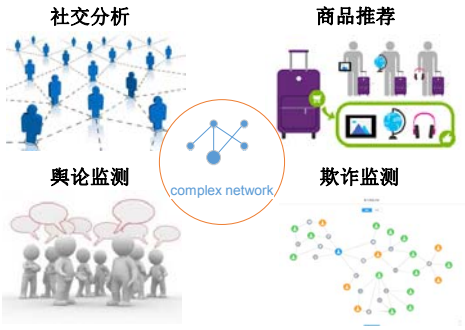
基于Flink Gelly的复杂网络分析算法库

赵伟、邵佳琦、胡家煊、康锴、许利杰、王伟
软件工程技术研究开发中心
wangwei@otcaix.iscas.ac.cn xulijie@otcaix.iscas.ac.cn

复杂网络在各种技术、社会和生物领域中无处不在，复杂网络分析对于理解人类、社会和经济之间的关系起到至关重要的作用。然而，在如今大规模复杂网络场景下，传统算法在时间和空间上都受到了巨大的考验。

本项目基于Apache Flink Gelly分布式图计算平台，针对3类、17项传统复杂网络分析算法进行并行化设计与实现，利用集群所具有的计算优势来处理海量的复杂网络数据，降低算法执行所消耗的时间。在公有云集群环境下，对并行化算法进行实验验证，结果表明并行化算法在准确性和扩展性两个方面均表现优异。本项目为华为杭研院合作项目，项目成果转化至华为大数据引擎部门，并发布技术白皮书。



算法库

本项目对17个典型的复杂网络分析算法进行了并行化设计与实现，算法主要分为三类：顶点分析算法、网络结构分析算法和社区分析算法。

顶点分析算法

类别	算法
中心度	Clossness Centrality算法
	Betweeness Centrality算法
邻居结构	K-Neighbor算法
链路结构	TrustRank算法
	LeaderRank算法
	SimRank算法

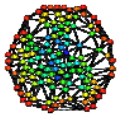
网络结构分析算法

类别	算法
图遍历	BFS算法
图划分	SCC算法
	Graph Coloring算法

社区分析算法

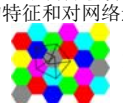
类别	算法
重叠社区	Semi-Clustering算法
	BMLPA算法
	MLPA算法
	CPM算法
	Cliques算法
非重叠社区算法	SCAN算法
	CNM算法
	KCore算法

顶点分析算法适用于分析顶点特征和发现异常顶点的场景，包括用户风险分析等。



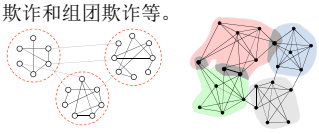
Centrality算法分析顶点在网络中的中心程度。

网络结构分析算法适用于分析网络结构特征和对网络进行划分的场景。



Graph Coloring算法将相邻顶点划分到不同集合。

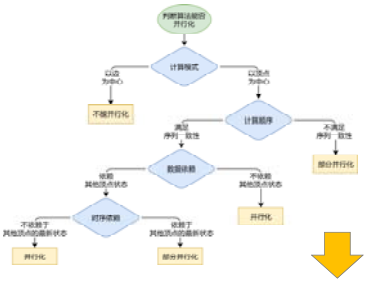
社区分析算法适用于分析社区结构和发现异常顶点集合的场景，包括平台欺诈和组团欺诈等。



社区结构的合理发现，有助于简化复杂网络，凸显出网络中潜在的结构和关系

成果转化至华为大数据引擎部门，算法库将部署到华为云平台

并行化方法

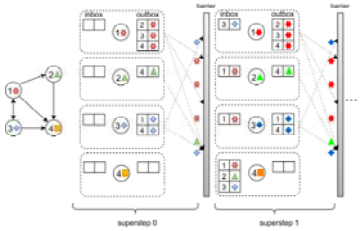


1.特征分析

从计算模式、计算顺序、数据依赖和时序依赖四个方面，判断算法能否并行化，其中，部分并行化指的是算法不能实现全部顶点的并行计算，但可以实现部分顶点的并行计算。

2.并行化设计

基于BSP同步模型和TLAV编程模型，将每个顶点抽象为一个单独的计算单元，通过发送消息来实现与其他顶点的通信，并根据收到的消息来进行顶点状态更新。

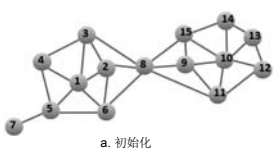


3.并行化实现

基于Apache Flink Gelly分布式图计算平台实现，根据算法的计算性质、通信范围和计算逻辑三个特征，在Gelly提供的VC、SG和GAS三种编程模型中选择适合的模型进行算法实现。

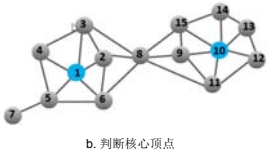
算法设计

以SCAN算法为例说明算法的并行化设计与实现。**(1) 特征分析**：算法以顶点为计算中心，并且满足计算顺序、数据依赖和时序依赖特征，可以并行化；**(2) 并行化设计**：顶点保存顶点标签、社区状态和ε邻居集合，算法执行分为两步，第一步计算所有顶点的社区状态，第二步用于发现枢纽和离群点；**(3) 并行化实现**：采用VC编程模型，具体的执行过程如下：



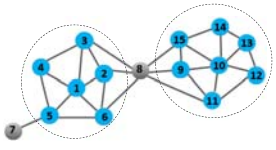
a. 初始化

顶点将社区状态初始化为未分类状态，计算自己的邻居结构并发送给所有邻居顶点。



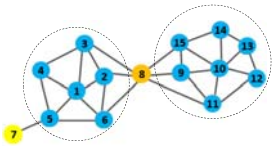
b. 判断核心顶点

V_1和V_{10}判断自己是核心顶点，更新社区状态为1和10，并把社区状态发送给所有ε邻居。



c. 扩展社区

除V_7和V_8外的其他顶点收到消息，更新社区状态，但它们不是核心顶点，不能向ε邻居发送消息。

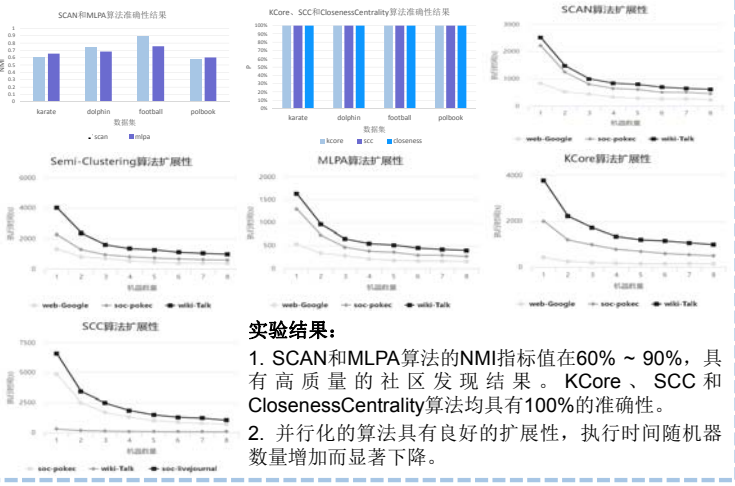


d.发现outlier和hub

V_7收到来自V_5的消息并更新社区状态为outlier，V8收到多个消息并更新社区状态为hub。

实验结果

在公有云集群上进行实验，以并行化的SCAN、Semi-Clustering、MLPA、KCore、SCC和ClosenessCentrality六个算法为例，展示实验结果如下：



实验结果：

1. SCAN和MLPA算法的NMI指标值在60% ~ 90%，具有高质量的社区发现结果。KCore、SCC和ClosenessCentrality算法均具有100%的准确性。
2. 并行化的算法具有良好的扩展性，执行时间随机器数量增加而显著下降。