。点集 L 到点 v、点集 R'到点 p 中，每个点对之间的最短路径有三条，其中一条经过 w，长度为3。因此，这三类最短路径对应的f项求和为

$$\frac{1}{3} * \frac{1}{2} + 2 * N * \frac{1}{3} * \frac{1}{3} = \frac{1}{6} + \frac{2}{9}N$$

而点集 L 到点集 R'之间的最短路径则会受到集合 X 和 Y 影响。对$i \in [0, N^2-1]$，假设元素 i 不同

时属于集合 X 和 Y。根据构筑图的定义，点 l_{imodN} 与点 $l'_{[i/N]}$ 之间，点 r_{imodN}与点 $r'_{[i/N]}$ 之间至少存在一条边，那么 l_{imodN} 与 $r'_{[i/N]}$ 的距离为3，并且最短路径不经过点 w。如图 2，元素0属于 X，但不属于 Y，点 r_0 与点 r'_0 之间存在边，l_0 可以通过 $l_0 - m_0 - r_0 - r'_0$ 到达 r'_0，路径长度为3。同理，元素1属于 Y 但不属于 X，点 l_1 与点 l'_0 之间存在边，l_0 可以通过 $l_0 - l'_0 - m'_0 - r'_0$到达 r'_0，路径长度为3。而 l_0经过点 w 的路径长度至少为4，此时点集 L 到点集 R'之间的最短路径不经过点 w。

反之，假设元素i同时属于集合 X 和 Y，点 l_{imodN} 与点 $l'_{[i/N]}$ 之间，点 r_{imodN} 与点 $r'_{[i/N]}$之间均不存在边，l_{imodN}与$r'_{[i/N]}$的距离为4。这是因为边不存在使得之前描述的途径集合 M 或者集合 M' 的路径消失，图中不存在连通两点且长度为3的路径。此时连通点 l_{imodN}与点 $l'_{[i/N]}$，经过点 w 且长度为4的路径，例如 $l_0 - p - w - v - r'_0$，成为最短路径。

综上，如果集合 X 和 Y 不相交，即没有任何元素同时属于两个集合，那么点集 L 到点集 R'之间的所有最短路径都不会经过点 w，不会对 f 项有任何贡献，$bc_{SP}(w)$的f项为 $\frac{1}{6}+\frac{2}{9}N$。如果集合 X 和 Y 存在元素相交，那么点集 L 到点集 R'之间的部分最短路径经过点 w，$bc_{SP}(w)$的 f 项根据公式必然大于 $\frac{1}{6}+\frac{2}{9}N$。

最后，对于变换公式中的辅助函数 g，对于任意的源点 s，无论集合 X 和 Y 包含哪些元素，$d_s(w)$ 都不会发生任何变化，因此 $bc_{SP}(w)$ 的 g 项恒定。为了方便，我们将辅助函数 g 的结果设置为一个常数 c_g。综合对公式中各项的分析可以得到 $bc_{SP}(w)$ 在不同情况下的取值。

证毕.

定理6. 在CONGEST模型下，求解基于$Shapley$值的介数中心度问题的轮数复杂度下界为 $\Omega(\frac{n}{\log n}+D)$。

证明. 首先由于图中点集合的大小均为 N，构造图的点数量 $n = O(N)$。Alice和Bob各自拥有一个集合X和Y，两个集合均为 $\{0,1,\dots,N^2-1\}$ 的子集，其中每个元素是否存在可以通过二进制编码 $\{0,1\}$ 表示。因此Alice与Bob想要判断各自拥有的集合是否不相交，他们至少需要交换 N^2 比特的信息[20]。

根据引理 1，假设有一分布式算法 alg'试图获得构造图中 $bc_{SP}(w)$与$\frac{1}{6}+\frac{2}{9}N+c_g$ 的大小关系，那么Alice和Bob可以通过集合X和Y的集合不相交问题模拟算法 alg'的下界。设集合不相交问题为 $func'$，基于$Shapley$值的介数中心度求解问题为 $func$。已知$Bit(func') = \Omega(N^2)$，根据定理5可以得到：

$$\frac{Bit(func')}{2c \cdot B} = \Omega\left(\frac{N^2}{2N \cdot \log n} + D\right) = \Omega\left(\frac{n}{\log n} + D\right) \leq \text{Round}(func)$$

其中c为图中E_C的边数量，$c = O(N)$，同时因为在构造图中$n = 6N + 5$，$c = O(n)$。B为每条边的拥塞大小，$B = O(logn)$。在网络中信息从一个节点移动到另一个节点至多需要 $O(D)$ 的时间，因此下界包含额外的$O(D)$项。而仅判断 $bc_{SP}(w)$与某一常数大小关系的算法 alg'的复杂度下界，也是直接计算$bc_{SP}(w)$的算法的复杂度下界。

证毕.

由于 $Shapley$ 值是半值的一种特殊情形，该下界也适用于基于半值的介数中心度问题。结合MR-SMBC的轮数复杂度，说明MR-SMBC已经近似最优。

6 实验

本节通过实验研究分析所提出的MR-SMBC算法的各项属性。具体而言，实验分为以下部分：首先对MR-SMBC算法的可扩展性进行考察，通过在不同规模的问题实例上运行该算法，评估其在处理大规模数据时的效率和资源消耗情况；随后探讨在不同源点批次配置下MR-SMBC算法的性能表现，旨在为实际应用提供最优的配置建议；接着对算法的通信与计算开销分析，量化通信流量和计算时间，探讨两者在整体运行时间中的占比及其对性能的影响；将MR-SMBC算法与SPBC算法以及当前最先进的介数中心度算法MRBC[4]的运行时间进行对比分析。对照实验通过对SPBC算法和MR-SMBC算法在性能上的差异进行评估，间接比较MR-SMBC算法与SMBC算法，

表 2 实验数据集

数据集	$\|V\|$	$\|E\|$	无向或者有向
PA(roadNet-PA)	1.08M	1.54M	无向
CA(roadNet-CA)	1.96M	2.76M	无向
WK(wiki-Talk)	2.39M	5.02M	有向
LJ(soc-LiveJournal)	4.84M	68M	有向
TW(twitter-2010)	41.65M	1.46B	有向
UK(uk-2007-05)	105.90M	3.74B	有向

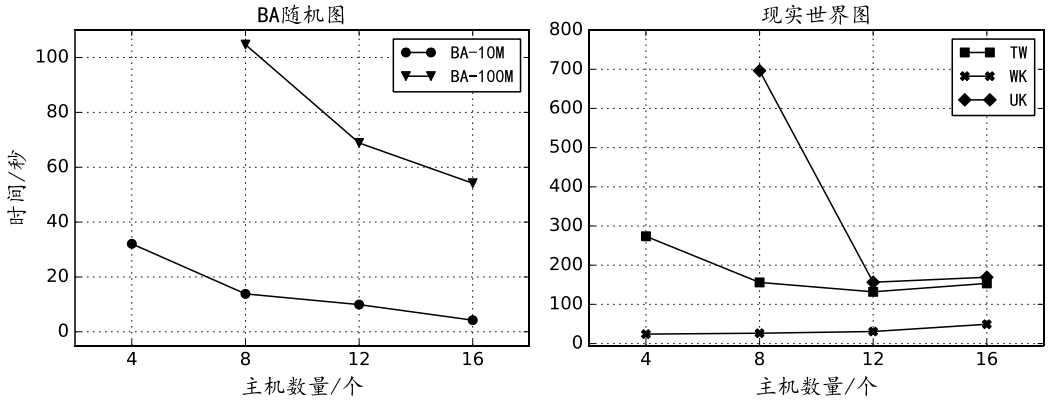

图 3 可拓展性测试

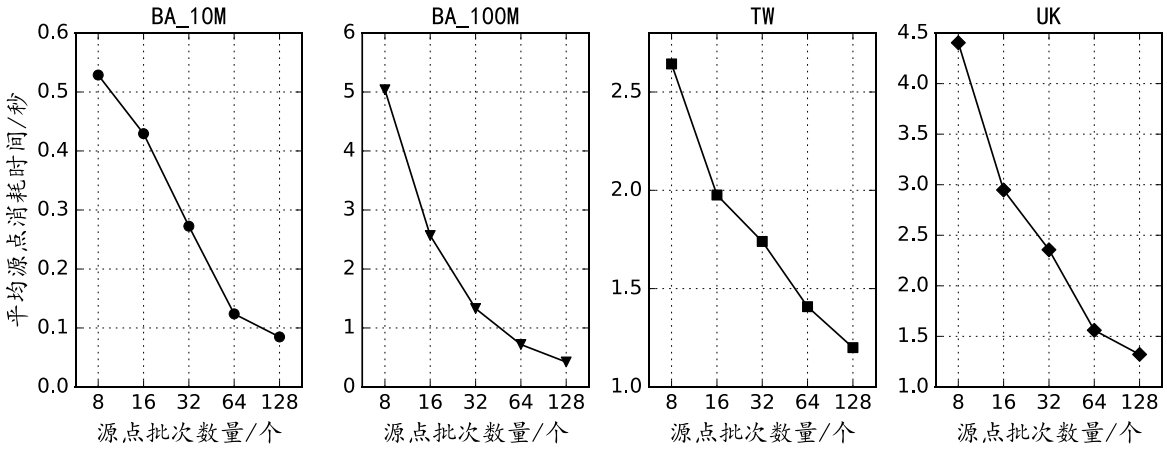

图 4 源点批次测试

从而说明MR-SMBC算法在实际运行时间上的优势。

本研究的实验部分在一组高性能计算集群上进行，该集群由16个服务器组成，每个服务器均配备两颗24核、3.0GHz的IntelXeonGold6248R处理器和256GB的DDR4内存。所有算法均在Gemini[21]上实现。Gemini是一款高效的分布式图处理系统，通过稀疏-稠密信号槽抽象、分块划分、双重表示、NUMA优化和细粒度工作窃取等技术，实现低通信开销、高内存效率和负载均衡。它具备高性能和可扩展性，适用于大规模图数据处理。我们从SNAP[1]和Webgraph[22]中选择了6个真实世界的图，包括社交网络图、通信网络图以及路径网络图。表 2 显示了这些网络的统计信息。另一方面，我们构造不同规模的Barabási-Albert随机图，分别为BA_10M和BA_100M，以测试算法性能，随机图名称中的数字表示图的顶点数量。

6.1 算法可拓展性测试

为测试算法的可扩展性，本小节测试算法在不同数量的主机上的性能表现差异。实验结果如图 3 所示，具体来说，我们在三个现实世界图和两个不同规模的BA随机图，BA_10M和BA_100M，上进行测试。在每个图中随机选择128个顶点作为源点，并以这些随机抽取的顶点为基础，采用MR-SMBC算法计算所有的基于半值的介数中心度。实验分别使用4、8、12和16个主机参与计算。图 3 的两个子图的横轴表示主机数量，纵轴表示MR-SMBC算法所消耗的总时间。需要额外说明的是，对于随机图BA100M和现实世界图UK，因其规模相对较大，当主机数量较少时，计算博弈论介数中心度等复杂度较高的计算任务时，可能会出现内存不足的情况。

本小节结果表明，在处理以上图的大多数情况下，增加主机数量都能显著提高算法的运算效率，尤其是在大规模图中更为明显。例如，在BA_10M图中，当主机数量从4个增加到16个时，总计算时间显著下降，表明算法在并行计算中的优势得到了充分发挥。然而，随着主机数量增加到一定阈值，例如从12个增加到16个时，效率提升速度显著减缓，甚至在某些图中，算法完成的总时间反而增加。

产生这一现象的主要原因在于：当主机数量较少时，通信成本低，且可以充分利用多机优化以提升读写效率。当主机数量过多时，各主机之间的复

[1] https://snap.stanford.edu/

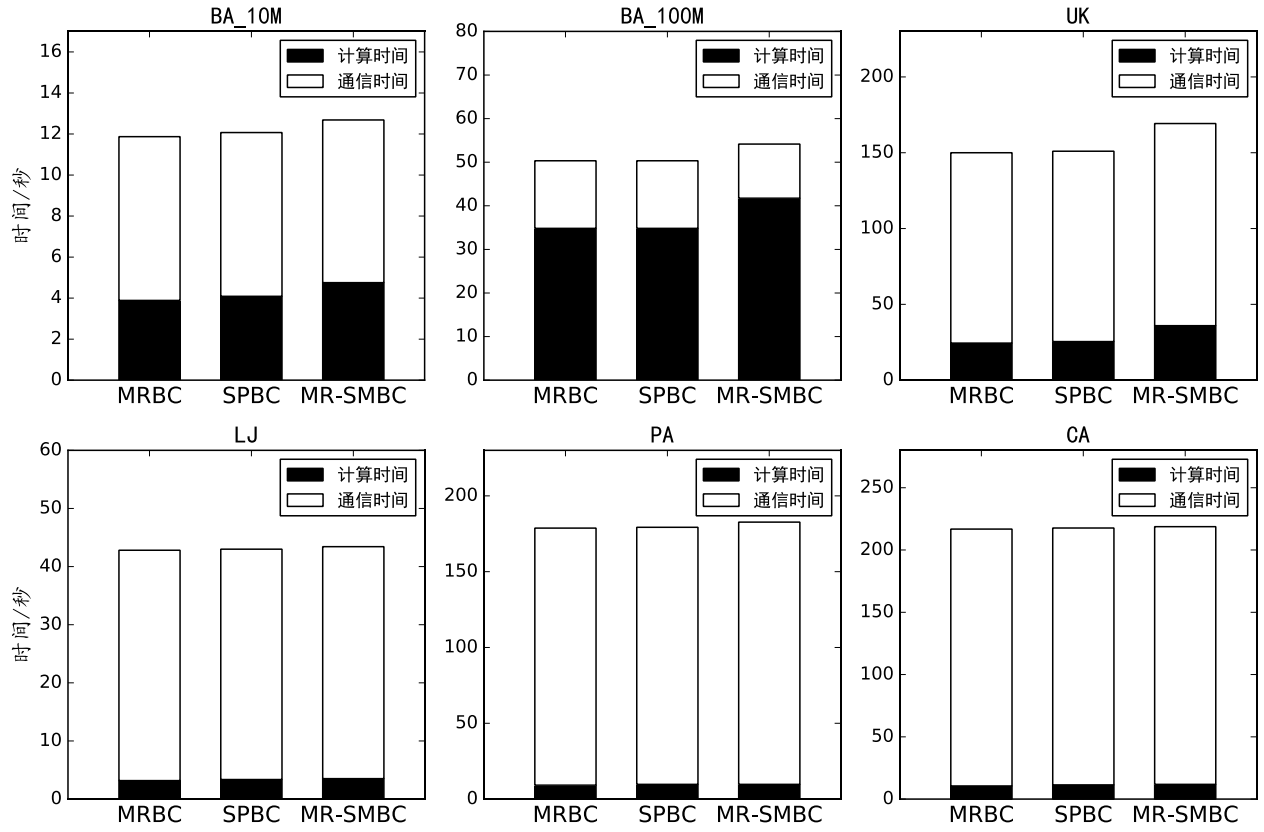

图 5 MRBC 算法，SPBC 算法和 MR-SMBC 算法的通信-计算时间占比

杂通信所需的时间将影响算法整体效率，导致收益递减。具体而言，通信方面，主机数量的增加使得每轮主机之间的通信次数上升，通信时间显著增加；计算方面，尽管主机数量的增加在理论上能够提高计算能力，但在实验中主机数量的线性增加并不意味着计算时间的线性降低。关于计算和通信的时间消耗在MR-SMBC算法中所占的比例的更具体分析将在后续部分提供，以便于进一步理解不同因素对算法性能的影响。

综上所述，实验结果表明，在合适的主机数量范围内，MR-SMBC算法的可扩展性得到了验证。然而，在特定的阈值之后，主机数量的增加可能并不会导致性能的进一步提升，反而可能因为通信开销的增加而导致效率下降

6.2 源点批次测试

本小节讨论MR-SMBC算法选择的源点批次和算法效率的关系。由于硬件限制，MR-SMBC算法在实际应用中无法一次性并行处理所有源点。因此在现实应用场景中，针对目标源点集，我们通常将其随机划分为多个小集合，按批次处理。尽管在算法设计中通常会优先选择一次性处理更多的源点以提高并行性，但在CONGEST模型或其他信道带宽有限的分布式环境中，过多的源点同时参与计算，可能导致信道拥塞，而部分信道则未得到有效利用，造成负载不均衡，这种情况下更大的批次反而会降低整体算法的效率。

为了验证这一现象是否存在，本节测试MR-SMBC算法在处理不同大小的源点批次时的效率变化。具体来说，我们选择了两个BA图和两个现实世界图，统计采用16机，随机选择128个点作为源点，并将这些点分成8、16、32、64和128等不同规模的批次。每次MR-SMBC算法仅处理当前批次内的源点，并记录128个源点全部计算完毕所需的时间，从而计算每个源点的平均时间消耗。实验结果如图 4，其中横轴表示不同的源点批次数量选择，纵轴表示平均每个源点的时间消耗。

图 4 的结果展示算法的耗时与源点批次之间的关系：所有实验结果表明，随着批次规模的增加，平均每个源点的时间消耗呈下降趋势，虽然下降的幅度存在差异。该结果验证了在特定硬件条件下，MR-SMBC算法同时处理更多源点的确是一种更高效的选择。此外，随着批次规模的增加，效率提升的程度（即曲线的斜率绝对值）也逐渐减小，这暗示在批次规模达到一定程度后，继续增加批次规模所带来的效率提升可能会减缓。

6.3 算法效率测试以及通信-计算占比分析

本小节通过实验验证本文算法的性能优势。实验结果表明：1)在传统介数中心度计算方面，本文提出的算法性能与MRBC算法相当；2)在基于半值的

介数中心度计算方面，MR-SMBC算法相较于SMBC算法能够实现线性级别的加速。

SMBC算法需要执行 $O(n)$次SPBC算法，实验中使用的图的顶点数 n 至少为百万级别。而SPBC算法在当前配置下每次运行时的时间消耗需要常数秒。因此，直接计算SMBC算法的资源消耗过高且没有实际意义。事实上，SMBC算法可以看作是SPBC算法的重复调用。我们不直接比较MR-SMBC算法和SMBC算法，而是将SPBC算法与MR-SMBC算法的运行时间进行对比，以展示MR-SMBC算法相对于SPBC算法的优势。因此为达成第二个实验验证目标，我们选择证明相比于SPBC算法的总执行时间，MR-SMBC算法仅需要相较于总时间极少量的额外时间即可完成基于半值的介数中心度计算，明显优于执行 $O(n)$ 次SPBC算法的SMBC算法。

本小节选择两个随机图、两个社交网络图以及两个路径网络图进行测试。每个图中随机选择128个顶点作为源点，使用16个主机分别测试MRBC，SPBC和MR-SMBC算法的性能。实验通过算法的执行时间来衡量性能。每组实验重复三次，最终结果如图 5 所示。为了进一步分析算法表现，采用不同颜色的柱状图表示算法不同部分所消耗的时间。图中的白色部分表示算法的通信时间，黑色部分表示算法的计算时间，两者之和为算法的总时间。

需要注意的是，通过记录Gemini中相应通信接口函数的处理时间来测量通信时间，而计算时间则是从总运行时间中减去通信时间得到。因此，极少量的计算或内存访问时间可能会被误计入通信时间。为了简化分析，我们没有对算法的轮数进行比较，因为在相同实验条件下，两个算法的轮数是相同的。

观察MRBC算法与SPBC算法的结果。在总执行时间方面，MRBC算法与SPBC算法大致持平。两者之间微小的差异主要来源于计算时间，这是由于SPBC算法在计算基于 $Shapley$ 值的介数中心度时需要额外计算辅助函数，而计算传统介数中心度时不需要。另一方面，两个算法的通信时间基本相同，这与理论分析的结果一致。

观察SPBC算法和MR-SMBC算法的结果，MR-SMBC算法和SPBC算法的时间差异通常不到5%，且大部分时间差异也基本来自于计算时间。例如，在BA_100M图上，SPBC算法的计算时间和通信时间分别为34.82秒和15.5秒，而MR-SMBC算法的计算时间和通信时间分别为41.74秒和13.42秒。对于SPBC算法，MR-SMBC算法的额外开销主要来自于对辅助函数f和g的计算，通信开销几乎没有变化。

进一步分析不同类型图的MR-SMBC算法的计算时间和通信时间占比，可以发现不同图在这两者的占比上存在显著差异。例如，在人工生成的随机图BA_100M中，计算时间占比约为70%，而在路径图PA中，计算时间占比仅为5%。这主要是因为两种图的结构特点不同：虽然BA_100M的顶点数量较多，但由于其优先连接高阶顶点的生成策略，其直径较小；而PA图的规模仅为BA_100M的1%，但由于路径图的特性，其直径较大。这导致了两者在计算时间与通信时间占比上的截然不同。

值得注意的是，即使是同规模的社交网络图，如UK，其计算与通信时间占比也与BA_100M有较大区别。这在一定程度上说明，BA随机图的生成策略在模拟现实社交网络时存在局限性。现实社交网络中复杂的互动关系，通常无法通过单一的生成规则准确模拟，这使得BA随机图在某些社交图的模拟中并不贴切。

7 总结

本文提出了一种新的分布式算法MR-SMBC，专门用于计算无权图中的基于半值的介数中心度。该算法基于经典的CONGEST模型，其轮数复杂度为 $O(n)$。此外本文证明了该问题的轮数复杂度下界，表明MR-SMBC算法在CONGEST模型下达到了近似最优的性能。在实验部分，对真实世界图和人工生成图进行了广泛的测试。结果表明，与传统的基础算法相比，MR-SMBC算法能够实现线性级别的加速，显著提升了计算效率。

未来的工作将集中在以下方面：对MR-SMBC算法进行进一步的优化，以降低其在特定类型图（如稠密图或特定拓扑结构的图）中的轮数复杂度。这一优化将通过利用适应性策略和并行计算方法来实现，旨在增强算法在不同图结构中的适用性，提高其在实际应用中的效率和准确性。

其次，我们希望将该算法扩展到加权图和动态网络中，以便更好地适应现实应用场景。此外，针对动态网络的实时更新，我们也会考虑设计增量式的更新机制，以提高算法的灵活性和响应速度。

致 谢 本文的计算工作在华中科技大学高性能计算公共服务平台上完成。

参 考 文 献

[1] Szczepanski P. L., Michalak T. P., Rahwan T. Efficient algorithms for game-theoretic betweenness centrality. Artif. Intell., 2016, 231: 39-63

[2] Brandes U. A faster algorithm for betweenness centrality. The Journal of Mathematical Sociology, 2001, 25(2): 163-177

[3] Hua Q., Fan H., Ai M., Qian L., Li Y., Shi X., Jin H. Nearly optimal distributed algorithm for computing betweenness centrality. 36th IEEE International Conference on Distributed Computing Systems (ICDCS 2016). Nara, Japan, 2016: 271-280

[4] Hoang L., Pontecorvi M., Dathathri R., Gill G., You B., Pingali K., Ramachandran V. A round-efficient distributed betweenness centrality algorithm. 24th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2019). Washington, DC, USA, 2019: 272-286

[5] Lenzen C., Peleg D. Efficient distributed source detection with limited bandwidth. ACM Symposium on Principles of Distributed Computing (PODC '13). Montreal, QC, Canada, 2013: 375-382

[6] Behera R. K., Rath S. K., Misra S., Damasevicius R., Maskeliunas R. Distributed centrality analysis of social network data using mapreduce. Algorithms, 2019, 12(8): 161

[7] Crescenzi P., Fraigniaud P., Paz A. Simple and fast distributed computation of betweenness centrality. 39th IEEE Conference on Computer Communications (INFOCOM 2020). Toronto, ON, Canada, 2020: 337-346

[8] Wang W., Tang C. Y. Distributed computation of vertex and edge betweenness on tree graphs. 52nd IEEE Conference on Decision and Control (CDC 2013). Florence, Italy, 2013: 43-48

[9] Wang W., Tang C. Y. Distributed computation of classic and exponential closeness on tree graphs. American Control Conference (ACC 2014). Portland, OR, USA, 2014: 2090-2095

[10] You K., Tempo R., Qiu L. Distributed algorithms for computation of centrality measures in complex networks. IEEE Trans. Autom. Control, 2017, 62(5): 2080-2094

[11] Solomonik E., Besta M., Vella F., Hoefler T. Scaling betweenness centrality using communication-efficient sparse matrix multiplication. International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2017). Denver, CO, USA, 2017: 1-14

[12] Hua Q., Ai M., Qian L., Yu D., Shi X., Jin H. Low-complexity distributed algorithms on large-scale graphs: A brief review. Journal of Nanjing University of Information Science & Technology, 2017, 9(5): 533-543 (in Chinese)
（华强胜, 艾明, 钱立祥, 等. 大规模图中低复杂度分布式算法浅析. 南京信息工程大学学报(自然科学版), 2017, 9(5): 533-543）

[13] Jamour F. T., Skiadopoulos S., Kalnis P. Parallel algorithm for incremental betweenness centrality on large graphs. IEEE Trans. Parallel Distributed Syst., 2018, 29(3): 659-672

[14] Behera R. K., Naik D., Ramesh D., Rath S. K. MR-IBC: MapReduce-based incremental betweenness centrality in large-scale complex networks. Social Network Analysis and Mining, 2020, 10(1): 25

[15] Yang J.-X., Wang Z.-K., Wang M., et al. Algorithms for local betweenness centrality of fully dynamic multi-dimensional networks. Chinese Journal of Computers, 2015, 38(9): 1852-1864(in Chinese)
（杨建祥, 王朝坤, 王萌, 等. 全动态多维网络局部介数中心度算法. 计算机学报, 2015, 38(9): 1852-1864）

[16] Grofman B. N., Owen G. A game theoretic approach to measuring degree of centrality in social networks. Social Networks, 1982, 4: 213-224

[17] Szczepanski P. L., Michalak T. P., Rahwan T. A new approach to betweenness centrality based on the Shapley value. International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012). Valencia, Spain, 2012: 239-246

[18] Tarkowski M. K., Szczepanski P. L., Michalak T. P., Harrenstein P., Wooldridge M. J. Efficient computation of semivalues for game-theoretic network centrality. Artif. Intell. Res., 2018, 63: 145-189

[19] Kushilevitz E., Nisan N. Communication complexity. Cambridge: Cambridge University Press, 1997

[20] Saglam Mert, Tardos Gabor. On the communication complexity of sparse set disjointness and exists-equal problems. 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013). Berkeley, CA, USA, 2013: 678-687

[21] Zhu X., Chen W., Zheng W., Ma X. Gemini: A computation-centric distributed graph processing system. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016). Savannah, GA, USA, 2016: 301-316

[22] Boldi P., Vigna S. The WebGraph Framework I: Compression techniques. 13th International World Wide Web Conference (WWW 2004). New York, USA, 2004: 595-601

Wang Yefei, Ph.D. candidate. His current research interests include distributed algorithms and graph algorithms.

Qiang-Sheng Hua, Ph.D., Professor, CCF Distinguished member. He is interested in parallel and distributed computing, particularly in the design of efficient algorithms and systems with solid proofs

Gao Wenjie, M.S. candidate. His current research interests include distributed algorithms and graph algorithms.

Jin Hai, Ph.D., Professor, CCF Fellow, IEEE Fellow. His research interests include computer architecture, computing system virtualization, cluster computing and cloud computing, network security, peer-to-peer computing, network storage, and parallel I/O.

Background

In various fields—such as social sciences, biology, transportation, and computer science—networks play a crucial role. Among the many properties of networks, centrality measures have been a key research focus, with betweenness centrality being one of the most widely studied. Betweenness centrality helps identify vertices that play a pivotal role in the dissemination of information. A vertex with high betweenness centrality is often a bottleneck or bridge within the network, serving as a crucial link between different components. For example, in social networks, users with high betweenness centrality are typically key connectors between distinct social groups and act as hubs for information flow.

However, in some scenarios, the standard betweenness centrality measure may not accurately identify critical vertices. For instance, in network defense against unpredictable adversaries, protecting vertices with high betweenness centrality may not always be the best strategy. To address such challenges, the concept of game-theoretic betweenness centrality has been proposed, combining game theory to better identify important vertices in specific situations. Despite its potential, the computation of game-theoretic betweenness centrality requires significant computational resources, which makes it difficult to implement on a single machine.

Therefore, designing efficient algorithms for game-theoretic betweenness centralities in distributed environments is of paramount importance. Although there exist sequential algorithms for game-theoretic betweenness centrality, their direct application in distributed settings results in high round complexities. In this paper, we first introduce a basic implementation of a game-theoretic betweenness centrality algorithm based on Shapley values in the classic CONGEST model. Moreover, we also discuss the trivial high-complexity algorithms for betweenness centrality based on semi-values under the same model. To improve efficiency, we propose the MR-SMBC algorithm, a low-round complexity algorithm for semi-value-based betweenness centrality, which reduces the need for multiple traversals and significantly decreases the number of rounds required. Additionally, the lower bound of the distributed semi-value-based betweenness centrality problem using the two-party communication complexity framework, showing that the proposed algorithm is near-optimal. Experimental results on large-scale real-world networks and synthetic random graphs demonstrate that MR-SMBC achieves linear speedup compared to the trivial algorithms.

This project is funded in part by the National Science and Technology Major Project (Grant No. 2022ZD0115301) and the National Natural Science Foundation of China (Grant No. 62372202). The experiment was conducted on the HPC Platform of Huazhong University of Science and Technology.