THU AI TR

人工智能之图计算

Research Report of Graph Computing

2019年 第3期

清华大学人工智能研究院
北京智源人工智能研究院
清华-工程院知识智能联合研究中心

2019年2月

目录

图表目录

摘要

图计算是基于图数据的分析技术与关系技术应运而生的，图计算系统是针对处理图结构数据的系统，图计算也是人工智能中的一个使能技术。基于此背景，本研究报告对图计算这一课题进行了简单梳理，包括以下内容：

图计算的概念与图计算特征。对图计算的概念进行阐述，对代表性分布式图计算系统进行介绍，并列出图计算的特征。

图计算技术。从图计算面临的挑战出发，介绍图算法，图计算模型主要解决的问题，并图计算框架进行介绍。同时对技术资源和图计算的高引论文进行相关介绍。

图计算领域专家介绍。依据 AMiner 数据平台信息，对图计算领域研究学者进行梳理，重点介绍研究学者的研究方向与代表性文章，旨在为学术界、产业界提供图计算技术及学者的分析依据。对顶尖学者的全球分布、迁徙概况、学者机构分布、h-index 分析进行介绍。

图计算产业应用。从医疗行业、金融行业和互联网行业三个方面介绍领域图计算的技术构建应用与研究现状。

图计算趋势研究。对图计算的发展趋势特点进行分析。并基于 AMiner 数据平台，对近期图计算领域研究热点进行可视化分析，与其他学科进行交叉分析研究，对未来图计算研究方向进行预测。

1 概述篇

如今，数据已经渗透到每一个行业和业务职能领域，尤其近年来，全球大数据进入加速发展时期，数据量呈现爆发式增长，大数据吸引了越来越多的关注，大数据时代已然来临。

图计算简单来讲就是研究在这些大量数据中，如何高效计算、存储并管理图数据等问题的领域。因此，传统的关系型数据暴露出了建模缺陷、水平伸缩等问题，于是具有更强大表达能力的图数据受到业界极大的重视。如果把关系数据模型比作火车的话，那么现在的图数据建模可比作高铁。

1.1 图计算的定义

图（Graph）是一种重要的数据结构，它由节点 V（或称为顶点，即个体），与边 E（即个体之间的联系）构成，我们一般将图表示为 G（V，E）。图数据的典型例子有网页链接关系、社交网络、商品推荐等。对应互联网来说，可以把 web 网页看作顶点，页面之间的超链接关系作为边；对应社交网络来说，可以把用户看作顶点，用户之间建立的关系看作边。比如微信的社交网络，是由节点（个人、公众号）和边（关注、点赞）构成的图；淘宝的交易网络，是由节点（个人、商品）和边（购买、收藏）构成的图。

图 1 图数据典型例子

如此一来，抽象出来的图数据便可作为研究和商用的基础，由此探究出“世界上任意两个人之间的人脉距离”、“关键意见领袖”等。将这些应用到商业领域，其底层的运算往往是图相关的算法。比如图的最短路径算法可以做好友推荐，计算关系紧密程度；对图做 PageRank 可以用于传播影响力分析，找出问题的中心，做搜索引擎的网页排名；最小连通图可以识别洗钱或虚假交易等等。

在大数据时代，数据规模通常十分庞大。以社交网络为例，Facebook 在 2014 年 7 月的用户已经达到 22 亿户，而用户之间的关系数量则更多，以数据的方式进行存储通常会占用几百 GB 甚至 TB 级的存储量。在电信工业中，广州市仅一个月内由电话呼叫方和被呼叫方

组成的图就超过 4.5 千万个节点、1.5 亿条边。而著名的 ClueWeb 数据包含全量的 Web 站点和网页，2012 年公布的数据集已经达到 1 亿个节点、425 亿条边，仅是存储边的列表的磁盘文件就超过 400GB。因此，图计算不仅是计算密集型，同时也是存储密集型问题，如何在可以接受的时间内对大图进行计算，是需要解决的难题。

表 1 图计算的应用

应用	元素（图计算顶点）	连接（图计算的边）
社交网络	社交成员	友情
计算机网络	计算机	网络链接
网页内容	网页	超链接
交通	城市	公路
电路	设备	电线
商务	顾客、商品	交易方式
工厂	机械设备	生产线
供应链	供应商	距离
电信	移动电话	电话沟通

1.2 图计算的产生与发展

尽管图形分析一直以来都是计算机相关研究的一个重要领域，但图计算的研究在近年来才受到了重点关注。自 2001 年以来，分布式方法就一直是比较热议的处理大图数据的方法。图计算研究真正开始的标志是 2004 年 Google 开发出面向大数据并行处理的计算模型 MapReduce，这一模型的推出给大数据并行处理带来了巨大的革命性影响。随后，2006 年 Apache Hadoop 团队引入了 Hadoop 分布式文件系统（HDFS）以及新的 Hadoop MapReduce 框架。2009 年，加州大学伯克利分校 AMP Lab 开发出 Spark 系统。此后，许多研究团队都专注于设计新颖的 MapReduce 算法，改进特定工作负载的框架，解决大图分析问题。但是，这些系统与 MapReduce 一样依赖于磁盘，仍然存在局限性，执行速度慢，处理大型图数据效率较低。

为满足各类基于图数据分析计算的应用需求，谷歌基于 BSP（bulk synchronous processing）框架提出了最早的图计算模型，将图计算转换为细粒度的节点计算。自此，BSP 模型成为之后各类图计算系统的基础。BSP 模型将迭代计算划分为多个超步运算，一次超步运算完成一轮迭代计算，三个任务在超步内并行执行，并在每次超步运算结束后完成任务间的数据同步。BSP 模型为图计算模型面临的迭代图算法的分割和细粒度并行计算提供了解决思路。因此，由 BSP 模型抽象而来的计算—通信—数据同步的三步计算模型成为图计算模型的经典框架。

基于 BSP 模型实现的首个图计算模型即 2010 年谷歌提出的同步节点计算模型，该模型应用于以顶点为中心的图计算系统 Pregel，其专为大规模图数据处理而设计，之后 Pregel 便成为 Google 搜索服务的重要系统。对于普通程序员来说，Pregel 可以更直观、更自然地用于各种图形分析任务，其以顶点为中心的编程模型也非常具有表现力，因为顶点可以与任何其他顶点进行通信。Pregel 的出现引发了大量相关研究，同年，卡耐基·梅隆大学推出了图计算系统 GraphLab，其经过多个版本演化已发展成为当前最有影响力的图计算系统之一。虽然 pregel 和 GraphLab 都是对于复杂机器学习计算的处理框架，用于迭代型（iteration）计算，但是二者的实现方法却采取了不同的路径：Pregel 是基于大块的消息传递机制，GraphLab 是基于内存共享机制。2014 年，Spark 也提供了专门支持图计算的模块——GraphX，可以用于实现复杂的图数据挖掘。2015 年，微软经过图计算系统 Kineogragh、Chronos 和 ImmortalGraph 的探索，正式发布大规模图数据计算引擎 GraphEngine。此外，由于商用集群和云的图处理系统变得格外受欢迎，出现了多个具有不同编程模型和功能的分布式图处理框架，比如 Giraph、PowerLyra、Gemini 等。图计算系统已成为大数据分析的重要平台。

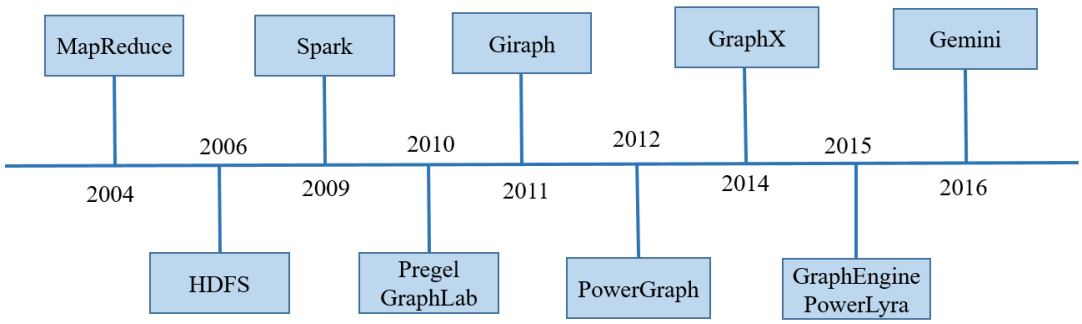


图 2 图计算的发展

1.3 图计算的特征

初提图计算，很多人会以为这是一种专门进行图像处理的技术。事实上，图计算中的“图”是针对“图论”而言的，是一种以“图论”为基础的对现实世界的一种“图”结构的抽象表达，以及在这种数据结构上的计算模式。图数据结构很好地表达了数据之间的关联性，关联性计算是大数据计算的核心——通过获得数据的关联性，可以从噪音很多的海量数据中抽取有用的信息。图计算技术解决了传统的计算模式下关联查询的效率低、成本高的问题，在问题域中对关系进行了完整的刻画，并且具有丰富、高效和敏捷的数据分析能力，其特征有如下三点。

- **基于图抽象的数据模型**

图计算系统将图结构化数据表示为属性图，它将用户定义的属性与每个顶点和边缘相关联。属性可以包括元数据（例如，用户简档和时间戳）和程序状态（例如，顶点的 PageRank 或相关的亲和度）。源自社交网络和网络图等自然现象的属性图通常具有高度偏斜的幂律度分布和比顶点更多的边数。

图计算系统中最基础的数据结构由顶点 V（或节点）、边 E、权重 D 这三因素组成，即 G=(V，E，D)，其中 V 为顶点（vertex），E 为边（edge），D 为权重（data）。顶点表示某一事件中的对象，而边则是对不同对象关系的描述。图计算系统基于顶点和边的方式存储图数据和计算，能够建构任意复杂的网络和模型，完整且形象地映射分析人员想要研究的问题域。

比如说，对于一个消费者的原始购买行为，有两类节点：用户和产品，边就是购买行为，权重是边上的一个数据结构，可以是购买次数和最后购买时间。对于许多我们面临的物理世界的数据问题，都可以利用图结构的来抽象表达：比如社交网络，网页链接关系，用户传播网络，用户网络点击、浏览和购买行为，甚至消费者评论内容，内容分类标签，产品分类标签等。

图计算系统的数据结构很好地表达了数据之间的关联性，关联性计算是大数据计算的核心——通过获得数据的关联性，可以从噪音很多的海量数据中抽取有用的信息。比如，通过为购物者之间的关系建模，就能很快找到口味相似的用户，并为之推荐商品；或者在社交网络中，通过传播关系发现意见领袖与其操作符（例如，连接）可以跨越多个集合的数据流系统相比，图处理系统（例如，顶点程序）中的操作通常相对于具有预先声明的稀疏结构的单个属性图来定义。虽然这有助于进行一系列优化，但它也会使可能跨越多个图和子图的分析任务的表达变得复杂。

- **图数据模型并行抽象**

图的经典算法中，从 PageRank 到潜在因子分析算法都是基于相邻顶点和边的属性迭代地变换顶点属性，这种迭代局部变换的常见模式形成了图并行抽象的基础。在图并行抽象中，用户定义的顶点程序同时为每个顶点实现，并通过消息（例如 Pregel）或共享状态（例如 PowerGraph）与相邻顶点程序交互。每个顶点程序都可以读取和修改其顶点属性，在某些情况下可以读取和修改相邻的顶点属性。

顶点程序并发运行的程度因系统而异。大多数系统采用批量同步执行模型，其中所有顶点程序以一系列“超级步”同时运行。但是也有一些系统支持异步执行模型，通过在资源变得可用时运行顶点程序来减轻落后者的影响。

- **图模型系统优化**

对图数据模型进行抽象和对稀疏图模型结构进行限制，使一系列重要的系统得到了优化。比如 GraphLab 的 GAS 模型更偏向共享内存风格，允许用户的自定义函数访问当前顶点的整个邻域，可抽象成 Gather、Apply 和 Scatter 三个阶段。GAS 模式的设计主要是为了适应点分割的图存储模式，从而避免 Pregel 模型对于邻域很多的顶点、需要处理的消息非常庞大时会发生的假死或崩溃问题。

1.4 图计算的发展困境

近年来，图数据规模呈指数级增长，可能达到数十亿的顶点和数万亿的边，且还在不断增长，单机模式下的图计算已经不适合目前的数据增长，传统的分布式大数据处理平台比如 MapReduce、Spark 也出现网络和磁盘读写开销大、运算速度慢、处理效率极低的问题[1]。在实际应用中，图数据规模巨大、耦合性强、动态变化等特点和图计算频繁迭代操作的特点使得传统计算架构如 MapReduce 等已无法满足图数据分析的新需求。

对于图计算而言，性能成本、容错机制以及可拓展性都是非常重要的。如果性能可以显著提高，结点显著减少，那么就能极大地缩短运行时间。在此基础上，如果使用更大开销的容错技术，例如检查点的方式，那么故障产生的概率将更低。但是，传统的大数据分析平台往往只在性能与可拓展性中选择了一方。比如 MPI、OpenMP 等注重性能的平台只支持可读写的数据，容错困难，可扩展性差，自动负载不平衡；专注于拓展性的大数据分析平台，如 MapReduce、Spark 等支持只读数据集，容错机制和扩展性好，自动负载平衡，但性能较低。以 Spark 为例，其基于 Scala 语言，运行在 JVM 上，内存表示冗余，占用内存大，垃圾收集对性能影响大。在一些迭代的图算法上，开启 128 个线程的 Spark 程序性能有时候还不如优化很好的单线程序，并且需要的内存容量是原始数据集的 20 倍——对于 10TB 级的数据，往往需要数百 TB 的内存，这在绝大部分生产环境中是不可能的。以 Sogou 的网页链接数据为例，网页链接数据量为 137TB，这是很难使用 Spark 进行计算的。

此外，早期的图计算方法主要局限于智能社区或社交网络分析，如果图计算方案的性能和容量限制可以克服，图计算可以应用于更广泛的场景，如资本市场风险管理、生命科学研究、医疗保健交付、监控和应对道路事故、智能基础设施管理和其他领域。因此，为应对高效处理大规模图数据的巨大挑战，可扩展分布式图计算成为了当前热点研究问题。研究高效处理大规模数据的图计算，能推动社交网络分析、语义 web 分析、生物信息网络分析、自然语言处理等新兴应用领域的发展。

2 技术篇

从上文的概述篇中我们看到，图计算领域面临大数据环境带来的巨大挑战。随着图数据量的骤增，图数据计算模型、图计算系统的构建、图计算领域应用的关键技术等领域受到越来越多的关注。本篇将重点介绍图计算技术领域的发展成果，以期更清晰地展现图计算在各个领域的应用。

2.1 图算法

图算法指利用特制的线条算图求得答案的一种简便算法。无向图、有向图和网络能运用很多常用的图算法。对于图数据，遍历算法是其它算法的基础。典型的图算法有 PageRank、最短路径、连通分支、极大独立集、最小代价生成树以及 Bayesian Belief Propagation 等。图的最小生成树在生活中常代表着最低的成本或最小的代价，常用 Prim 算法和 Kruskal 算法。社区发现、最短路径、拓扑排序、关键路径也都有对应的算法。下面简单对几种图算法进行介绍。

- 遍历算法

图的遍历是指从图中的任一顶点出发，对图中的所有顶点访问一次且只访问一次。图的遍历操作是图的一种基本操作，图的许多操作都建立在遍历操作的基础之上。由于图中节点之间的关系是任意的，所以图的遍历比较复杂，主要表现在以下四个方面：（1）在图结构中，没有一个明显的首结点，图中任意一个顶点都可作为第一个被访问的结点，所以要提供首结点。（2）在非连通图中，从一个顶点出发，只能够访问它所在的连通分量上的所有顶点，因此，还需考虑如何选取下一个出发点，以访问图中其余的连通分量。（3）在图结构中，可能有回路存在，那么一个顶点被访问之后，有可能沿回路又回到该顶点，在访问之前，需要判断结点是否已经被访问过。（4）在图结构中，一个顶点可以和其它多个顶点相连，当这样的顶点访问过后，存在如何选取下一个要访问的顶点的问题。

因此，在遍历图时，为保证图中各顶点在遍历的过程中访问且仅一次，需要为每个顶点设计一个访问标记，设置一个数组，用于标示图中每个顶点被访问过，它的初始值全部为 0，表示顶点均未被访问过；某个顶点被访问后，将相应访问标志数组中的值设为 1，以表示该顶点已经被访问过。

- 社区发现（Community Detection）

社区发现算法是用来发现网络中的社区结构，也可以看做是一种聚类算法。社区发现算法可以用来发现社交网络中三角形的个数（圈子），可以分析出哪些圈子更稳固，关系更紧

密，用来衡量社群耦合关系的紧密程度。从一个人的社交圈子里面可以看出，三角形个数越多，说明他的社交关系越稳固、紧密。在图计算的社交应用当中，像 Facebook、Twitter 等社交网站，常用到的的社交分析算法就是社群发现。

- PageRank

PageRank 是 Sergey Brin 与 Larry Page 于 1998 年提出来的，用来解决链接分析中网页排名的问题。PageRank 的计算充分利用了两个如果：数量如果和质量如果。PageRank 源自搜索引擎，它是搜索引擎里面非常重要的图算法，可用来对网页做排序。比如我们在网页里搜索 weibo，会出来非常多有着 weibo 关键字的网页，可能有上千上万个相关网页，而 PageRank 可以根据这些网页的排序算法将其排序，将一些用户最需要的网页进行优先展示。

- 最短路径

最短路径用于计算一个节点到其他所有节点的最短路径。主要特点是以起始点为中心向外层层扩展，直到扩展到终点为止。最短路径在社交网络里面，有一个六度空间的理论，表示你和任何一个陌生人之间所间隔的人不会超过五个。最短路径是图算法中的一种，在图计算应用上很常见。

2.2 图数据计算模型

图计算模型即针对图数据和图计算特点设计实现的计算模型，一般应用于图计算系统中。与传统计算模型相比，图计算模型主要针对解决以下问题：（1）图计算的频繁迭代带来的读写数据等待和通信开销大的问题；（2）图算法对节点和边的邻居信息的计算依赖问题；（3）图数据的复杂结构使得图算法难以实现分布不均匀的分块上并行计算的问题。

为提高图计算系统分析计算图数据的能力，图计算系统需要针对图数据和图计算的特点，设计并实现支持频繁迭代操作、细粒度并行计算和通信开销小的全新计算模型，即图计算模型。按照计算对象，图数据计算模型可以分为节点中心计算模型、边中心计算模型、路径中心计算模型和子图中心计算模型四类。其中节点计算模型提出时间最长，且被多个图计算系统引用，因此将其按计算任务调度方法进一步分为同步计算模型、异步计算模型和混合计算模型三类。

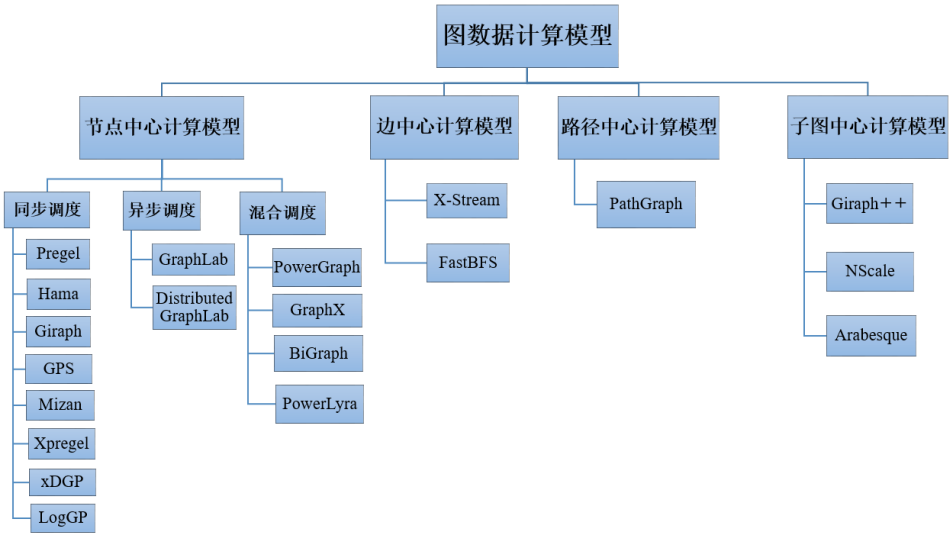


图 3 图数据计算模型分类及应用系统

- 节点中心计算模型

在图计算模型提出之前，图数据分析系统基本都采用 MapReduce 架构，然而 MapReduce 框架无法满足图计算的需求，无法解决图数据的耦合性、稀疏性和图计算的频繁迭代操作等特点带来的数据划分重组频繁、通信开销过大和计算并行性受限等问题。为解决以上问题，Google 在 2010 年首先基于 BSP 模型提出了节点中心计算模型，即将图算法细粒度划分为每个节点上的计算操作。节点中心计算模型将频繁迭代的全局计算转换成多次超步运算，且所有的节点独立地并行执行计算操作，数据间依赖关系仅存在于两个相邻的超步之间。节点中心计算模型成功解决了图计算为传统计算模型带来的难题后，众多采用节点中心计算模型的图计算系统相继出现，且各个图计算系统分别针对不同的计算需求对节点计算模型的数据同步和通信方式进行改进。

✧ 同步节点中心计算模型

同步节点计算模型基于图计算局部性差、节点计算量小、并行性差异大的特点提出，将图算法的每一次迭代转换为图中每一个节点执行一次超步（superstep）运算。一次超步运算包括三个步骤：（1）接收当前节点所有入边上的邻居（in-neighbor）在上一个超步中发出的信息；（2）根据接收到的信息计算当前节点的新值，即执行程序员自定义的 compute 函数；（3）向当前节点所有出边邻居（out-neighbors）发送更新消息。同步节点计算模型所有节点完成一次超步运算后同步更新节点信息，之后进入下一次超步运算。

由于图算法在不同节点上的迭代次数不同，每个节点都执行相同轮次的迭代计算是不合理的，所以同步计算模型将节点分为活跃（active）和不活跃（inactive）两种状态。当节

点接收到入边邻居（in neighbor）发送的信息时，其变为活跃状态；而未接收到更新消息的节点，不需要重新执行计算操作，其变为不活跃状态。当图中所有节点都是不活跃状态时（不考虑初始时刻），或者没有信息在超步间传递时，图计算操作结束。

同步节点中心计算模型解决了 MapReduce 分布式框架无法有效支持迭代型图算法的局限性，为并行图计算系统设计提供了新的思路，使得大规模图算法的实现更加简洁明了。因此，同步节点计算模型自提出后被多个图计算系统采用并改进，如 Giraph、GPS、Mizan、xDGP 和 LogGP。然而，全局同步节点中心计算模型要求每次超步运算需等待全局数据同步，使得算法计算效率受最慢节点计算速度限制，计算资源利用率有待进一步提高。

✧ 异步节点中心计算模型

2010 年，卡内基·梅隆大学的学者基于异步图计算模型提出了异步图计算系统 GraphLab，并在 2012 年发布了改进系统 distributed GraphLab，实现了分布式异步图计算模型。异步计算模型与同步计算的迭代设计相同，仍为 BSP 三步操作模型，但是在接收上一轮超步计算的消息时，不再是由邻居节点推送更新的数据，而是由计算节点采用“拉”的方式选择性地读取邻居节点的消息。一次超步运算包括三步操作：（1）获取当前节点的邻居信息（关联边或邻居节点）；（2）执行用户定义计算操作；（3）更新当前节点所有可写邻居（关联边或邻居节点）的信息。异步节点计算模型不设置同步障碍进行全局数据同步，每个节点异步地读取或更新邻居节点和边的信息，完成以上三步操作后，计算节点独立执行下一次超步运算。异步执行计算操作的节点可能同时对相同节点或边产生读写访问操作，因此为保证数据一致性，异步节点中心计算模型中根据每个节点可异步读取和更新的关联边和邻居节点的范围提出三种一致性方案：全局一致性、边一致性、节点一致性。在节点一致性方案中，节点仅能写自己的数据，或读取关联边的信息，而无法访问邻居节点的信息；在边一致性方案中，节点可以对自己的关联边进行读和写操作，但对邻居节点的信息只能读取而不能修改；在全局一致性方案中，节点可以对自己的关联边和邻居节点进行读和写操作。

异步节点中心模型提高了计算资源利用率，随着图数据规模的增加，其计算优势更加显著。但异步节点中心模型同时引入了维护数据一致性的开销，数据一致性策略实现复杂，且容易产生冲突和错误。因此大部分图计算系统选择同时实现同步和异步节点计算模型。

✧ GAS（gather、apply、scatter）节点中心计算模型

同步和异步计算模型均以节点作为计算中心，将边作为信息传递的路径，因此，这种节点模型的计算能力受图数据中节点和边分布特点限制。Gonzalez 等人提出了 GAS 节点中心计算模型，解决图计算的上述问题。

GAS 计算模型沿用同步节点中心计算模型中超步的概念，并且通过划分大度数节点在单个计算节点内实现并行计算。每一次的超步运算分为三个步骤：（1）信息收集（gather）和汇总（sum），即收集计算节点的邻接节点和边上的信息，执行用户自定义汇总函数将得到的信息汇总到主节点；（2）应用（apply）和更新（update），即由主节点执行用户自定义的计算操作，并将镜像节点更新为计算所得的新值；（3）分发（scatter），即由主节点和镜像节点将更新信息推送给各自相关联的邻居节点和边。GAS 计算模型通过切分节点可以有效降低超步运算带来的通信开销并提高计算并行性。GAS 计算模型将大度数节点划分为多个计算单元，即一个主节点和多个镜像节点，从而将单个大度数节点上的一次超步运算划分为多个计算单元上的并行计算，即一个节点的信息收集和分发由多个计算节点并行执行，且将一次超步运算中通信开销降低为各个计算单元之间的消息交换。

GAS 计算模型通过划分节点实现了节点内并行计算，其计算并行性比异步节点中心模型更好。当分析计算的图数据中节点之间的度存在显著差异时，GAS 计算模型的计算优势也更加显著。因此，GAS 计算模型被多个图计算系统采用及改进，如 BiGraph、PowerLyra。然而，GAS 计算模型的实现比异步节点中的模型和同步节点中心模型更加复杂。

- 边中心计算模型

节点中心模型提高了图计算系统实现图数据计算分析的能力，但是在实际应用中仍面临一些问题：（1）为提高计算节点随机访问邻居的速度，图计算系统一般将完整图数据载入内存，因此当图数据规模过大时，通常在分布式系统中实现节点中心计算模型，即对实现系统设备资源要求过高；（2）当图数据中边数目远远大于节点数目时，每次超步运算中的信息更新和分发操作的时间开销将远大于节点计算时间，通信成为图计算的主要瓶颈，限制了完成主要计算操作的速度。

为解决设备资源受限和边数目远大于节点数目时的图数据分析计算问题，洛桑联邦理工学院在 2013 年提出边中心模型，并将其应用于图计算系统 XStream。边中心计算模型将图算法构建为在图数据的边列表上的流式迭代计算，每一次迭代地完成计算、排序和更新三步操作：（1）读取边列表流，完成用户定义的计算操作，输出更新信息到目的节点列表中；（2）应用将目的节点列表重排序（shuffle）为更新消息流；（3）读取更新消息流和源节点列表，更新源节点值。三步操作在一次迭代计算中顺序执行。

边中心计算模型将图数据以边列表为核心数据结构并维护源节点列表，每次迭代的计算操作更新目的节点列表（Uout），其记录在边列表上的计算操作产生的对每条边的目的节点进行更新的消息序列。边中心计算模型将图算法的迭代计算转换为可在边列表上顺序执行，避免了随机读写数据对内存资源的高要求，从而解决了节点中心计算模型面临的资源受

限和通信开销过大的难题。边中心计算模型的流式顺序计算特点使得在全局图数据上的计算可以分块实现，顺序访问存储在硬盘上的数据，降低了图数据分析计算对内存容量的要求，即可在单机上实现对大规模图数据的分析处理。此外，边中心计算模型将每次迭代计算生成的目的节点更新消息序列进行重排序，获得按源节点合并排序的更新消息流，则更新消息数不大于节点数，大大简化了消息更新同步的开销。因此，边中心计算模型满足了在单机系统上分析边数目远大于节点数的图数据的计算需求

- 路径中心计算模型

边中心计算模型和节点中心计算模型分别将图算法转换为可在节点和边上执行的迭代计算，但同时也将图计算的并行性限制在了节点和边层次上。然而图算法在图结构上是沿节点到边再到节点的顺序计算的，因此，华中科技大学服务计算技术与系统实验室提出更接近理想图计算分析的模型———路径中心计算模型，并将其应用于图计算系统 PathGraph。

路径中心计算模型以路径为计算单元，即从源节点出发到目的节点的边序列。路径中心计算模型将图数据组织为前向边遍历树（forward-edge traversal tree）和后向边遍历树（reverse-edge traversal tree），从而将图计算转换为在树上的迭代计算。路径中心计算模型基于前向遍历树和后向遍历树的每次迭代运算分为两步：（1）消息分发（Scatter），即父节点沿前向边遍历树更新子节点或出边信息；（2）信息收集（Gather），即父节点沿后向边遍历树收集子节点或入边信息。

路径中心计算模型从数据结构决定图算法计算顺序的角度出发，设计前向边遍历树和后向边遍历树，简化了图计算时节点访问入边邻居和出边邻居的查找操作。因此，相比边中心计算模型，当图节点规模在百万级以上和边规模在亿级以上的数据集上测试 BFS、PageRank、联通子图等图算法，路径中心计算模型的计算速度提高了 1～2 倍。但是全局数据同步设计限制了计算并行性，同步数据会带来大量通信开销，导致计算资源利用率低。

- 子图计算模型

路径中心计算模型相比以节点或边作为计算中心的模型更接近图结构上的理想计算状态，然而以上计算模型都面临两个问题：（1）所有节点只有自己的直接邻居信息，图中传递的更新消息每次只能扩散一层，因此从源节点到目的节点的一次更新消息需要多次迭代才能完成，消息更新过慢会带来额外的计算时间开销；（2）节点或边均产生大量的通信开销，超步内执行计算操作的节点或边均产生更新消息，每次超步运算结束后，大量的更新消息带来了全局数据同步的等待或维持数据一致性的开销。

为解决以上问题，IBM 阿尔马登研究中心于 2013 年在图计算系统 Giraph++中提出子图中心计算模型，将完整图结构上的计算转换为多个子图上的迭代超步运算。子图中心计算模

型完成图划分后，在多个子图上并行执行迭代图计算，一次超步运算执行两步操作：（1）子图并行执行用户定义计算操作，并输出计算结果；（2）包含相同节点的子图间更新节点信息。步骤（2）可在所有子图的步骤（1）操作结束后同步执行，或者在保持数据一致性前提下异步执行。

子图中心计算模型通过子图划分方法，将图算法转换为多个子图上的迭代计算，成功减少了计算时的通信开销和迭代操作次数。因此，自子图中心计算模型提出后的短时间内，多个图计算系统采用了子图中心计算模型，并针对子图划分的问题。

表 2 图计算模型对比

图计算模型	任务调度	数据划分	并行性	系统实现	优势	局限
节点中心	同步/异步	节点序列子集	高	分布式/单机	模型实现简单且易于算法迁移；计算并行性高，可采用同步或异步调度，适用于各类算法	计算节点之间通信开销大；完成图计算所需迭代次数多；计算并行性受数据一致性限制
边中心	同步/异步	边序列分块	中	单机	模型实现对设备资源要求低；数据存储、分块和读写访问更加简单；数据访问顺序执行，易于维护数据一致性	计算并行性受边列表分块限制；图算法迁移复杂，且适用范围小
路径中心	同步	子树	中	分布式/单机	数据查找访问简单快捷；基于两步操作，图算法实现简单	构建遍历树的初始化开销大；数据一致性实现复杂；模型实现困难，且数据存储复杂
子图中心	同步/异步	子图	低	分布式/单机	超步运算之间通信开销小；完成图算法的迭代次数少	计算并行性受限；基于子图的数据划分困难；实现用户可自定义的子图划分复杂

2.3 图计算系统

本节将重点介绍图计算系统，为便于读者理解，接下来先简要介绍一下图数据库。

在众多不同的数据模型里，关系数据模型自 20 世纪 80 年代就处于统治地位，而且出现了不少巨头，如 Oracle、MySQL，它们也被称为关系数据库管理系统（RDBMS）。然而，随着关系数据库使用范围的不断扩大，也暴露出一些它始终无法解决问题，其中最主要的是数据建模中的一些缺陷和问题，以及在大数据量和多服务器之上进行水平伸缩的限制。

因此，近年来诞生了 Neo4j、InfiniteGraph 等专注于图结构化存储与查询的图数据库。与传统的关系型数据库相比，图数据库善于处理大量的、复杂的、互联的、多变的网状数据，效率远远高于传统型数据库，性能约有百倍以上的提升，特别适合用于社交网络、实时推荐、银行交易环路、金融征信系统等领域。

图计算是基于图数据的分析技术与关系技术应运而生的，图计算系统就是针对处理图结构数据的系统，并对这样的数据进行针对性优化的高效计算。传统的图论和图算法领域侧重于研究图问题的算法复杂度以及现实规模的图算法问题。在当前的大数据分析领域，需要处理的图数据规模往往高达数十亿以上，并且图数据结构复杂多变，图算法难以在传统计算系统中进行高效处理，因此，需要设计支持大规模、高效图计算的系统以应对上述挑战。

依据大规模图计算系统的使用场景以及计算平台架构的不同，我们将其分为单机内存图计算系统、单机外存图计算系统、分布式内存图计算系统和分布式外存图计算系统。

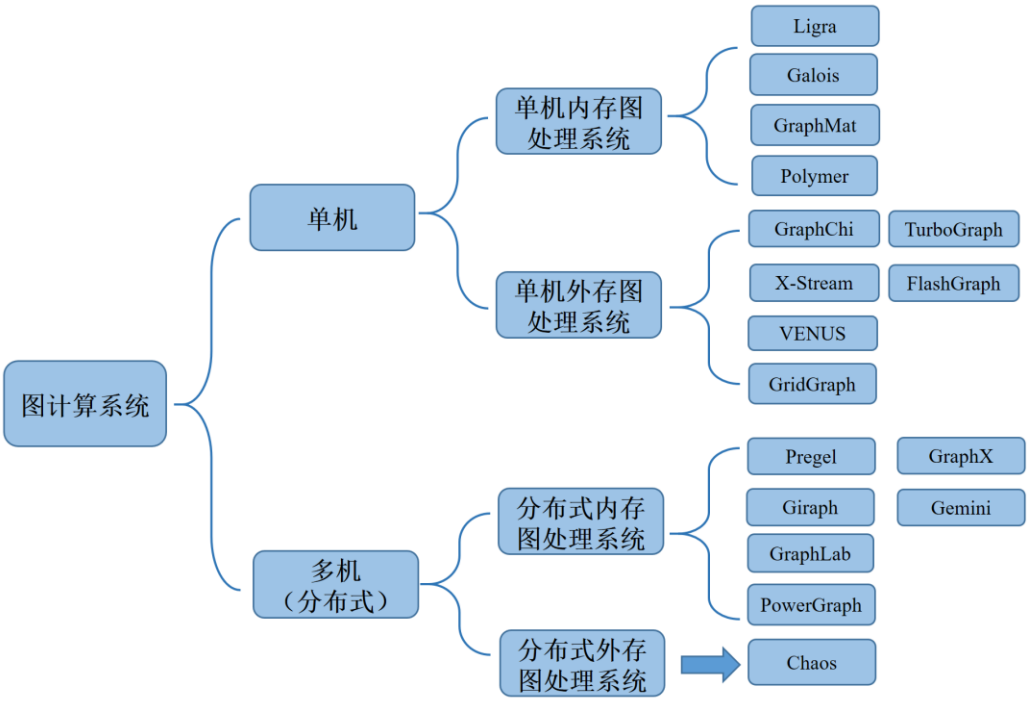


图 4 计算系统框架分类

单机内存图处理系统就是图处理系统运行在单机环境，并且将图数据全部缓冲到内存当中。单机外存图处理系统就是图处理系统运行在单机环境，并且通过计算将图数据不断地与内存和磁盘进行交互的高效图算法。分布式内存系统就是图处理系统运行在分布式集群环境，并且所有的图数据加载到内存当中。分布式外存图计算系统将单机外存系统（Single-machine out-of-core systems）拓展为集群，能够处理边数量级为 trillion 的图。

下面对各类典型的图计算系统逐一做简单介绍：

- 单机内存图处理系统

此类图计算系统单机运行，可直接将图完全加载到内存中进行计算。但是单机的计算能力和内存空间总是有限，故只能解决较小规模的图计算问题，比较有代表性的系统有 2013 年发布的 Ligra 和 Galois，以及 2015 年发布的 GraphMat 和 Polymer。

其中 Ligra 提出了根据图稠密情况自适应的切换计算模式，并提供了一种基于边映射，顶点映射以及顶点集映射的并行编程算法。Galois 使用 DSLs（domain-specific languages）写出更复杂的算法完成图分析工作，并发现当输入图是道路网络或者具有较大直径的图时能获得一个数量级的性能提升，在现有的三种图 DSLs 基础上提供了轻量级的 API，简化了图算法的实现。GraphMat 是第一个对多核 CPU 进行优化的以顶点为编程中心的轻量级图计算

框架，为用户和开发者提供了友好的接口。Polymer 则是针对在 NUMA 特性的计算机结构上运行图算法的优化，作者发现无论是随机或者交错地分配图数据都会重大地束缚数据的本地性和并行性，无论是 intra-node 还是 inter-node，顺序访存都比随机访存的带宽高的多。

- 单机外存图处理系统

此类图计算系统单机运行，但是将存储层次由 RAM 拓展到外部存储器如 SSD、Flash、SAS、HDD 等，使其所能处理的图规模增大。但受限于单机计算能力和外存存储系统的数据交换的带宽限制，该类系统无法在可接受的情形下处理超大规模的图数据。典型的单机外存图计算系统有 GraphChi、TurboGraph、X-Stream、PathGraph、GridGraph 和 FlashGraph。这些系统在最大化磁盘顺序读写、选择调度和同异步计算模式等方面做出了重要探索，TurboGraph 和 FlashGraph 主要采用分页方式分割图来提高内外存的数据交换性能。

✧ GraphChi

GraphChi 采用了传统的以顶点为中心的编程模型，计算模式为隐式 GAS。它使用了名为 shard 的外存数据结构来存储边，而将顶点划分为多个连续的区间，提出了一种基于并行滑动窗口（PSW）模型达到对存储在磁盘上的图数据最大的顺序读写性能。但是构建 shard 是需要对边按源顶点排序，这样耗费了大量的预处理时间，PSW 对计算密集型的算法更有利。另外在构建子图时出现大量的随机访存现象，通过顺序地更新子图内有共享边顶点来避免数据争用问题。

GraphChi 在计算前首先会对图数据进行预处理，将输入的图划分成多个 shard，每个 shard 中存储对应点集的所有入边，并且将入边按照其源节点的 ID 进行排序，划分时需要保证每个 shard 中边的数量大致相同，每个 shard 都能够加载进内存。GraphChi 使用以点为中心的计算模型，使用并行滑动窗口（parallel sliding window）来加载数据进行计算。每次（interval）计算一个子图，即一个 shard 所对应点集中所有点的值，需要顺序读取某个点集对应的入边（深灰色部分）以及该点集在其他 shard 中所对应的出边（黑色矩形框部分），这种数据布局和计算模型可以保证每次计算的 I/O 是顺序的。这样，一次迭代计算整个图中所有点的值，多次迭代，直到算法收敛。

✧ X-Stream

X-Stream 则介绍了一种以边为中心的编程模型。在 scatter 阶段以流的形式处理每条边和产生传播顶点状态更新集，在 gather 阶段它以流的形式处理每一个更新并应用到对应的顶点上。自然图中顶点集远远大于边集，所以 X-Stream 把顶点存储在高速存储设备（Cache 对于 RAM，RAM 对于 SSD/Disk）中表现为随机读写，把边集和更新集存于低速存储设备中表现为最大程度的顺序读写。X-Stream 流式访问图数据，其流划分相比于 GraphChi 无需

对 shard 内的边进行排序大大缩短了预处理时间，并使用 work-stealing 避免 Scatter-Gather 导致的线程间负载不均衡的问题。但是 X-Stream 在计算过程中，每轮迭代产生的更新集非常庞大，接近于边的数量级；而且需要对更新集中的边进行 shuffle 操作；缺乏选择调度机制，产生了大量的无用计算。

✧ VENUS

从 2013 年起，华为诺亚方舟实验室启动了大规模图数据挖掘的研究项目，负责研发图计算系统、图挖掘算法和商用。学术上，该项目已产出了若干研究成果，取得了一定的技术突破；商业上，在华为的业务中获得了重要应用。其中，所开发的图计算系统 VENUS 成为业界最快的单机图计算平台之一，能高效支撑各种大规模图数据挖掘应用。目前，VENUS 系统在华为手机应用市场中分析 10 亿量级的用户日志数据，支撑了重要的广告推送目标客户选择和手机应用推荐业务。

当前图计算面临几大主要挑战是单机系统具有很高的磁盘 IO 并且无法有效利用较大的内存；而分布式系统需要处理很难的图分割问题，并且计算过程会产生很高的网络开销以至于成为系统整体的扩展性能瓶颈，GraphLab 甚至会出现在增加节点时处理时间反而变长的情形。为了应对这些挑战，VENUS 图计算系统从最初设计开始考虑了一种新的扩展模式：纵向扩展+横向扩展。特别考虑在先不往外扩展（scale out）数据的情况下，先充分发挥单个计算节点往上（scale up）图计算系统的能力，再去大规模计算集群上往外扩展（scale out）多节点上的同质的计算。以处理计算量大，复杂度高的图算法。

VENUS 系统同时使用了新的数据分片方法以及新的图计算的外存模型。在该系统中，每一个数据分片被分为 v-shard 和 g-shard 两个部分：其中 v-shard 为一个分片中的顶点数据，数据量较小且需要经常修改；g-shard 为分片中图的结构即边数据，数据量较大且不会被修改。VENUS 进而在新的数据分片上创造了新的图计算的外存模型：在图计算执行中时，系统对全部 g-shard 使用顺序磁盘 IO 去扫描，不断从磁盘扫描 g-shard 磁盘块读入内存，并一直有该 s-shard 执行所需的某一个 v-shard 全部存放于内存。处理 g-shard 中所有对应的更新函数时，所有相关的所需的顶点数据都已经在内存中对应的 v-shard 中供直接访问存取。注意到 g-shard 按磁盘块读入就立刻计算并从内存丢弃，以保证更多的 v-shard 数据可以进到内存。这样，VENUS 能够使用较快的顺序访问 IO 带宽读取整个图中边的数据（g-shard），而且系统在读入数据的同时也可以及时的进行计算，减少了系统整体的运行时间。

✧ GridGraph

GridGraph 将顶点划分为 P 个顶点数量相等的 chunk，将边放置在以 P*P 的网格中的每一个 block 中，边源顶点所在的 chunk 决定其在网格中的行，边的顶点所在的 chunk 决定其

在网格中的列。它对 Cache/RAM/Disk 进行了两层级的网格划分，采用了 Stream vertices and edges 的图编程模型。计算过程中的双滑动窗口（Dual Sliding Windows）大大减少了 I/O 开销，特别是写开销。以 block 为单位进行选择调度，使用原子操作对保证线程安全的方式更新顶点，消除了 X-Stream 的更新集和 shuffle 阶段。但是，其折线式的边 block 遍历策略不能达到最大化的 Cache/Memory 命中率。

- 分布式内存图处理系统

此类图计算系统将图数据全部加载到集群中的内存中计算，理论上随着集群规模的增大其计算性能和内存容量都线性增大，能处理的图数据也按线性扩大。图分割的挑战在分布式系统愈加明显，再加上集群网络总带宽的限制，所以整体性能和所能处理的图规模也存在一定的缺陷。这类图计算系统主要包括同步计算模型的 Pregel 及其开源实现 Piccolo，同时支持同步和异步的系统 PowerGraph、GraphLab 和 GraphX。PowerSwitch 和 PowerLyra 则对 PowerGraph 做了改进，Gemini 则借鉴了单机内存系统的特性提出了以计算为核心的图计算系统。以下对部分分布式内存图处理系统做一介绍。

✧ Pregel

为了更有效地解决大规模图上的计算问题，学术界与工业界提出了大量专门为图优化的计算系统。Pregel 是来自 Google 的图计算领域的开山之作，是首个采用 BSP 计算模型的分布式内存图计算系统，计算由一系列的“超步”（super-step）组成，在一个超步内并行地执行用户自定义函数。很多后续的图计算系统均借鉴了其中的核心思想，例如“以顶点为中心”（vertex-centric）的编程模型，让用户将计算过程抽象为基于顶点的计算和基于边的消息传递（message passing），以及 BSP 的处理模型，顶点之间并行处理（计算和通信），通过超步（super-step）之间的路障 Barrier 来同步计算过程。Giraph 是 Pregel 的一个开源实现，在 Facebook 内部进行了大规模的部署与应用。

在 Pregel 中，用户可以自定义点的 compute( )函数，每个点多次迭代执行这个函数，并最终得出整个图的计算结果。具体地，在每一次迭代（superstep）中，每个活跃的点（active vertex）会执行 compute( )函数，在这个函数中，该点读取在前一次迭代中其邻点发送的消息，通过这些消息计算自己新的状态，再将自己最新的状态通过出边发送给其邻点（邻点将会在下一次迭代中收到这些消息），然后该点会进入不活跃状态（inactive）。当不活跃的点（inactive vertex）在下一轮收到消息时，就会重新处于活跃状态。当所有活跃的点执行完 compute( )函数之后，当前迭代结束，并且进入到下一次迭代。如果系统当中所有的点都处于不活跃状态，并且没有任何新的消息，算法结束。

✧ Giraph

Giraph 构建在 Hadoop 之上，是对 Google 公司 Pregel 的开源实现。Facebook 使用 Giraph 来进行社交关系图的分析。为了提升系统的性能，在原有 Giraph 基础上增加了一些优化的措施。Facebook 在 Giraph 的加载图数据、写回图数据以及计算阶段引入了多进程，提升了系统的整体性能，尤其对计算密集型的应用，引入多线程可以使性能随着处理器的增加获得接近线性的加速比。

✧ GraphLab

与 Pregel 的同步数据推送的 BSP 模型不同，GraphLab 使用异步的 GAS 模型来实现大图分布式并行计算。GraphLab 使用共享内存（shared memory）的方式来实现以点为中心的计算模式，在这种方式下，每个点可以直接读取和修改其邻点和邻边的值。在 GraphLab 上实现算法时，用户需要实现符合算法要求的 GAS 函数，在算法执行时，图的每个点都会执行该函数。在 gather 阶段，每个执行 GAS 函数的活跃点从其邻点和邻边获取数据，然后使用这些值来计算自己的更新值，这里计算操作必须满足交换律和结合律。在 apply 阶段，活跃点将原来的旧值更新为计算得到的新值。在 scatter 阶段，活跃的点会通过邻边激活对应的邻点。在 GraphLab 中使用一个全局的调度器，各个工作节点通过从该调度器获取活跃的点来进行计算，这些正在被计算的点也可能会将其邻点调入调度器中。最后当调度器中没有任何可调度的点时，算法终止。这种调度器的使用使得 GraphLab 同时支持算法的异步调度执行和同步调度执行。在同步执行（synchronous execution）计算模式下，每个点或者边的更新不能马上被当前迭代中接下来的计算感知到，直到当前迭代结束时，在下一次迭代当中才能读取到更新的值。异步执行（asynchronous execution）与同步执行不同，点或者边的更新能够马上被接下来的计算所感知并使用到，这种计算模式可以使得如 PageRank 的一些算法收敛速度更快，但也同时会导致数据竞争，从而产生额外的计算开销。另外，在分布式系统中，这种模式会产生随机的信息传递，因而也会产生较大的通信开销。一般来说，对于计算密集型的算法（如 BP）来说，更适合使用异步计算的模式。

✧ PowerGraph

PowerGraph 是面向分布式内存的解决方案，通过使用更多的机器来扩展能够处理的图的规模。PowerGraph 包含在 GraphLab 2.2 中，是在 GraphLab 的基础上对符合幂律分布（power-law）的自然图计算性能的改进。PowerGraph 的一个重要贡献是提出了基于“顶点切割”（vertex-cut）的图划分思想，通过在不同机器上创建顶点的多个副本（replica），以主－从（master-mirror）副本间的同步来替代传统的沿着边传递消息的通信模式，有效地减少了通信量以及由度数较高顶点导致的负载不均衡。后续的很多分布式图计算系统比如 GraphX、PowerLyra 等均沿用了 PowerGraph 的处理模型。

✧ GraphX

GraphX 是一个基于 Apache Spark 的嵌入式图计算框架（Apache Spark 是一种广泛使用的分布式数据流系统）。GraphX 支持 Pregel 和 GraphLab 的计算模型，并且拓展了 Spark 中的 RDD（resilient distributed dataset，弹性分布数据集），引入了 RDG（resilient distributed graph，弹性分布图），这种结构可以支持许多图操作，因此现有的大多数图算法都可以使用系统中提供的基本操作算子（如 join、map 和 group-by）来实现，并且实现十分简单。为了利用 Spark 中这种算子操作，GraphX 重构了新的 vertex-cut 图划分方法，将图划分成水平分区的顶点和边的集合。GraphX 的性能比直接使用分布式数据流框架好一个数量级，稍差于 GraphLab。GraphX 提供了一个熟悉的可组合图抽象，足以表达现有的图 API，但只能使用少数基本数据流操作符来实现。为了实现与专用图系统的性能平衡，GraphX 将图特定的优化重新分配为分布式连接优化和材料化视图维护。通过利用分布式数据流框架的进步，GraphX 为图处理带来了低成本的容错能力。

✧ Gemini

Gemini 是在单机内存图计算系统的高效性和分布式内存图计算系统的伸缩性之间找到差异性，从而提出的一个以计算为核心优化目标的分布式图计算系统。其主要关注点为计算而非通信，一方面尽可能避免分布式带来的开销，另一方面吸纳现有单机系统的技术来实现高效的计算。Gemini 针对图结构的稀疏或稠密情况使用与 Ligra 相同的自适应推动/拉动（push/pull）方式的计算，将推动（push）模式和拉动（pull）模式的双模式计算引擎从单机的共享内存扩展到了分布式环境中，从而将分布式系统的通信从计算中剥离出来。此外，除了节点间的负载均衡，Gemini 在节点内也通过细粒度的块式划分结合工作偷取（work stealing）的方式，使得多核间的负载尽可能均衡。在性能方面，Gemini 尽可能地减少了分布式的开销，使得运行于单机时的效率能够接近甚至超过现有最佳性能的单机系统，比现有已知分布式图计算系统的性能也提高了数倍。

- 分布式外存图处理系统

此类图计算系统将 Single-machine out-of-core systems 拓展为集群，能够处理边数量级为 trillion 的图，目前有 2015 年发布的 Chaos。Chaos 是对 X-Stream 系统的拓展，分别设计了计算子系统和存储子系统。它的主要贡献表现在：是第一个拓展到多机外存存储结构的图计算系统；采用简单的图分割方案，即不强调数据的本地性和负载均衡，而是通过存储子系统达到外存存储的高效顺序读写；使用 work-stealing 机制实现动态负载均衡。但是 Chaos 的信息量过大，随着集群规模的扩大，网络将会成为瓶颈；简单的拓展未优化的 X-Stream，其更

新集依然很庞大与边量级相当；计算与存储独立设计增加了系统的复杂性和不可避免的通信开销；存储子系统为了使存储设备时刻忙碌而占用了较多的计算资源。

- 面向机器学习的分布式图计算系统

TuX² 作为一个全新的分布式图引擎，致力于融合图计算和分布式机器学习系统。TuX² 继承了传统图计算系统中的优势：简洁的计算模型，高效的数据排布，均衡的负载分配以及超过 10 亿条边的规模处理能力；并对于分布式机器学习进行了大幅扩展和优化，以支持异质性、延时同步并行。TuX² 上实现了一系列具有代表性的分布式机器学习算法，涵盖了监督学习和非监督学习，具有一个量级上的性能优势。

表 3 图计算系统概览

年份	系统	编程模型	架构	运算模型	通信模型	调度	存储方式
2009	PEGASUS	N/A	D	N/A	DF	Synch	DB
2010	Pregel	V	D	BSP	MP	Synch	DB
2010	Signal/Collect	V	S	Signal/Collect	MP	Both	DB
2010	Surfur	V	D	Transfer-combine	MP	Synch	DB
2010	JPregel	V	D	BSP	MP	Synch	DB
2010	GraphLab	V	S	N/A	SM	Asynch	DB
2010	Piccolo	Da	D	Three Phases	DF	Synch	DB
2011	GoldenOrd	V	D	BSP	SM	Synch	DB
2011	GBase	E	D	N/A	DF	Synch	DB
2011	HipG	V	D	BSP	SM	Both	DB
2012	Giraph	V	D	BSP	MP	Synch	DB
2012	Distributed GraphLab	V	D	GAS	SM	Both	DB
2012	KineoGraph	V	D	Push/pull	MP	Synch	MB
2012	PowerGraph	E	D	GAS	SM	Both	DB
2012	Sedge	V	D	BSP	MP	Synch	DB
2012	GraphChi	V	S	PSW	SM	Asynch	DB
2013	TOTEM	V	H	BSP	MP&SM	Asynch	MB
2013	Mizan	V	D	BSP	MP	Synch	DB
2013	Trinity	V	D	TSL	SM	Asynch	MB
2013	Grace	V	S	Three phases	MP	Asynch	DB
2013	GPS	V	D	BSP	MP	Synch	DB
2013	Giraph++	C	D	BSP	MP&SM	Both	DB
2013	Naiad	V	D	Timely dataflow	SM	Both	MB
2013	PAGE	V	D	Partition-aware	MP	Synch	DB
2013	Stratospher	V	D	Push/pull	DF	Synch	DB
2013	TurboGraph	V	S	Pin-and-slide	SM	Asynch	DB
2013	xDGP	V	D	BSP	MP	Synch	DB
2013	X-Stream	E	S	Scatter-gather	MP	Synch	DB
2013	GiraphX	V	D	BSP	SM	Asynch	DB
2013	GraphX	E	D	GAS	DF	Synch	MB
2013	Galois	V	S	ADP	SM	Asynch	DB
2013	GRE	V	D	Scatter-combine	MP	Synch	DB
2013	Ligra	C	S	Push-pull	SM	Asynch	MB
2013	LFGreaph	V	D	N/A	SM	Synch	MB
2013	PowerSwitch	V	D	Hybrid	SM	Both	DB
2013	Presto	V	D	N/A	DF	Synch	DB

2013	Medusa	V	H	EMV	MP	Synch	MB
2014	RASP	V	S	Scatter-gather	SM	Asynch	DB
2014	GoFFish	C	D	IterativeBSP	MP&SM	Synch	MB
2014	GasCL	V	H	GAS	MP	Synch	MB
2014	CuSHa	V	H	GAS	SM	Asynch	MB
2014	BPP	V	S	BSP	SM	Asynch	DB
2014	Imitator	V	D	BSP	MP	Synch	DB
2014	GraphHP	V	D	BSP	MP	Synch	DB
2014	PathGraph	P	S	Scatter-gather	SM	Asynch	DB
2014	Seraph	V	D	GES	MP	Synch	DB
2014	GraphGen	V	H	N/A	SM	Synch	MB
2014	Blogel	B	D	N/A	MP	Synch	MB
2015	Pregelix	V	D	Join-operator based	MP	Synch	DB
2015	FlashGraph	V	S	BSP	MP&SM	Asynch	DB
2015	GraSP	V	D	N/A	MP	Synch	MB
2015	Chaos	E	D	GAS	MP	Synch	DB
2015	GraphMap	V	D	BSP	MP	Synch	DB
2015	GridGraph	E	S	Streaming-Apply	SM	Asynch	DB
2015	GraphTwist	E	S	Slice/Cut pruning	SM	Asynch	DB
2015	GraphQ	V	S	Check/Refine	SM	Asynch	DB
2016	Gunrock	Da	H	BSP	SM	Synch	MB
2016	GraphIn	V	D	I-GAS	MP	Synch	MB
2016	LCC-Graph	V	D	LLC-BSP	MP	Synch	MB
2016	DUALSIM	V	S	N/A	SM	Asynch	DB
2016	iGiraph	V	D	BSP	MP	Synch	DB
2017	GraphMP	V	S	VSW	SM	Asynch	DB
2017	GraphGen	V	S	N/A	SM	Asynch	MB
2017	Mosaic	V/E	S	PRA	MP	Synch	DB

-编程模型：以顶点为中心（V），以边缘为中心（E），以组件为中心（C），以路径为中心（P），以数据为中心（Da），以块为中心（B）
-架构：分布式（D），单机（S），异构（H）
-运算模型：不同系统使用不同的名称
-通信模型：消息传递（MP），存储器共享（SM），数据流（DF）
-调度：同步（Synch），异步（Asynch），混合调度（Both）
-存储方式：基于磁盘（DB），基于内存（MB）
-N/A 表示没有特定的名称或定义

2.4 图计算中的关键技术

本节将重点介绍在分布式和单机图处理系统中常用的技术。

- 异构计算平台

在异构计算系统中，存在着计算能力和计算特点不同的计算单元。比如，GPU 具有比 CPU 更强的多线程并行计算能力，因此在异构系统中，CPU 会把一些或者全部的计算交给 GPU 来执行。在图计算领域，相关的异构计算系统已经被开发出来。TOTEM 会将度高的点交给 CPU 计算执行，而将度低的点交给 GPU 来执行。而另外一些系统，如 MapGraph 和 CuSha 等，会将整个图都交给 GPU 来执行。除了 GPU 和 CPU 的异构图计算平台之外，一些研究人员发现，solid-state drive（SSD）有着与传统 hard disk drive （HDD）不同的访存特

性。一些图计算系统（如 TurboGraph 和 FlashGraph）针对 SSD 对计算系统进行了优化，使得系统在 SDD 上有着很高的计算性能。目前使用异构计算的平台的图处理系统主要是单机图处理系统。

- 通信模型

在消息传递的通信模型中，算法中点的状态保存在本地，通过消息传递的方式更新在其他机器上点的状态。在 Pregel 和 Giraph 中，使用了消息传递的通信模型，为了确保所有更新的数据可用，需要在前后两次迭代计算之间加入一个同步操作。在共享内存的通信模型中，各个处理单元允许并发访问和修改相同地址的数据。在一些分布式的计算系统（如 GraphLab 和 PowerGraph）中，使用了虚拟共享内存来实现各计算节点之间的透明的同步。在这些图处理系统中，使用了假点（ghost vertex）的方式来实现虚拟共享内存。在假点的这种实现策略中，图中的每个点有一个归属的工作节点，另外，有一些工作节点拥有该点的副本。因此，在这种通信模型中，当多个工作节点并发访问同一内存地址时，需要考虑数据一致性的问题。

- 执行模型

执行模型主要分为两类：同步执行和异步执行。同步执行指的是，许多图算法由一系列迭代计算组成，在前后两次迭代之间有一个全局的同步过程。这种执行模式将计算节点之间的通信控制在每次迭代的结束，因此适合于那些计算量小而通信量大的算法。异步执行指的是在图中某个点的值有了更新值之后，立即将这个最新的更新值更新到该点上。在这种执行模式中，节点之间的通信是不规则的，因此这种模式对于计算量不均衡，并且节点之间通信量小的算法非常适用。

- 图的划分

图的划分是进行高效图计算的一个关键问题。通常，一个理想的图划分情况是各工作节点的任务量基本相同，同时各工作节点之间的通信量最小，但是这是一个 NP 难的问题。现在，常用的图划分算法分为三类。第一类，首先对输入的图数据进行一个预处理，将初始的图数据转化为某个特定的存储格式，使得图计算的访存局部性更好或者使图数据的数据量占用更少。比如，GraphChi 使用 shard 以及 shard 内存源点的排序来增强磁盘访存的局部性。另外，X-Stream 使用简单的流划分来降低预处理的开销。第二类，在算法执行过程中使用动态的重划分，由于算法在执行之前行为是无法预测的，所以这种动态划分的策略可以根据现有算法的执行状态进行相应地划分，提高系统的性能。这种动态划分策略需要对图进行多次划分，引入了图划分开销。第三类，使用 edge-cut 和 vertexcut 划分。edge-cut 将图中的点均匀地划分，并且保证跨不同划分块之间的边最少。vertex-cut 将边均匀地划分，同时保证跨不同块之间的点最少。现实生活中的许多大图符合幂律分布，因此，相比于 edge-cut，使用

vertex-cut 有助于系统的负载均衡，但是图计算系统需要使用以边为中心的计算模型，如PowerGraph。

- 负载均衡

负载均衡的算法分为静态负载均衡和动态负载均衡，静态负载均衡在算法执行之前进行任务的分配，但是由于算法在执行之前无法预测其具体的行为，因而在算法的执行过程中可能出现负载不均衡的情况。动态的负载均衡策略针对静态负载策略进行了改进，即在算法的运行过程中，系统中任务少的工作节点可以从任务量大的工作节点“偷取”任务来实现负载均衡，提高系统的整体性能。

- 容错

容错在分布式图处理系统中是需要解决的一个问题。在分布式处理系统中，每台机器都会有一定的概率出错失效，如果不加以处理，将对系统产生严重的影响。常见的分布式图处理系统使用主从节点的方式，在这种构建方式中，主节点负责整个系统的管理和调度，从节点负责具体的计算。主要的容错方式有多副本策略、日志重做策略等。在多副本策略中，当主工作节点执行其任务时，另外，有一个工作节点作为副本工作节点会执行相同的任务；当主节点失效时，副本会接管主节点的工作任务，这种容错方式基本没有错误恢复时间，但是会消耗掉很多计算和内存资源。在日志重做的策略中，使用 checkpoint 或者 log 的方式记录工作节点的计算操作，当机器出现失效时，可以将记录的操作重做来进行恢复，这种恢复方式会消耗一定的恢复时间，但是对计算和内存资源的消耗相对较少。

2.5 技术挑战

图提供了非常灵活的抽象，用于描述离散对象之间的关系。科学计算、数据分析和其他领域的许多实际问题可以通过图以其基本形式建模，并通过适当的图算法求解。随着图的问题规模越来越大，复杂性越来越大，它们很容易超过单处理器的计算和内存容量。鉴于并行计算在许多科学计算领域取得了成功，并行处理似乎可以克服图计算中单个处理器资源受到的限制。

当整体计算问题解决方法得到很好的平衡时，应用程序可以更好地执行和扩展，即，当需要解决的问题、用于解决问题的算法、用于表达算法的软件以及运行软件的硬件使两者都能很好地相互匹配。在很大程度上，并行科学计算的成功归功于这些方面，与典型的科学应用完全匹配。解决科学领域中典型问题（通常涉及求解偏微分方程系统）的常用习语已经发展并成为科学计算界的标准实践。同样，适用于典型问题的硬件平台和编程模型也变得很普遍。比如，世界各地的机房包含运行用 MPI 编码的商用集群。

然而，尽管分布式图处理系统随着问题规模的扩大具有很好的拓展性，在提高系统处理效率方面仍然面临着许多挑战。比如图的划分，要提高系统性能需要在保证集群各节点负载均衡的情况下，使得集群内各节点的通信量最少，是一个 NP 难问题。此外，一个分布式系统需要解决集群内各节点协同工作、容错等一系列问题，而这类问题对系统的性能有重要的影响。另一方面，对于使用分布式系统的程序员来说，环境的搭建、编写分布式程序比较复杂，而且程序的调试和优化又相对困难。

可以说，对于开发主流并行科学应用程序而言，效果良好的算法、软件和硬件对于大规模图问题并不一定有效。图问题具有一些固有的特征，使它们与当前的计算问题解决方法不匹配。大图计算是大数据计算中的一个子问题，除了满足大数据的基本特性之外，大图计算还有着自身的计算特性，相应地面临着新的挑战。特别是，图问题的以下属性对高效并行性提出了重大挑战。

- **局部性差**

图表示着不同实体之间的关系，而在实际的问题当中，这些关系经常是不规则和无结构的，因此图的计算和访存模式都没有好的局部性，而在现有的计算机体系架构上，程序的性能获得往往需要利用好局部性。所以，如何对图数据进行布局和划分，并且提出相应的计算模型来提升数据的局部性，是提高图计算性能的重要方面，也是面临的关键挑战。

- **数据及图结构驱动的计算**

图计算基本上完全是由图中的数据所驱动的。当执行图算法时，算法是依据图中的点和边来进行指导，而不是直接通过程序中的代码展现出来。所以，不同的图结构在相同的算法实现上，将会有着不同的计算性能。因此，如何使得不同图结构在同一个系统上都有较优的处理结果，也是一大难题。

- **图数据的非结构化特性**

图计算中图数据往往是非结构化和不规则的，在利用分布式框架进行图计算时，首先需要对图进行划分，将负载分配到各个节点上，而图的这种非结构化特性很难实现对图的有效划分，从而达到存储、通信和计算的负载均衡。一旦划分不合理，节点间不均衡的负载将会使系统的拓展性受到严重的限制，处理能力也将无法符合系统的计算规模。

- **高访存/计算比**

绝大部分的大图计算规模使得内存中无法存储下所有的数据，计算中磁盘的 I/O 必不可少，而且大部分图算法呈现出迭代的特征，即整个算法需要进行多次迭代，每次迭代需要遍

历整个图结构，而且每次迭代时所进行的计算又相对较少。因此，呈现出高的访存/计算比。另外，图计算的局部性差，使得计算在等待 I/O 上花费了巨大的开销。

2.6 技术资源

Open Academic Graph（OAG）是通过链接 Microsoft Academic Grapg（MAG）和 AMIner 两个大型学术图表生成的。（https://www.aminer.cn/oag2019）我们将 OGA2019 的统计数据呈现为下面三个图，包括 Venue Collection、Paper Collection 和 Author Collection 三部分。

Venue Collection

Table 1: statistics of OAG venue data

Data set	#Pairs/Venues	Date
Linking relations	29,841	2018.12
AMiner venues	69,397	2018.07
MAG venues	52,678	2018.11

图 5 Venue Colletion of OAG

Paper Collection

Table 2: statistics of OAG paper data

Data set	#Pairs/Papers	Date
Linking relations	91,137,597	2018.12
AMiner papers	172,209,563	2019.01
MAG papers	208,915,369	2018.11

图 6 Paper Collection of OAG

Author Collection

Table 3: statistics of OAG author data

Data set	#Pairs/Authors	Date
Linking relations	1,717,680	2019.01
AMiner authors	113,171,945	2018.07
MAG authors	253,144,301	2018.11

图 7 Author Collection of OAG

2.7 高引论文

下边是一些图计算相关引用量较高的论文：

Pregel: a system for large-scale graph processing Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, Grzegorz Czajkowski SIGMOD Conference,, pp. 135-146, (2010)
Distributed GraphLab: a framework for machine learning and data mining in the cloud Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, Joseph M. Hellerstein Proceedings of the VLDB Endowment, no. 8 (2012): 716-727
PowerGraph: distributed graph-parallel computation on natural graphs Joseph E. Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, Carlos Guestrin OSDI,, pp. 17-30, (2012)
GraphLab: A New Framework For Parallel Machine Learning. Yucheng Low, Joseph Gonzalez, Aapo Kyrola, Danny Bickson, Carlos Guestrin, Joseph M. Hellerstein Clinical Orthopaedics and Related Research, (2014)
GraphChi: large-scale graph computation on just a PC Aapo Kyrola, Guy Blelloch, and Carlos Guestrin. GraphChi: large-scale graph computation on just a PC. OSDI, 2012.
Graphx: Graph processing in a distributed dataflow framework Joseph E. Gonzalez, Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin, Ion Stoica OSDI,, pp. 599-613, (2014)
X-Stream: edge-centric graph processing using streaming partitions Amitabha Roy, Ivo Mihailovic, Ivo Mihailovic,Willy Zwaenepoel SOSP,, pp. 472-488, (2013) https://static.aminer.org/pdf/20170130/pdfs/sosp/ozz12hqruancqbijue8b6pm7siewytf0.pdf
Ligra: a lightweight graph processing framework for shared memory Julian Shun, Guy E. Blelloch PPOPP,, pp. 135-146, (2013) https://static.aminer.org/pdf/20170130/pdfs/ppopp/gxrite4sseq1juzxofcanzbnplrhh5kv.pdf
PowerLyra: differentiated graph computation and partitioning on skewed graphs

Rong Chen, Jiaxin Shi, Yanzhe Chen, and Haibo Chen. PowerLyra: differentiated graph computation and partitioning on skewed graphs. EuroSys, 2015.
GridGraph: Large-Scale Graph Processing on a Single Machine Using 2-Level Hierarchical Partitioning Xiaowei Zhu, Wentao Han, Wenguang Chen USENIX Annual Technical Conference,, (2015)

3 人才篇

在大数据时代，图计算具有表达能力强、可扩展性支持、快速高效的并行计算能力等优势，能解决大容量的数据分析问题。图计算的研究风潮席卷了全球，无论是 Google 还是 Facebook，科技企业都在全力以赴地投入图计算领域的研究；在国内，进行相关研究的主要有清华大学、上海交通大学等高校以及阿里、华为等互联网企业。

图计算相关的重要会议包括：

OSDI（USENIX Symposium on Operating Systems Design and Implementation）

SOSP（Symposium on Operating Systems Principles）

USENIX ATC（USENIX Annual Technical Conference）

VLDB（International Conference on Very Large Data Bases）

SIGMOD（Special Interest Group on Management Of Data）

EuroSys（The European Conference on Computer Systems）

PPoPP（ACM SIGPLAN Annual Symposium Principles and Practice of Parallel Programming）等。

我们对这些会议近 10 年（2009 年至 2018 年）的论文进行挖掘，筛选出其标题、关键字以及摘要中涉及 graph、graph computing、graph pattern、graph mining、graph processing 等图计算领域关键词的论文，再对其作者进行挖掘，整理出图计算相关活跃学者 1000 人。

3.1 学者情况概览

本节对这些学者进行了简单的统计分析，包括他们的分布地图、迁徙状况、机构分布、h-index 水平、性别分布等。

图 8 全球图计算领域活跃学者分布图

由 AMiner 绘制的全球图计算领域活跃学者所在地区的分布地图（如图 8 所示）可以看出，图计算领域学者在北美洲最为集中，亚洲次之，欧洲西部该领域的人才分布也较多。

分国家来看，美国东部该领域学者较为集中，其次是中国，欧洲主要在德国和英国拥有该领域较多数量的学者。

图 9 是国内图计算领域活跃学者的分布图，由该图可见，国内学者集中分布在长三角、珠三角以及京津冀等地区。从全局来看，内陆地区的学者分布数量相对沿海地区要少，沿海地区珠三角是全国学者分布最为集中的地区，内陆地区的学者则主要集中分布在华北及我国东北三省地区，学者数量呈由南到北逐渐减少的分布状态。

图 9 中国图计算领域活跃学者分布图

除了地区分布，我们还对图计算领域学者的迁徙路径做了分析。由图 10 可以看出，美国是图计算领域人才流动大国，人才输入和输出幅度领先于其他国家，且从数据来看人才流入大于人才流出。中国、英国、印度等国人才迁徙流量小于美国，其中中国与印度有轻微的人才流失现象。除美国和中国外，其余各国图计算学者的流失和引进是相对比较均衡的。

人才的频繁流入流出，使得该领域的学术交流活动增加，带动了人才质量的提升的同时，也促进了领域理论及技术的更新迭代，至此逐渐形成一种良性循环的过程。

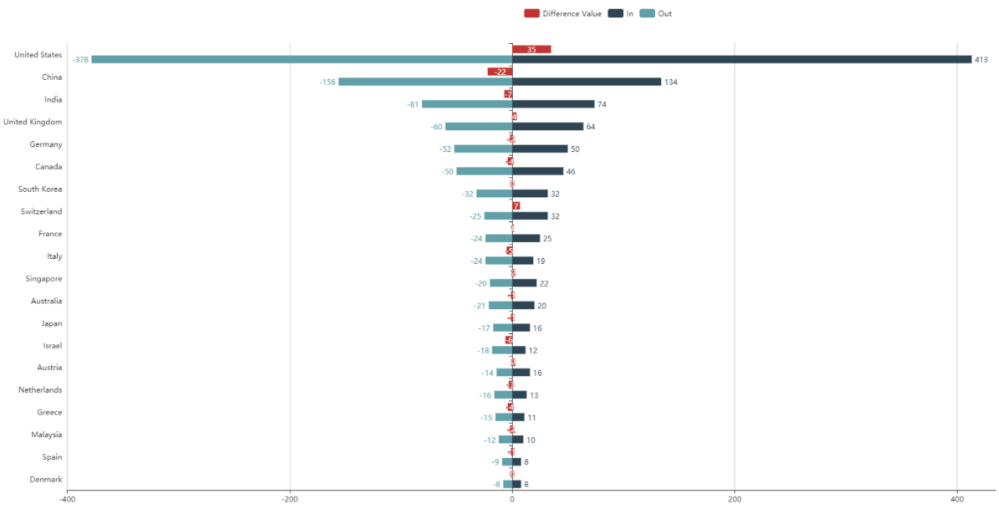


图 10 全球图计算领域活跃学者迁徙图

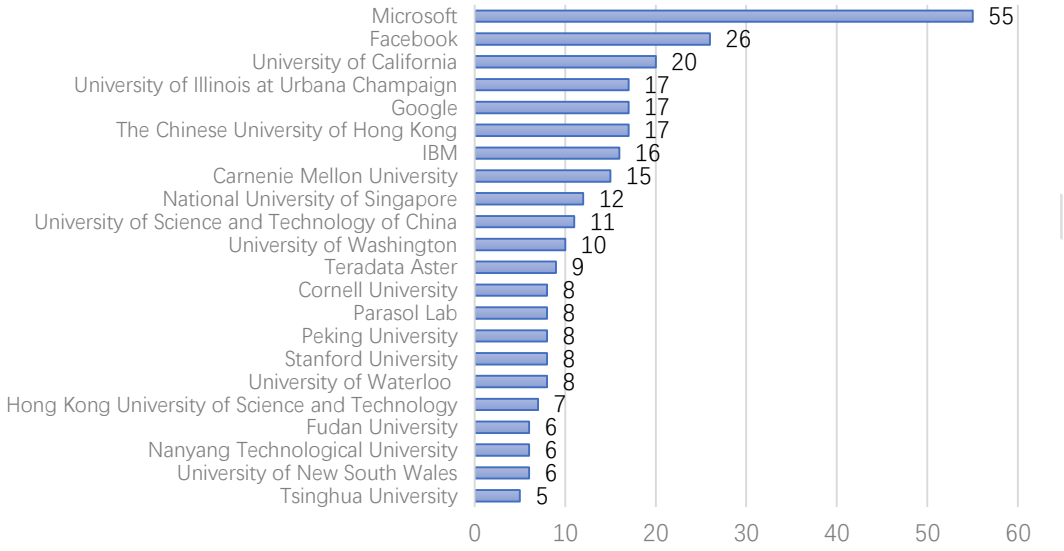


图 11 全球图计算领域活跃学者机构分布

如图 11 所示，是对全球计算机图计算领域学者所属机构的统计，数据显示多名学者就职于同一机构的现象较为普遍，机构所拥有的学者数量越多，一定程度上越能反映出该机构的科研与创新能力，直观地印证其在该领域具备更强的竞争力，因此我们对表现最突出的机构进行了以上排名。

在图计算领域，Microsoft（微软）拥有的影响力学者数量最多，共 55 人，位于榜首，Facebook 以 26 名位居第二，University of California（加利福尼亚大学）共拥有 20 名学者位居第三。中国拥有 10 位以上学者的机构有香港中文大学和中国科学技术大学，分别拥有 17 名和 11 名学者。中国拥有该领域学者数 10 人以下的机构分别有 Peking University（北京大学）、Hong Kong University of Science and Technology（香港科技大学）、Fudan University（复旦大学）以及 Tsinghua University（清华大学）。

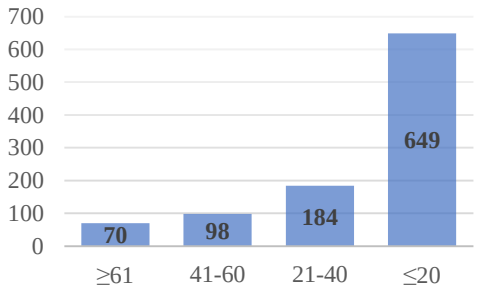


图 12 全球图计算领域活跃学者 h-index 分布

全球图计算领域学者 h-index 的平均数为 21，其中，h-index≥21 的学者占比 38.03%；可见，h-index 低于平均值的学者人数超过了一半，占比 61.97%。从每个 h-index 分度区间来看，h-index 在 20 以下区间的学者人数最多，有 649 人，占比 64.84%；h-index≥60 的人数最少，有 70 人，占比 6.99%。

图 13 全球图计算领域活跃学者性别比

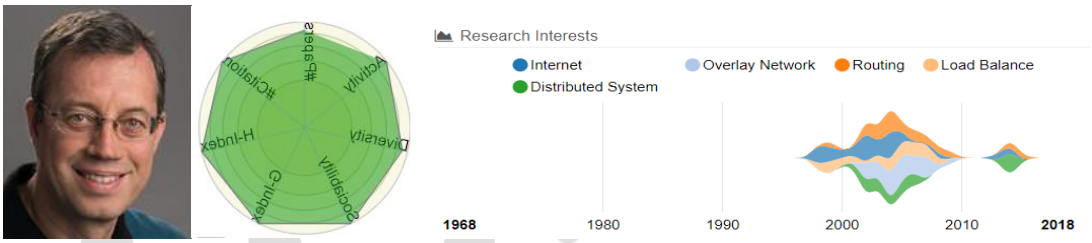
在性别比例方面，男性（占比 96.31%）占据多数，比例要远高于该领域女性所占比例（3.69%）。

3.2 典型学者简介

综合 h-index 以及领域知名度与活跃度，我们分别选取了国内外（按国籍）两部分图计算领域具有代表性的学者做一介绍，排名不分先后。由于本报告侧重于介绍图计算系统，因此代表性学者均选自该研究领域。此外，限于报告篇幅，我们对所有学者不能逐一罗列，如有疏漏，还请与 AMiner 编者联系，或者登录 https://www.aminer.cn/获取更多资料。

3.2.1 国外学者简介

- Ion Stoica



Ion Stoica，美国加州大学伯克利分校的计算机科学系教授，AMPLab 的共同创始人，Spark 核心作者。

他是一位罗马尼亚裔美国计算机科学家，专门研究分布式系统，云计算和计算机网络。他于 2000 年获得了卡耐基梅隆大学（CMU）Hui Zhang（张辉）教授的电气与计算机工程专业博士学位。其研究主题包括 Chord（点对点），核心无状态公平队列（CSFQ）和 Internet 间接基础设施。

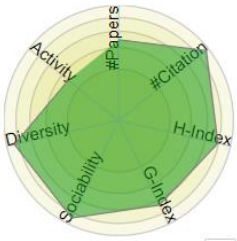
他的博士生导师 Hui Zhang 是 P2P 思想的重要先驱，论文《End System Multicast》的应用层覆盖网多播思想的合作提出者之一。Ion Stoica 在 Hui Zhang 的基础上进一步将 P2P 技术带到了互联网。其在 SIGCOMM'01 会议上发表的论文《Chord: A scalable peer-to-peer lookup service for internet applications》成为结构化 P2P 网络的开山之作。

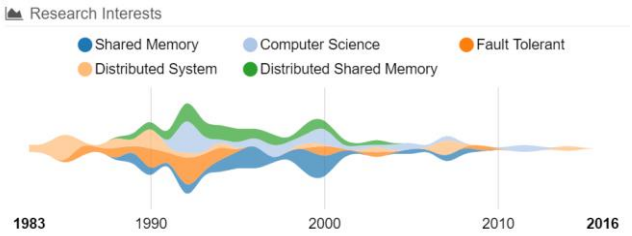
Ion Stoica 的研究兴趣包括云计算、网络、分布式系统和大数据。他在计算机科学的各个领域撰写或共同撰写了 100 多篇同行评审论文。

他在 2001 年获得了计算机协会博士论文奖；2002 年获得 PECASE（工程师总统早期职业奖）奖；2003 年获得斯隆基金会奖学金；2007 年获得 CoNEXT（International Conference on emerging Networking EXperiments and Technologies）新星奖；2011 年获得了 SIGCOMM（Special Interest Group on Data Communication 美国计算机协会 ACM 数据通信专业组）时间测试奖。

在加入加州大学伯克利分校担任终身教授之前，Ion Stoica 在麻省理工学院（MIT）担任博士后研究职位。2006 年，他成为 Conviva 的联合创始人兼首席技术官（CTO），该公司（Conviva）之前是 CMU 的 End System Multicast 项目。2013 年，他联合创立了 Databricks，担任首席执行官，直到 2016 年 1 月被 Ali Ghodsi 取代后他继而担任执行主席。此外，他也是 ACM（计算机协会）Fellow。

- Willy Zwaenepoel





Willy Zwaenepoel，ACM Fellow，IEEE Fellow，自 2018 年 6 月 15 日起被任命为悉尼大学工程与信息技术学院院长。

Willy Zwaenepoel 在 1984 年获得斯坦福大学博士学位。他从事过操作和分布式系统的各个方面的工作，包括微内核、容错、工作站集群上的并行科学计算、Web 服务集群、移动计算、数据库复制和虚拟化。他最著名的作品是 Treadmarks 分布式共享内存系统，该系统已获得英特尔许可，并成为英特尔 OpenMP 集群产品的基础。他在网络 I/O 高性能软件方面的工作促成了 iMimic Networking 公司的创建，2000 年至 2005 年公司由他领导。

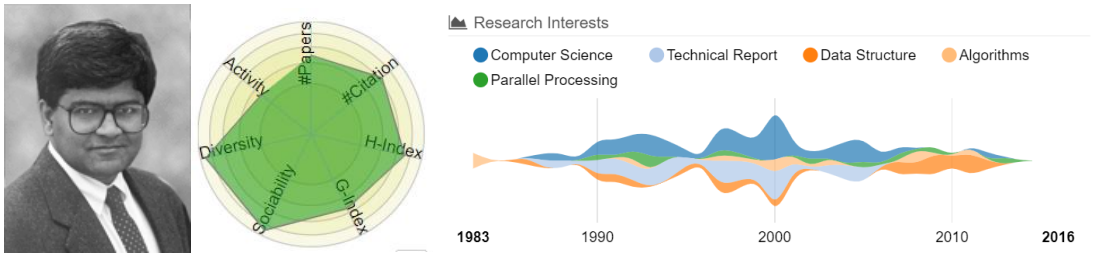
他目前的兴趣包括大规模数据存储和软件测试。他在软件测试方面的工作促成了一家位于洛桑的创业公司 BugBuster 的创建。因为他的研究成果，iMimic 和 BugBuster 已被思科收购。

他于 1998 年当选为 IEEE Fellow，2000 年当选为 ACM Fellow。他还是 1998 年至 2002 年 IEEE 并行和分布式系统交易的副主编。他是多个会议的项目主席：1996 年担任 OSDI 项目主席，2004 年任 Mobisys（The International Conference on Mobile Systems, Applications, and Services）主席，2006 年担任 Eurosys 项目主席。

2002 年 9 月，他加入了 EPFL（École polytechnique fédérale de Lausanne 洛桑联邦理工学院）。2002 年至 2011 年，他担任 EPFL 计算机与通信科学学院院长。在加入 EPFL 之前，他是莱斯大学（Rice University）的教师，担任计算机科学与电气和计算机工程系的 Karl F. Hasselmann 教授，并在 2000 年获得了莱斯大学研究生协会教学与指导奖。

他分别在 1984 年 SigComm，1999 年的 OSDI、Usenix 2000、Usenix 2006 和 Eurosys 2007 上获得了最佳论文奖。他还获得了 2007 年 IEEE Tsutomu Kanai 奖。

- Keshav K Pingali



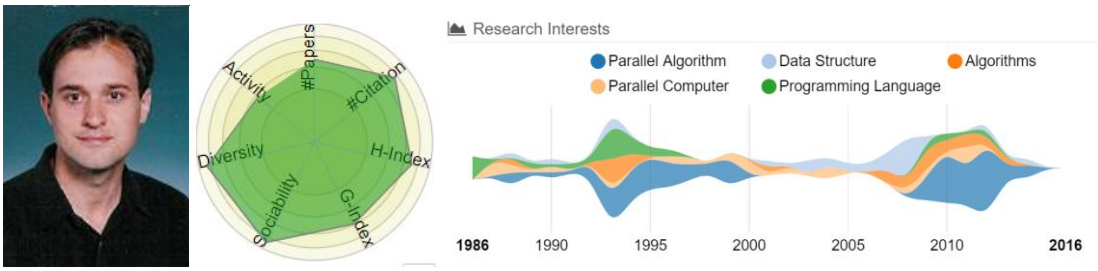
Keshav K Pingali，美国计算机科学家，ACM Fellow，IEEE Fellow，德克萨斯大学奥斯汀分校网格和分布式计算的 William Moncrief 主席。

Keshav K Pingali 于 1986 年获得麻省理工学院硕士和博士学位。他曾获得 2013 年 IIT Kanpur 杰出校友奖。

Keshav K Pingali 从事编程语言和编译器技术，用于程序理解，优化和并行化。他目前的研究兴趣是编程多核处理器的方法和工具，重点关注图，社交网络和数据挖掘等领域的非常规应用程序。

在成为德克萨斯大学奥斯汀分校网格和分布式计算的 William Moncrief 主席之前，他曾在 2003 年-2006 年任印度康奈尔大学计算机科学系主席，并在印度理工学院任 N. Rama Rao 教授。他还是 AAAS（美国科学促进会）、ACM 以及 IEEE Fellow。

- Guy E. Blelloch



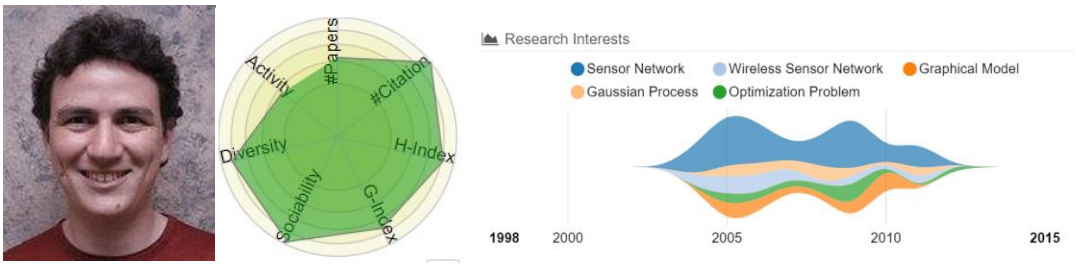
Guy E. Blelloch，ACM Fellow，卡内基梅隆大学计算机科学系教授，以并行编程和并行算法领域的研究而闻名。

Guy E. Blelloch 在卡内基梅隆大学教授现实世界中的算法、并行算法课程，以及并行和顺序数据结构和算法课程。

目前，他正在研究的主题主要有：基准并行算法、缓存高效的并行算法、数据结构和算法。他与合作研究者一直在开发用于计算并行机器上的局部性的高级算法模型，开发在这些

模型下有效的算法。此外，他还研究了用于图、索引、网格和集合的紧凑数据结构。由他的研究兴趣河流图可见，他对算法的研究未曾中断，并行算法在近 10 年成为他的重点研究领域。

- Carlos Guestrin



Carlos Guestrin，机器学习界知名学者，Turi（曾用名 GraphLab 和 Dato，后更名为 Turi）创始人，曾被 Popular Science（大众科学）杂志评为 2008 年“Brilliant 10”（“辉煌 10 强”）。

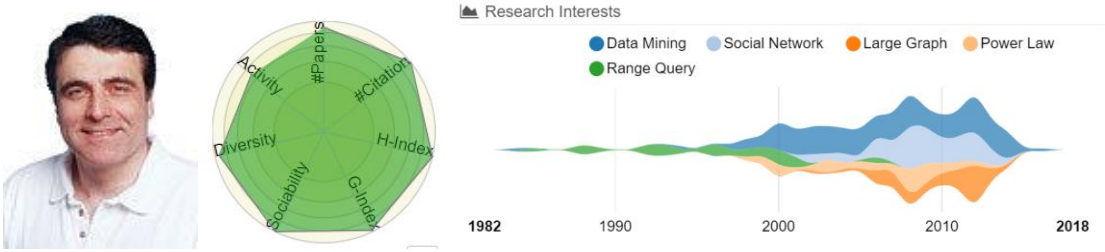
Carlos Guestrin 拥有斯坦福大学的博士学位，是华盛顿大学 Paul G. Allen 计算机科学与工程学院的亚马逊机器学习教授。Carlos Guestrin 还曾任职卡内基梅隆大学的 Finmeccanica 副教授和英特尔伯克利研究实验室的高级研究员。

他还是 Turi（最初是 GraphLab）的联合创始人兼首席执行官。Turi 最初是 2009 年卡耐基梅隆大学的一个开源项目，之后从学校分离成为独立公司。该团队专注于大规模机器学习和图分析，帮助开发者在应用中加入机器学习和人工智能技术，具体使用场景包括推荐引擎、反欺诈、预测用户数变化、情绪分析以及用户分类等，该公司已经被苹果于 2016 年收购。

他的工作得到了多个会议和两个期刊的奖项，分别是：NIPS（Conference and Workshop on Neural Information Processing Systems）2003 和 2007，VLDB 2004，UAI（Conference on Uncertainty in Artificial Intelligence）2005，ICML（International Conference on Machine Learning）2005，IPSN（International Conference on Information Processing in Sensor Networks）2005 和 2006，KDD（Knowledge Discovery and Data Mining）2007 和 2010，JWRPM 2009，AISTATS（International Conference on Artificial Intelligence and Statistics）2010，JAIR（Journal of Artificial Intelligence Research）2007 和 2012。他还是 ONR 青年研究员奖、NSF（National Science Foundation 美国国家科学基金会）职业奖、Alfred P.Sloan 奖学金，IBM 教师奖学金，Siebel 奖学金和斯坦福百年教学助理奖的获得者。他还获得了 IJCAI（International Joint Conference on Artificial Intelligence）计算机和思想奖以及科学家和 PECASE（工程师总统早期职业奖）。

他还是 DARPA（Defense Advanced Research Projects Agency 美国国防高级研究计划局）信息科学与技术（ISAT）咨询小组的前成员。

- Christos Faloutsos



Christos Faloutsos，被誉为“数据库大师”。

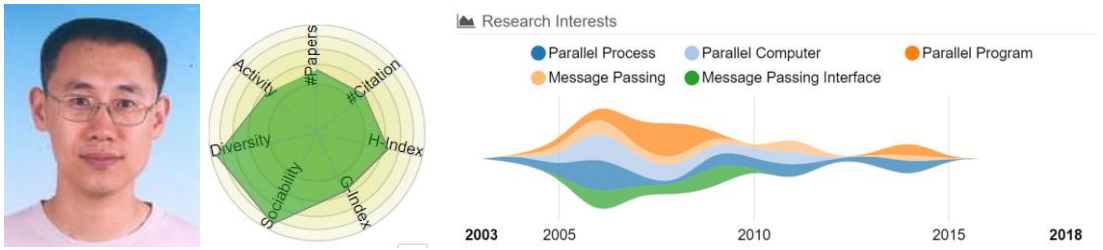
Christos Faloutsos 是希腊计算机科学家，拥有多伦多大学博士学位，任卡内基梅隆大学计算机科学系教授。他的研究兴趣包括流和网络的数据挖掘，分形，多媒体和生物信息学数据库的索引以及性能。

他于 1989 年获得了国家科学基金会颁发的总统青年研究员奖，22 个最佳论文奖和几个教学奖。1994 年以有关在时间序列数据库中快速子序列匹配的论文获得 SIGMOD 最佳论文奖。1997 年以他的 R+树论文获得 VLDB 十年论文奖，此外，他还获得了 ACM（Association for Computing Machinery）2010 SIGKDD 创新奖。

Christos Faloutsos 发表了 300 多篇评论文章，一篇专著，并拥有五项专利。他是 SIGKDD 国际会议执行委员会成员，并在 2010 年被任命为 ACM Fellow。

3.2.2 国内学者简介

- 陈文光



陈文光，CCF（中国计算机学会）高级会员，清华大学计算机系教授，主要研究领域为操作系统、程序设计语言与并行计算。

陈文光于 2000 年获得清华大学计算机系博士学位。2007 年-2015 年任清华大学计算机系副主任、现任清华大学计算机系学术委员会副主任，兼任青海大学计算机系主任。

他是 ACM 中国理事会副主席，ACM 中国操作系统分会 ChinaSys 主任，CACM（Central American Common Market 中美洲共同市场）中文版主办《Journal of Computer Science and

Technolog》计算机系统与体系结构 Leading Editor，《软件学报》责任编委，曾经参与编著《JAVA 虚拟机规范》以及《MPI 与 Open MP 并行程序设计：C 语言版世界著名计算机教材精选》等书。

他担任 ASPLOS（ACM International Conference on Architectural Support for Programming Languages and Operating Systems），PLDI（ACM SIGPLAN conference on Programming Language Design and Implementation），PPoPP，SC（Supercomputing Conference 全球超级计算大会），CGO（International Symposium on Code Generation and Optimization），IPDPS（International Parallel and Distributed Processing Symposium），APSYS（ACM Asia-Pacific Workshop on Systems）等领域内重要会议的程序委员会委员。

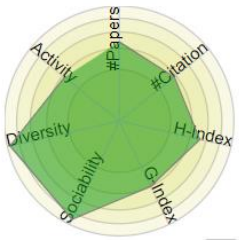
他在国内外学术期刊和学术会议上共发表论文多篇，是国家杰出青年基金获得者，获国家科技进步二等奖一次、部级科技一等奖两次，已授权中国发明专利 1 项，申请美国专利 1 项。他还是中国计算机学会杰出会员和杰出讲者、副秘书长，青年科技论坛荣誉委员。

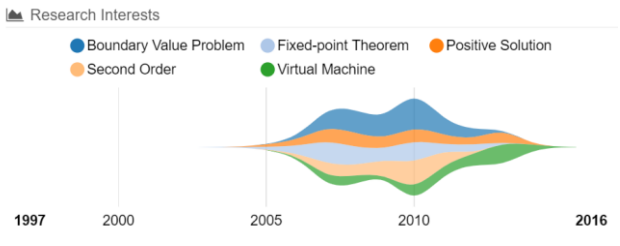
陈文光长期研究高性能计算编程模型和编译系统，近几年在以图计算系统为代表的新一代大数据处理系统方面取得了进展。

2014 年提出并实现了一种单机图处理引擎 GridGraph，通过一种基于源和目的节点双层混洗的图数据结构，能够高效利用外存放置图的边，从而实现在单机上处理十亿结点以上的图。GridGraph 性能比国际上同类单机图处理引擎如 X-STREAM 和 GraphChi 性能提高了一个数量级，论文在 USENIX ATC 15 上发表。

2016 年初，他进一步研制成功了名为“双子座”的分布式图计算系统，通过稀疏/稠密双模式计算引擎、稀疏性敏感的紧凑数据结构以及细粒度动态负载平衡等技术，在典型大数据分析应用（如 PageRank、ALS 等）上的性能是国际同类图计算系统 PowerGraph 和 PowerLyra 的十倍以上，是目前流行的大数据系统 Spark 性能的 100 倍以上，占用内存仅为其十分之一，其论文在 OSDI 16 上发表。

- 陈海波





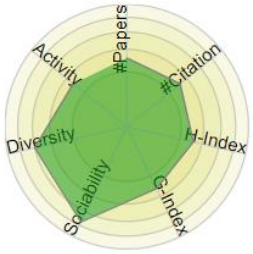
陈海波，现任上海交通大学软件学院教授，PowerLyra 共同作者，主要研究领域为系统软件，系统结构与系统虚拟化。

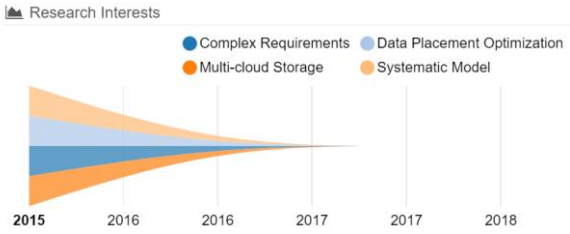
他于 2009 年获得复旦大学计算机系统结构专业博士学位。

陈海波曾在麻省理工学院参与众核操作系统研究，并曾在虚拟化公司 VMWare 加州总部从事可信虚拟化的研究。他曾担任国际学术会议 2011ACM 亚太系统会议（APSys 2011）的共同主席，并担任多个国际顶级学术会议如 2012 年 USENIX 技术年会（UsenixATC 2012），2013 年国际操作系统原理大会（SOSP 2013），2013 年欧洲系统大会（EuroSys 2013）等的程序委员会委员。其相关研究成果发表在诸多国际顶级会议如国际操作系统原理大会（SOSP），USENIX 操作系统设计与实现大会（OSDI），国际体系结构大会（ISCA），国际微处理结构大会（MICRO），国际软件工程大会（ICSE），国际并行处理原理与实践大会（PPoPP）与 IEEE Transactions on Software Engineering 等。

他于 2009 年被评为中国计算机学会上海青年 IT 新锐，2010 年获得 IBM X10 学院奖学金，他的博士论文《云计算平台可信性增强技术的研究》获得 2011 年全国优秀博士论文及 2009 年中国计算机学会优秀博士论文奖，他还于 2012 年获得 NetApp 学院奖学金。

- 武永卫





武永卫，清华大学计算机科学与技术系高性能计算研究所副所长，中国国家网格运行管理中心副主任，IEEE Cloud Computing 编委。

武永卫于 2002 年获得中科院系统所应用数学博士学位，研究方向为并行与分布式处理，云计算，存储系统，主要开展分布式计算与处理（如网格计算、云计算）等方面的研究工作，研究分布式资源的动态管理和调度机制、应用软件/系统的远程部署方法、基于网络的高性能计算用户使用模式、分布式文件系统和数据管理技术等。具体的研究方向包括运行时环境、数据存储与管理、虚拟化、资源管理调度、性能分析、分布式协同与一致性、可靠性与容错等。

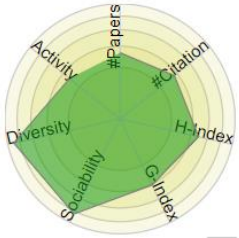
他提出了基于区间的高性能计算系统的负载预测方法，为改善高性能计算系统的运行效能提供支持。该方法系统分析和聚类各种高性能计算系统的运行 trace 特点，通过自适应

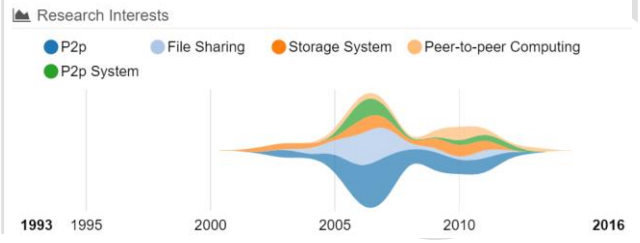
的负载趋势分析，提前更多时间进行预测，并将预测结果应用到作业调度，用于改善高性能计算系统的吞吐能力。他还提出了通过远程动态部署的方法改善和优化网格/云计算平台性能的方法，避免了热点应用引起的热点资源问题。该方法利用虚拟化技术，提供用户按需定制的物理/虚拟运行环境，并完成系统平台和应用软件的自动按需部署。

作为开发组组长，他研制完成的网格中间件系统 CGSP 支持了中国教育科研网格（ChinaGrid），其中的热部署技术被网格中间件系统 GlobusToolkits 采纳。他所在的研究组研制完成的清华大学云计算平台“TsinghuaCloud”被美国的 CloudBook 引用，其中的云存储平台 Corsair 于 2008 年 11 月开始在清华大学投入实际运行。

武永卫主要获奖经历有：2008 年获得国家科技进步二等奖；2009 年 ScalCom 最佳论文奖；2011 年 IMIS 最佳论文奖和中创软件人才奖；2014 年获得 FSE ACM SIGSOFT 杰出论文奖；2015 年获得国家技术发明二等奖：面向社区共享的高可用云存储系统；2015 年获得 Tsinghua Science and Technology 最佳论文奖。

- 代亚非





代亚非，现任北京大学信息科学技术学院教授，从事分布式系统方面的研究工作。

她于 1993 年从哈尔滨工业大学计算机系获得博士学位。

代亚非教授主要的科研领域包括计算机网络与分布式系统和 P2P 计算。她早期研究 P2P 共享及存储系统，后续开展了云计算，云存储，图计算等研究，所在课题组曾经开发了 P2P 存储系统 Amazingstore，实际运行多年，在高校内拥有一定规模的用户群。

她曾主持了包括 973 课题、863 课题、国家自然科学基金、国家科技重大专项课题、国家信息产业部等多项科研课题，在国内国际重要的期刊和会议发表相关论文百余篇。她所主持的北京大学“计算概论”课程被评为中国“国家级精品课程”，2011 年被评为北京市教学名师。她在 2014 年获得教育部科技进步二等奖，2015 年获得中国电子学会科技进步一等奖。

4 产业应用篇

2016 年 IBM 公司的全球技术展望（GTO）中提出，信息科技的发展已经从帮助人类完善自动化的工作向智能化迈进，而未来智能化认知计算发展的三大方向之一就是图计算以及基于图的认知技术，包括图分析、图特征挖掘、图认知推理、以及与机器学习技术的结合。利用图的强大的关联分析能力和对客观世界还原的优势，未来图计算技术将重点在分布式部署、大图分析、实时可视化等领域，而与行业应用的结合则是发挥图计算能力的最优选择

图就在我们身边，它们遍布各地，如 Facebook、谷歌、Twitter、电信、生物、医药、营销和股票市场。同时，它们影响计算机科学的众多领域，包括软件工程、数据库和集成电路的设计。

目前，图计算已应用于医疗、金融、社交分析、自然科学以及交通等领域，很多互联网公司以及很多年轻的人工智能领域创业公司也都开展了图计算相关的业务。以下我们就图计算的主要应用场景与公司进行简介。

4.1 医疗行业的应用

图计算的出现使得对病人的智能诊断成为可能。对病人开具处方需要依据病人的病情特征与以往的健康情况，以及药物的相关情况。过去的医疗大多依赖于医生的个人经验与病人的自我描述，传统的数据处理系统无法一次性调出多个与病人情况、保险情况、药物情况相关的数据库——挑战在于信息必须由多个在线资源拼凑而成，包括列出疾病和治疗的电子病历、医疗保险或其他跟踪医疗服务的数据库、描述药物的数据库，在某些情况下，还有跟踪临床试验的独立数据库。该场景是经典的链接网络，每个节点之间具有相互依赖性。变量可包括患者年龄和性别、特定药物（或药物组合）的结果、特定剂量，给药时的疾病阶段和潜在的药物相互作用。传统的 SQL 数据库实际上不可能计算这样的问题，因为传统的纯软件图无法提供应用所需的深度嵌套的连接，而图分析系统的出现则使得这样的场景成为了可能。

4.2 金融行业的应用

在金融实体模型中，存在着许许多多不同类型的关系，以及数十亿的结点和边。有些是相对静态的，如企业之间的股权关系、个人客户之间的亲属关系，有些则是不断地在动态变化，如转账关系、贸易关系等等。这些静态或者动态的关系背后，隐藏着很多以前我们不知道的信息。之前，我们在对某个金融业务场景进行数据分析和挖掘过程中，通常都是从个体（如企业、个人、账户等）本身的角度出发，去分析个体与个体之间的差异和不同，很少从个体之间的关联关系角度去分析，因此会忽略很多原本的客观存在，也就更无法准确达到该

业务场景的数据分析和挖掘目标。而图计算和基于图的认知分析正是在这方面弥补了传统分析技术的不足，帮助我们从金融的本质角度来看这个问题，从实体和实体之间的经济行为关系出发来分析问题。

在金融行业中，图计算以及认知技术重点应用的业务领域包括：金融风险的管控、客户的营销拓展，内部的审计监管、以及投资理财等方面。例如，银行面临的洗钱犯罪风险，目前很多情况下，银行对洗钱伎俩和手段的调查和分析还依赖调查员的手工方式来做，不仅效率低下，工作量大，而且极易造成漏查的问题。目前洗钱的手段和步骤错综复杂，整个过程大致可分为以下三个方面，并且交互交叉重叠，反复出现。

（1）存放（Placement）：将犯罪得益放进金融体系内。

（2）掩藏（Layering）：将犯罪得益转换成另一种形式，例如从现金换成支票、贵重金属、股票、保险储蓄、物业等。

（3）整合（Integration）：经过不同的掩饰后，将清洗后的财产如合法财产，融入经济体系。

面对这样的复杂困难问题，目前金融机构采取的手段都是基于预先设定的规则来分析每笔交易的背后是否存在洗钱可能，将这些高可能性的交易区分为交易组，交由调查员来进行审核调查，但是最后的结果通常都发现，利用这种预先定义的规则来识别出来的警告信息，误报率很高，有的达到80%以上，给调查员带来很大无必要的工作量。所以，如何提高高风险交易识别的准确率是在反洗钱领域急需解决的问题。而利用图计算和图认知技术，从交易本身出发，探查交易方的交易历史，跟踪交易的轨迹，追溯资金的流向，找出规则方法无法覆盖的新的洗钱模式，及时地更新现有的探查规则，从而大幅度降低误报率。

再如国内商业银行都面临的信贷风险问题：受国内经济下行的影响，企业客户贷款的不良率攀升。为了提高银行对企业不良风险传导的预测能力，利用图计算和图认知技术，完整刻画企业客户之间、企业与自然人之间的社会关系、经济往来关系，构建全方位的风险关联网络，实现风险要素的动态性和完整性呈现。当关联网络内某家企业发生信贷风险时，利用风险关联网络中风险客户的客户画像，经济行为轨迹等信息进行交叉关联分析，预测风险的传导路径和扩散范围，帮助银行采取有效措施，阻断风险传染源，进行风险隔离，从而提升风险管理的可靠性和准确率[3]。

4.3 互联网行业的应用

目前大数据在互联网公司主要应用在广告、报表、推荐系统等业务上。在广告业务方面需要大数据做应用分析、效果分析、定向优化等，在推荐系统方面则需要大数据优化相关排

名、个性化推荐以及热点点击分析等。图计算的出现满足了这些计算量大、效率要求高的应用场景的需求。

图计算模型在大数据公司，尤其是 IT 公司是非常流行的一大模型，它是很多实际问题最直接的解决方法。近几年，随着数据的多样化，数据量的大幅度提升和算力的突破性进展，超大规模图计算在大数据公司发挥着越来越重要的作用，尤其是以深度学习和图计算结合的大规模图表征为代表的系列算法。

图计算的发展和应用有井喷之势，各大公司也相应推出图计算平台，例如 Google Pregel、Facebook Graph、腾讯星图、华为图引擎服务 GES 等。接下来，本节将简要介绍图计算系统在互联网行业的应用。

- 华为——GES

华为图引擎服务 GES（Graph Engine Service）是针对以“关系”为基础的“图”结构数据，进行查询、分析的服务。广泛应用于社交关系分析、推荐、精准营销、舆情及社会化聆听、信息传播、防欺诈等具有丰富关系数据的场景。

其主要应用场景如下：

图 14 全球图计算领域活跃学者性别比

- 百度——Hugegraph

百度安全开源的图数据库 HugeGraph，是百度安全团队基于安全特定场景和实际运营中的业务需求衍生出的一款面向分析型、支持批量操作的图数据库系统。它能够与大数据平台无缝集成，解决海量图数据的存储、查询和关联分析需求。它可以存储海量的顶点（Vertex）和边（Edge），实现 Apache TinkerPop 3 框架，支持 Gremlin 查询语言。

- 腾讯——星图

腾讯星图（Star Knowledge Graph，即 SKG，也称知识图谱），是一个图数据库和图计算引擎的一体化平台：融合治理异构异质数据；提供关联查询、可视化图分析、图挖掘、机器学习和规则引擎；支持万亿关联关系数据的快速检索、查找和浏览；挖掘隐藏关系并模型化业务经验。其应用场景如图 14 所示。

图 15 腾讯星图应用场景

- 阿里云——关系网络分析

阿里关系网络分析软件是基于大数据时空关系网络的可视化分析平台，产品提供关联网络（分析）、时空网络（地图）、搜索网络、动态建模等功能，在阿里巴巴、蚂蚁金服就按内广泛应用于反欺诈、反作弊、反洗钱等风控业务。

- 费马科技

费马科技由清华大学计算机系教授及博士创办，是从事图数据分析和存储、提供图计算解决方案的高新技术企业。其主要产品有图计算平台、图数据库和图计算解决方案。公司致力于解决图数据的存储和分析难题，同时提供咨询服务，积极推进图计算在各行业的应用，其产品应用之一为金融行业风控。费马科技自主研发的 LightningGraph 是具有完全自主知识产权的图计算平台，用于大规模复杂关系网络的存储、查询以及计算分析。

- 亚马逊——Neptune

Amazon Neptune 是一项完全托管的图数据库服务，可构建和运行适用于高度互连数据集的应用程序。Amazon Neptune 的核心是专门构建的高性能图数据库引擎，它进行了优化以存储数十亿个关系并将图查询延迟降低到毫秒级。Amazon Neptune 支持常见的图模型 Property Graph 和 W3C RDF 及其关联的查询语言 Apache TinkerPop Gremlin 和 SPARQL。Neptune 支持图使用案例，如建议引擎、欺诈检测、知识图谱、药物开发和网络安全。

- TigerGraph

TigerGraph 总部位于纽约，在上海设有办公室。该公司提供企业级的实时图数据库平台，发挥图数据结构天然的优势，用来探索、发现和预测数据背后深度（3 层以上）的关联关系。需要这些重要特性的企业应用包括：风险控制，反欺诈，供应链物流优化，企业控制图谱，个性化推荐等等。TigerGraph 的 NPG（Native Parallel Graph）设计结合了原生图的存储和计算，支持实时图更新，并提供内置的并行计算。此外，TigerGraph 的架构是模块化的，并同时支持分布式多机扩展和单机多核扩展的应用部署模型。

5 趋势篇

随着大数据、云计算行业的发展，图计算领域也愈加受到关注，图计算技术的应用也愈加广泛。为了解该领域发展趋势，AMiner 基于图计算历史的科研成果数据，对其技术来源、热度进行了研究。

5.1 全局热度

我们对图计算全局热度做了统计，如图 16 所示，每个彩色分支表示一个关键词领域，其宽度表示该关键词的研究热度，各关键词在每一年份（纵轴）的位置是按照这一时间点上所有关键词的热度高低进行排序。

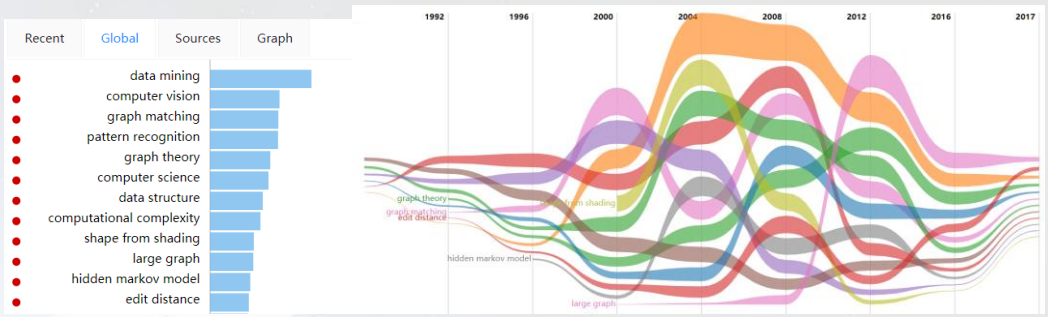


图 16 腾讯星图应用场景

从全局的角度来看，自 1992 年至今，data mining、computer vision、graph matching、pattern recognition 等一直都是研究人员研究的重点。

5.2 近期热度

相较于全局热点，large graph、social network、graph theory、data mining、edit distance 等则是近期的研究重点，具体热度趋势如下图所示：

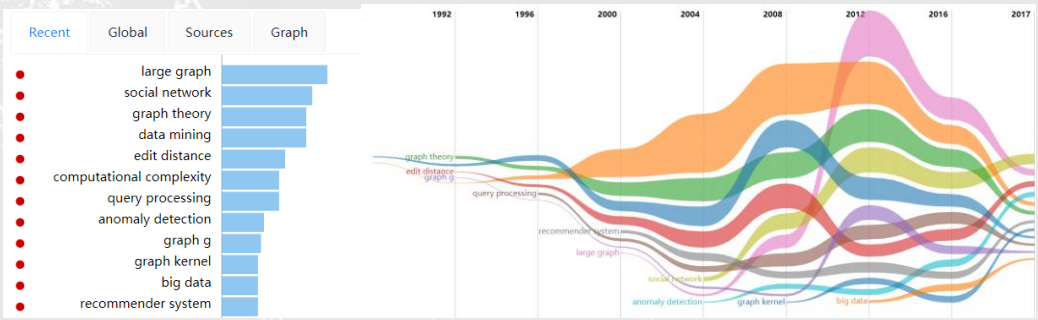


图 17 graph computing 近期热度

5.3 交叉研究分析

根据全局热点与近期热点的趋势图，我们选取 graph computing & data mining、graph computing & machine learning、graph computing & social network 等进行笛卡尔交叉分析。通过对两个领域的知识图谱的计算，再对两领域的细分子领域进行笛卡尔乘积热点挖掘，我们挖掘了历史数据分析和未来趋势预测两部分。本文主要探讨 2007 年至今的研究状况；趋势预测仅以未来 3 年为周期来探讨。

领域交叉热力值由交叉研究的论文的 citation 等数据加权计算得出，热力值越高，表明这两个交叉子领域交叉研究的越深入和广泛。

每个交叉热点中的研究学者，发表论文，中外学者和论文对比等数据均可以获得。用作展示时，研究学者和论文分别按照交叉领域研究影响度和论文相关度作为默认排序。

学者研究影响度由交叉领域内论文量，h-index 等计算得出；论文相关度由交叉领域内论文的关联程度和引用数量等计算得出。

对比分析中“中外研究人员对比”和“中外研究论文对比”是专家数量和论文数量的直接对比；而“中外论文影响对比”是论文引用量（citation）的对比。

5.3.1 Graph Computing & Data Mining

我们选取 graph computing 的 11 个相关领域作为研究对象，具体包括：

1. graph computing
2. data structures
3. graph mining
4. database
5. graph theory
6. algorithms
7. distributed computing
8. system modeling
9. computational geometry
10. numerical analysis
11. text analysis

我们选取 data mining 领域近期热度、全局热度最高、相关性最强的 10 个相关领域作为研究对象，具体包括：

1. data mining
2. clustering
3. text mining
4. classification
5. taxonomy
6. time series analysis
7. association rule
8. big data
9. data management
10. network analysis

对两个领域的细分子领域进行笛卡尔乘积热点挖掘，得出历史交叉热点图如下所示：

图 18 2007 至今 graph computing 与 data mining 领域交叉分析

2007 年至今，全球共有 492166 位专家投入了 graph computing 和 data mining 领域的交叉研究中，其中华人专家 169220 人，约占 34.38%，共产生交叉研究论 223063 篇。学者 h-index 和 citation 分布如下：

表 4　2007 至今 graph computing 与 data mining 领域交叉分析

h-index	专家人数	分布占比
小于 10	469379	95.37%
10~20	15822	3.21%
20~40	5253	1.07%
大于 40	1128	0.23%
总计	492166	100%

表 5 2007 年至今 graph computing 与 data mining 交叉研究论文 citation 分布

citation	专家人数	分布占比
小于 10	99119	44.44%
1~10	76533	34.31%
10~100	42654	19.12%
100~200	2946	1.32%
大于 200	1811	0.81%
总计	223063	100%

5.3.2 Graph Computing & Machine Learning

我们选取 graph computing 域近期热度，全局热度最高，相关性最强的 11 个相关领域作为研究对象，具体包括：

1. graph computing
2. data structures
3. supervised learning
4. database
5. algorithms
6. graph theory
7. network analysis
8. distributed computing
9. system modeling
10. numerical analysis
11. text analysis

我们选取 machine learning 域近期热度，全局热度最高，相关性最强的 10 个相关领域作为研究对象，具体包括：

1. machine learning
2. neural networks
3. unsupervised learning
4. cluster analysis
5. regularization
6. anomaly detection
7. reinforcement learning
8. dynamic programming
9. support vector machines
10. object recognition

对两个领域的细分子领域进行笛卡尔乘积热点挖掘，得出历史交叉热点图如下所示：

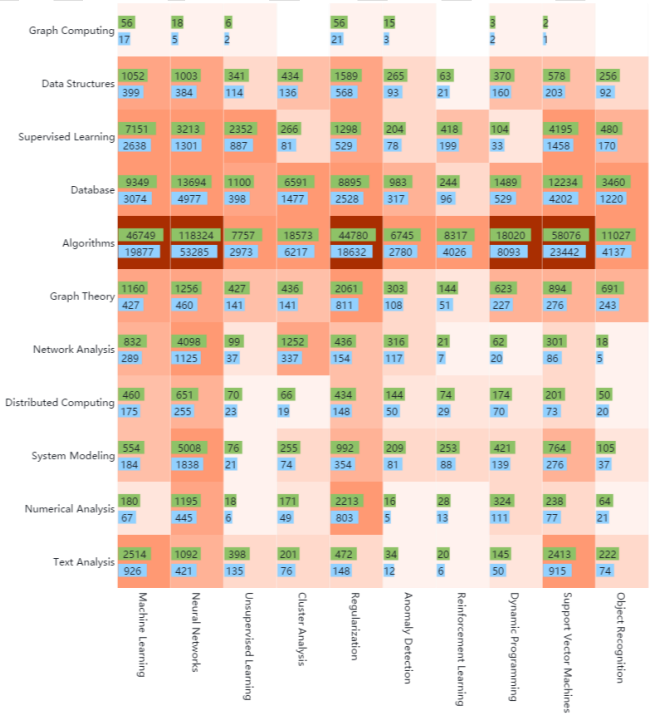


图 19 2007 至今 graph computing 与 machine learning 领域交叉分析

2007 年至今，全球共有 331387 位专家投入了 graph computing 和 machine learning 领域的交叉研究中，其中华人专家 126793 人，约占 38.26%，共产生交叉研究论 151511 篇。学者 h-index 分布和 citation 分布如下：

表 6 2007 年至今 graph computing 与 machine learning 领域交叉研究学者 h-index 分布

h-index	专家人数	分布占比
小于 10	310161	93.59%
10~20	13330	4.02%
20~40	5662	1.71%
大于 40	1344	0.41%
总计	331387	100%

表 7 2007 年至今 graph computing 与 machine learning 领域交叉研究论文 citation 分布

Citation	专家人数	分布占比
小于 10	65905	43.50%
1~10	53162	35.09%
10~100	28935	19.10%
100~200	2164	1.43%
大于 200	1345	0.89%
总计	151511	100%

5.4 技术预见

AMiner 根据图计算领域近十年的相关论文可以从图计算的技术层面进行分析，更直观的展现出图计算相关的关系图和发展趋势图，旨在基于历史的科研成果数据的基础上，对图计算技术发展趋势进行研究。

选取的热门关键词分别为：combinatorial algorithms、association scheme、huffman coding、empirical entropy、suffix tree、hypergraphs、sub-dominant、hypercube、dissimilarity、paths and connectivity problems、routing、np-completeness、computational complexity、shortest path、graph enumeration。

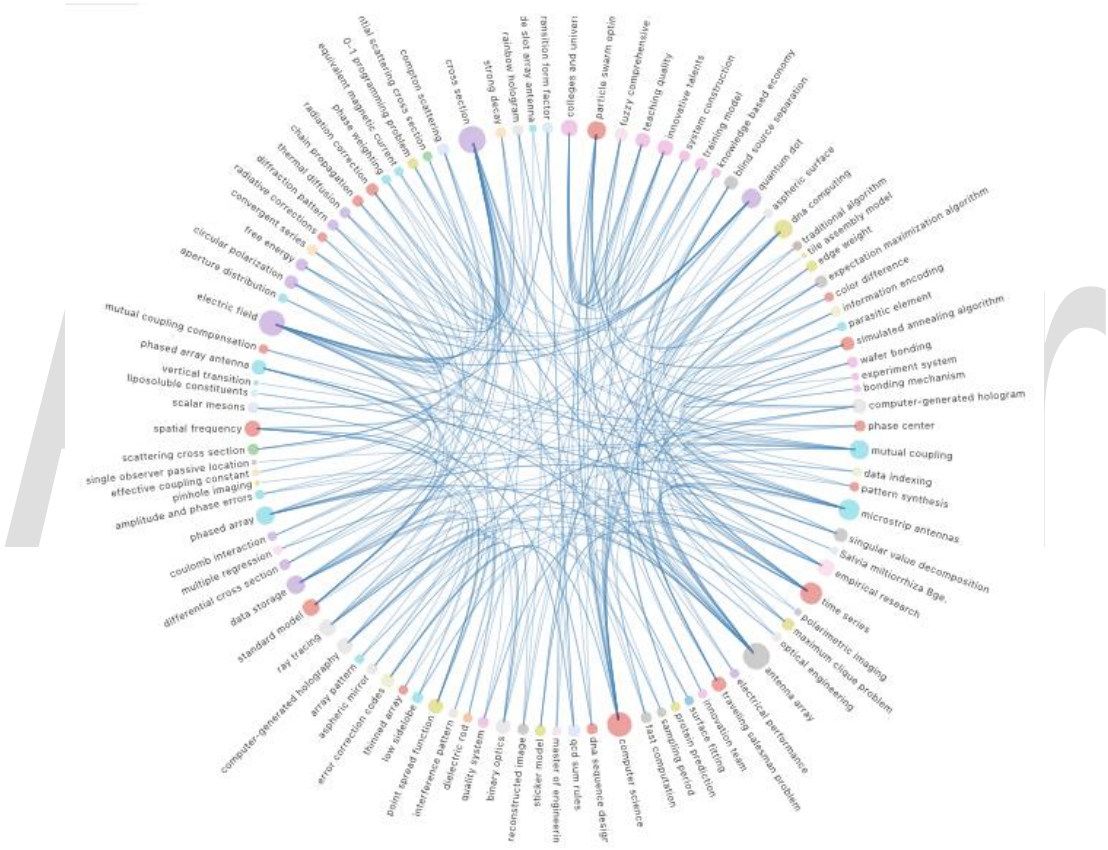


图 20 图计算技术预见图

参考文献

[1] Mcsherry F , Isard M , Murray D G . Scalability! But at what COST?[C]// Usenix Conference on Hot Topics in Operating Systems. USENIX Association, 2015.

[2] 朱晓伟，陈文光.Gemini：以计算为中心的分布式图计算系统[J].中国计算机协会通讯，2017,13（8）：16-21.

[3] Safiollah Heidari，Yogesh Simmahan，Rodrigon.Calheiros，Rajkumar Buyya.Scalable Graph Processing Frameworks: A Taxonomy and Open Challenges[J]. ACM Computing Surveys，2018-6，51（3）：1-53.

[4] Chengwen Wu，Guangyan Zhang，Keqin Li，Weimin Zheng.Large graph computing systems.大数据，2015.

[5] Da Yan，Yingyi Bu，Yuanyuan Tian，Amol Deshpande.Big Graph Analytics Platforms[J].now publishers，2017.

[6] Xuanhua Shi，Zhigao Zheng，Hai jin，Ligang He，Bo Liu，Qiang-Sheng Hua.Graph Processing on GPUs: A Survey[J].ACM Computing Surveys，2017.

[7] 刘梦雅，刘燕兵，于静，郭莉，孙志刚.图数据分析系统计算模型综述[J].计算机应用研究，2017，34（11）：3205-3206.

[8] 王雪.图计算及其认知技术在金融行业的应用[EB/OL].2017.

[9] Sabeur Aridhi, Engelbert Mephu Nguifo.Big Graph Mining: Frameworks and Techniques[J].Big Data Research，2016.

[10] 丁悦，张阳，李战怀，王勇.图数据挖掘技术的研究与进展[J].计算机应用，2012，32（01）：182-190.

[11] 张素智，张琳，曲旭凯.图数据挖掘技术的现状与挑战[J].现代计算机，2015.

[12] Safiollah Heidari, Yogesh Simmhan, Rodrigo N Calheiros, Rajkumar Buyya. Scalable Graph Processing Frameworks: A Taxonomy and Open Challenges[J]. ACM Computing Surveys, 2018.

[13] 吴城文,张广艳,郑纬民.从系统角度审视大图计算[J].大数据,2015,1(03):48-61.

[14] 刘梦雅,刘燕兵,于静,郭莉,孙志刚.图数据分析系统计算模型综述[J].计算机应用研究,2017,34(11):3204-3213.

[15] Chengwen Wu, Guangyan Zhang, Keqin Li, and Weimin Zheng. Large Graph Computing Systems[J]. Big Data Management and Processing, 2017.

附录

Graph Computing 知识图谱（共包含二级节点 15 个，三级节点 93 个）：

领域	二级分类	三级分类
图计算 graph computing	data structures 数据结构	data type 数据类型
		abstract data type 抽象数据类型
		graph representation 图表示
	formal languages 形式化语言	graph grammar 图文法
		formal specification 形式规约
		canonical form 规范形式
	algorithms 算法	algorithm design and analysis 算法设计与分析
		np complete problem np完全问题
		computational modeling 计算模型
		polynomial time 多项式时间
		computational complexity 计算复杂度
		linear time algorithm 线性时间算法
		upper bound 上界
		new algorithm 新型算法
		lower bound 下界
		time complexity 时间复杂度
		efficient algorithm 高效算法
		approximation algorithms 近似算法
		randomized algorithm 随机算法
		string matching 字符串匹配
		linear programming 线性规划
		combinatorial optimization 组合优化
	graph embedding 图嵌入	vector space 向量空间
		feature vector特征向量
		random walk 随机游走
		feature extraction 特征提取
		feature selection 特征选择
	graph mining 图挖掘	graph homomorphism 图同态
		graph isomorphism 图同构
		subgraph isomorphism 子图同构
		graph matching 图匹配
		pattern matching 模式匹配，
		pattern recognition 模式识别
		structural pattern recognition 结构化模式识别
	database 数据库	graph database 图数据库
		relational databases 关系型数据库
		query processing 查询处理
		xml database xml数据库
		materialized views 物化视图
		query optimization 查询优化
	natural computing 自然计算	dna computing dna计算
		membrane computing膜计算
		genetic algorithm 遗传算法
		ant colony optimization 蚁群算法优化

graph theory 图论	planar graph 平面图
	random graph 随机图
	median graph 中值图
	directed graph 有向图
	bipartite graph 二分图
	connected graph 连通图
	topological index 拓扑指数
	feedback vertex set 反馈点集
	laplacian matrix 拉普拉斯矩阵
	complete graph 完全图
	dominating set 支配集
	bounded-genus graph 有界的图亏格
	graph minor 图子式
	claw free graph 无爪图
	spanning tree 最小生成树
	shortest path 最短路径
	graph transformation 图转换
	graph transformation system 图转换系统
	graph rewriting 图重写
network analysis 网络分析	sensor network 传感器网络
	communication network 通信网络
	mobile robot 移动机器人
	mobile agents 移动设备
	wireless sensor network 无线传感器网络
	petri net 佩特里网
	stochastic petri net 随机佩特里网络
	pagerank 网页排名
	markov processes 马尔可夫过程
	markov chain 马尔科夫链
	hidden markov model隐马尔科夫模型
	iteration method 迭代式方法
distributed computing 分布式计算	graph cut 图切割
	fault tolerant 容错
	local computation 局部计算
	message passing 消息传递
	graph partitioning 图划分
	distributed system 分布式系统
	distributed algorithm分布式算法
	large graph 大图
machine learning 机器学习	support vector machines 支持向量机
	decision tree 决策树
	clustering algorithms 聚类算法
	cost function 损失函数
	energy minimization 能量最小化
	gaussian mixture model 高斯混合模型
	search space 搜索空间
	language model 语言模型
system modeling 系统建模	model transformation 模型变换
	linear system 线性系统

	computational geometry 计算几何	
	numerical analysis 数值分析	
	text analysis 文本分析	

版权声明

AMiner 研究报告版权为 AMiner 团队独家所有，拥有唯一著作权。AMiner 咨询产品是 AMiner 团队的研究与统计成果，其性质是供用户内部参考的资料。

AMiner 研究报告提供给订阅用户使用，仅限于用户内部使用。未获得 AMiner 团队授权，任何人和单位不得以任何方式在任何媒体上（包括互联网）公开发布、复制，且不得以任何方式将研究报告的内容提供给其他单位或个人使用。如引用、刊发，需注明出处为“AMiner.org”，且不得对本报告进行有悖原意的删节与修改。

AMiner研究报告是基于AMiner团队及其研究员认可的研究资料，所有资料源自AMiner后台程序对大数据的自动分析得到，本研究报告仅作为参考，AMiner 团队不保证所分析得到的准确性和完整性，也不承担任何投资者因使用本产品与服务而产生的任何责任。

顾　　问：陈文光　唐　杰

主　　编：刘　佳

编　　辑：张　蕊　何　杨　唐丽杭　戴雨洁

封面设计：李　娜