Accelerating PageRank using Partition-Centric Processing

文献综述

学院：武汉光电国家研究中心　班级：硕 1807　学号：M201872990 姓名：熊美珍

1. 前言

PageRank 是一种基本的链接分析算法，也是稀疏矩阵向量(SpMV)乘法性能的关键代表。传统的 PageRank 实现产生精细的随机存储器访问，导致大量浪费的 DRAM 流量和较差的带宽利用率。这篇论文提出了一种新的以分区为中心的处理方法(PCPM)来计算 PageRank，该方法在实现高持续存储带宽的同时，大大减少了 DRAM 通信量。PCPM 使用一个以分区为中心的抽象概念和收集-应用-分散(GAS)编程模型。通过仔细研究基于 PCPM 的实现如何影响算法的通信特性，我们提出了几个系统优化，大大提高了执行时间。与现有技术相比，使用此种技术的平均执行时间缩短了 2.7 倍，通信量减少了 1.7 倍。我们还表明，与其他基于 GAS 的实现不同，PCPM 能够利用增强局部性的智能节点标记进一步减少主内存流量。虽然我们在本文中使用 PageRank 作为目标应用，但是我们的方法可以应用于通用 SpMV 计算。

2. 背景

在许多领域，如网络和社交网络分析[8，2，20，9]，生物学[13]，交通[12，3]等，图表是数据表示的首选。这些领域问题的规模不断扩大，引发了对高性能图形分析的大量研究兴趣。这项研究的大部分集中在共享内存平台上，因为与分布式系统[19]相比，共享内存平台的通信开销较低。现代系统中的高 DRAM 容量还允许在单个服务器[26、24、28]上对大图形进行内存处理。然而，由于(1)低计算通信比和(2)图形算法的不规则内存访问模式，即使在单个节点上，计算能力的有效利用也是一个挑战。CPU 速度和 DRAM 带宽之间的差距越来越大，被称为“记忆墙[31]，已经成为高性能图形分析的一个关键问题。

2.1 传统的 PageRank 算法（PDPR）

PageRank 通常在拉动方向[26、29、28、21]上计算，其中每个顶点拉动其内邻居的值，并累加成其自身的值。这相当于按列主顺序遍历，并计算的点积 每一列都带有缩放的 PageRank 矢量 SPR。

在拉方向实现中，每列完全拥有输出向量中相应元素的计算。这使得可以并行异步遍历的所有列，而无需在内存中存储部分和。相反，在推送方向上，每个节点通过向外邻居添加自己的值来更新它们。这需要对 and 存储进行行主遍历，以获得部分和，因为每一行对输出向量中的多个元素都有部分贡献。此外，需要同步来确保更新同一输出元素的多行的无冲突处理。

性能挑战:稀疏矩阵布局如压缩稀疏列(CSC)，将列的所有非零元素顺序存储在内存中，允许快速遍历[27]。然而，节点的邻居可以分散在图中的任何地方，读取它们的值会导致随机访问(单个或双字)到→拉方向计算中的弹簧。类似地，推送方向实现使用压缩稀疏行(CSR)格式对进行快速的行主遍

历，但是受到对部分和向量的随机访问的影响。这些低局部性和细粒度访问导致高高速缓存未命中率，并导致大部分内存流量。

2.2 相关工作

PageRank 的性能在很大程度上取决于图的内存访问模式中的局部性(我们称之为图局部性)。由于节点标记对图的局部性有重大影响，许多先前的工作已经研究了使用节点重新排序或聚类[6，16，5，1]来提高图算法的性能。基于邻居的空间和节奏位置感知放置的重新排序已经显示出进一步优于众所周知的聚类和基于树的技术。然而，这种简单的算法也引入了大量的预处理开销，这限制了它们的实用性。此外，像社交网络这样的无标度图形由于其偏斜的程度分布，通过重新排序变换不太容易处理。

高速缓存阻塞(CB)是用于加速图形处理[30、23、34]的另一种技术。CB 通过限制随机访问节点的范围来诱导局部性，并且已经被证明可以减少缓存未命中[18]。CB 将 A 沿着行、列或两者分割成多个块矩阵。然而，使用 CB 进行 SpMV 计算需要为每个块重新读取部分和。这些块矩阵极其稀疏的特性也减少了缓存顶点数据[22]的重用。

聚集-应用-分散(GAS)是另一个流行的模型，它被纳入了许多图形分析框架[17，25，11]。它将分析计算分为散射和聚集阶段。在散射阶段，源顶点在其所有输出边上传输更新，在聚集阶段，这些更新被处理以计算相应目的顶点的新值。

分箱利用了两阶段计算模型，以半分类的方式存储更新。这导致了算法访问模式的时空局部性。分箱可以与顶点中心范式或边缘中心范式结合使用。Zhou 等人[32，33]使用带有以边缘为中心的处理的定制排序边缘列表来减少 DRAM 行激活并提高内存性能。然而，他们的分类机制引入了一个不平凡的预处理成本，并强制使用 COO 格式。这导致了比基于 CSR 的顶点中心实现[4，10]更大的通信量和执行时间。

当与以顶点为中心或以边缘为中心的抽象一起使用时，GAS 模型本身也是次优的。这是因为它在每次迭代中遍历整个图形两次。尽管如此，以顶点为中心的 GAS 分箱(BVGAS)是共享内存平台[4，10]上最先进的方法。

3. 本文现状

3.1 以分区为中心的处理方法

本文提出了一种新的以分区为中心的处理方法——ology (PCPM)，它大大提高了当前以顶点为中心或以边缘为中心的方法所能实现的处理器-内存通信的效率。我们将分区定义为连续标记节点的不相交集合。然后，以分区为中心的抽象将图视为从每个节点到对应于该节点邻居的分区的一组链接。我们将这个抽象与两阶段聚集-应用-散射(GAS)模型结合使用。

在 PCPM 分散阶段，每个线程一次处理一个分区。处理分区 p 意味着将消息从 p 中的节点传播到相邻分区。给分区 p 的消息包括源节点的更新值(PR[v])和位于 p 中的 v 的外邻居列表。PCPM 缓存 p 的顶点数据，并将消息流至主存储器。以分区为中心的方式生成来自 p 的消息，即从 p 中的所有节点到相邻分区 p 的消息是连续生成的，并且不与到任何其他分区的消息交织。

在收集阶段，每个线程一次扫描指定给一个分区 p 的所有消息。消息扫描将更新值应用于该消息的邻居列表中的所有节点。p 中节点的部分和被缓存，消息从主存储器中流出。扫描完所有发送给 p 的消息后，部分总和(新的 PageRank 值)被写回 DRAM。

通过为每个分区静态预分配不同的存储空间来写消息，PCPM 可以并行分散或收集多个分区。

3.2 优化

（1）以分区为中心的更新传播

PCPM 的独特抽象自然会导致将单个更新从一个节点传输到相邻分区。换句话说，即使一个节点在一个分区中有多个邻居，在分散阶段，它也只在相应的更新箱中插入一个更新值。

PCPM 操纵目的节点 IDs 的最高有效位(MSB)来指示分区中使用相同更新值的节点范围。在 destID 箱中，它连续写入同一源顶点附近所有节点的 ID，并将该范围内第一个 ID 的 MSB 设置为 1 以进行划界 。由于 MSB 是为这种功能性保留的，PCPM 支持具有多达 20 亿个节点的图表，而不是 4 字节节点 IDs 的 40 亿个。然而，据我们所知，这足以处理大部分公开可用的大型数据集。

（2）PNG 数据布局

二分分区节点图(PNG)数据布局展示了 PCPM 真正的以分区为中心的特性。在分散阶段，PNG 防止未使用的边缘读取，并确保在切换到另一个 bin 之前，对 bin 的所有更新都是一起流的。一旦 destID bins 被写入，PCPM 中唯一需要的信息就是节点和分区之间的连接性。因此，在一个分区中从一个源节点到所有目的节点的边可以被压缩成一个边，其新的划分是相应的分区号。首先扫描分区中所有节点的输出边，并在丢弃的同时单独计算所有目的分区的未使用的边。计算 CSR 矩阵的偏移量数组时，会使用这些度的前缀和。同样的偏移量也可以用于将不相交的写入位置分配到目的分区的容器中。在下一次扫描中，CSR 中的边缘阵列填充有源节点 IDs，完成压缩和置换。PNG 构造可以很容易地在所有分区上并行化，以有效地加速预处理。

（3）避免分支

避免分支会增强持续的内存带宽，但不会影响 DRAM 通信量。为了实现分支避免聚集功能本文不使用 MSB (id)上的条件检查，而是直接添加它来更新 ptr。当 MSB (id)为 0 时，指针不递增，并且在下一次迭代中从缓存中读取相同的更新值；当 MSB (id)为 1 时，指针递增，对更新箱[p]执行弹出操作。

3.3 推广

PCPM 可以通过将边缘权重和目标 IDs 一起存储在 destID 箱中，轻松扩展到加权图上进行计算。这些权重可以在收集阶段读取，并在更新目标节点之前应用于源节点值。PCPM 也可以通过将行和列分开来扩展到具有非正方形矩阵的通用 SpMV。

4. 分析评估

4.1 DRAM 通信模型对比

传统的 PDPR 的性能在很大程度上取决于 CMR。在最坏的情况下，所有访问都是高速缓存未命中，在最好的情况下，只遇到冷未命中来加载高速缓存中的 PageRank 值。另一方面，BVGAS 的通信保持不变。随着 θ(m)额外的负载和存储，BV GAScomm 永远不会达到 PDPRcomm 的下限。相比之下，当对于每个顶点，所有输出边都可以被压缩成单个边， PCPMcomm 达到最佳状态。在最坏的情况下，PCPM 仍然和 BVGAS 一样好。与 BVGAS 不同，PCPMcomm 实现了与 PDPRcomm 相同的下限。

4.2 随机访问模型对比

BVGAS 显示出比 PDPR 更少的随机访问。然而，PCPMra 远小于 BV GASra 和 PDPRra。

5. 总结

本文提出了一种以分区为中心的处理方法(PCPM)，该方法将图形视为节点和分区之间的一组链接，而不是节点和它们各自的邻居。文章介绍了这种抽象的几个特点，并开发了数据布局和系统级优化来利用它们。

本文对此方法进行了广泛的分析和实验评估。使用简单的基于索引的分区，观察到与现有技术相比，平均执行时间提高了 2.7 倍，DRAM 通信量减少了 1.7 倍。未来，基于此种模型，可探索边缘分区模型[15，7]，以进一步减少通信量并改善 PCPM 的负载平衡。

此方法在 PageRank 上展示了 PCPM 的优势，也展示了它可以很容易地扩展到通用 SpMV 计算。所以可以相信 PCPM 可以成为其他图形算法或图形分析框架的有效编程模型。

在这篇文章中，有许多可以进一步探索的方向。例如，支持 PNG 的 PCPM 的流存储器访问模式非常适合高带宽存储器(HBM)和基于磁盘的系统。探索 PCPM 作为异质存储器或处理器架构的编程模型是未来工作的一个有趣途径。PCPM 一次只能从一个图形分区访问节点。因此，G - Store 表示的最小位数可以用来进一步减少内存足迹和 DRAM 通信。设计新的增强压缩方法也可以使 PCPM 适合用于商品 PC 上的大规模图形处理。

6. 参考文献

[1] ABOU-RJEILI, A., AND KARYPIS, G. Multilevel algorithms for partitioning power-law graphs. In Proceedings of the 20th International Conference on Parallel and Distributed Processing (2006), IPDPS'06, IEEE Computer Society, pp. 124–124.

[2] ALBERT, R., JEONG, H., AND BARAB´ASI, A.-L. Internet: Diameter of the world-wide web. Nature 401, 6749 (1999), 130.

[3] ALDOUS, J. M., AND WILSON, R. J. Graphs and applications: an introductory approach, vol. 1. Springer Science & Business Media, 2003.

[4] BEAMER, S., ASANOVI´C, K., AND PATTERSON, D. Reducing pagerank communication via propagation blocking. In Parallel and Distributed Processing Symposium (IPDPS), 2017 IEEE International (2017), IEEE, pp. 820–831.

[5] BOLDI, P., ROSA, M., SANTINI, M., AND VIGNA, S. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In Proceedings of the 20th international conference on World wide web (2011), ACM, pp. 587–596.

[6] BOLDI, P., SANTINI, M., AND VIGNA, S. Permuting web and social graphs. Internet Mathematics 6, 3 (2009), 257–283.

[7] BOURSE, F., LELARGE, M., AND VOJNOVIC, M. Balanced graph edge partition. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2014), KDD '14, ACM, pp. 1456–1465.

[8] BRODER, A., KUMAR, R., MAGHOUL, F., RAGHAVAN, P., RAJAGOPALAN, S., STATA, R., TOMKINS, A., AND WIENER, J. Graph structure in the web. Computer networks 33, 1-6 (2000), 309–320.

[9] BRONSON, N., AMSDEN, Z., CABRERA, G., CHAKKA, P., DIMOV, P., DING, H., FERRIS,J., GIARDULLO, A., KULKARNI, S., LI, H. C., ET AL. Tao: Facebook's distributed data store for the social graph. In USENIX Annual Technical Conference (2013), pp. 49–60.

[10] BUONO, D., PETRINI, F., CHECCONI, F., LIU, X., QUE, X., LONG, C., AND TUAN, T.-C. Optimizing sparse matrix-vector multiplication for large-scale data analytics. In Proceedings of the 2016 International Conference on Supercomputing (2016), ACM, p. 37.

[11] GONZALEZ, J. E., LOW, Y., GU, H., BICKSON, D., AND GUESTRIN, C. Powergraph: Distributed graph-parallel computation on natural graphs. In Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12) (2012), USENIX, pp. 17–30.

[12] HAKLAY, M., AND WEBER, P. Openstreetmap: User-generated street maps. IEEE Pervasive Computing 7, 4 (2008), 12–18.

[13] HUBER, W., CAREY, V. J., LONG, L., FALCON, S., AND GENTLEMAN, R. Graphs in molecular biology. BMC bioinformatics 8, 6 (2007), S8.

[14] KUMAR, P., AND HUANG, H. H. G-store: high performance graph store for trillion-edge processing. In High Performance Computing, Networking, Storage and Analysis, SC16: International Conference for (2016), IEEE, pp. 830–841.

[15] LI, L., GEDA, R., HAYES, A. B., CHEN, Y., CHAUDHARI, P., ZHANG, E. Z., AND

SZEGEDY, M. A simple yet effective balanced edge partition model for parallel computing. In Proceedings of the 2017 ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems (2017), SIGMETRICS '17 Abstracts, ACM, pp. 6–6.

[16] LIU, W.-H., AND SHERMAN, A. H. Comparative analysis of the cuthill–mckee and the reverse cuthill–mckee ordering algorithms for sparse matrices. SIAM Journal on Numerical Analysis 13, 2 (1976), 198–213.

[17] MALEWICZ, G., AUSTERN, M. H., BIK, A. J., DEHNERT, J. C., HORN, I., LEISER, N., AND CZAJKOWSKI, G. Pregel: a system for largescale graph processing. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (2010), ACM, pp. 135–146.

[18] MALICEVIC, J., LEPERS, B., AND ZWAENEPOEL, W. Everything you always wanted to know about multicore graph processing but were afraid to ask. In 2017 USENIX Annual Technical Conference (USENIX ATC 17) (2017), USENIX, pp. 631–643.

[19] MCSHERRY, F., ISARD, M., AND MURRAY, D. G. Scalability! but at what cost? In Proceedings of the 15th USENIX Conference on Hot Topics in Operating Systems (2015), HOTOS'15, USENIX Association, pp. 14–14.

[20] NEWMAN, M. E., WATTS, D. J., AND STROGATZ, S. H. Random graph models of social networks. Proceedings of the National Academy of Sciences 99, suppl 1 (2002), 2566–2572.

[21] NGUYEN, D., LENHARTH, A., AND PINGALI, K. A lightweight infrastructure for graph analytics. In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles (2013), ACM, pp. 456–471.

[22] NISHTALA, R., VUDUC, R. W., DEMMEL, J. W., AND YELICK, K. A. When cache blocking of sparse matrix vector multiply works and why. Applicable Algebra in Engineering, Communication and Computing 18, 3 (2007), 297–311.

[23] PENNER, M., AND PRASANNA, V. K. Cache-friendly implementations of transitive closure. Journal of Experimental Algorithmics (JEA) 11 (2007), 1–3.

[24] PRABHAKARAN, V., WU, M., WENG, X., MCSHERRY, F., ZHOU, L., AND HARADASAN, M. Managing large graphs on multi-cores with graph awareness. 41–52.

[25] ROY, A., MIHAILOVIC, I., AND ZWAENEPOEL, W. X-stream: Edge-centric graph processing using streaming partitions. In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles (2013), ACM, pp. 472–488.

[26] SHUN, J., AND BLELLOCH, G. E. Ligra: a lightweight graph processing framework for shared memory. In ACM Sigplan Notices (2013), vol. 48, ACM, pp. 135–146.

[27] SIEK, J. G., LEE, L.-Q., AND LUMSDAINE, A. The Boost Graph Library: User Guide and Reference Manual, Portable Documents. Pearson Education, 2001.

[28] SUNDARAM, N., SATISH, N., PATWARY, M. M. A., DULLOOR, S. R., ANDERSON, M. J., VADLAMUDI, S. G., DAS, D., AND DUBEY, P. Graphmat: High performance graph analytics made productive. Proceedings of the VLDB Endowment 8, 11 (2015), 1214–1225.

[29] WANG, Y., DAVIDSON, A., PAN, Y., WU, Y., RIFFEL, A., AND OWENS, J. D. Gunrock: A high-performance graph processing library on the gpu. In ACM SIGPLAN

Notices (2016), vol. 51, ACM, p. 11.

[30] WILLIAMS, S., OLIKER, L., VUDUC, R., SHALF, J., YELICK, K., AND DEMMEL, J. Optimization of sparse matrix–vector multiplication on emerging multicore platforms. Parallel Computing 35, 3 (2009), 178–194.

[31] WULF, W. A., AND MCKEE, S. A. Hitting the memory wall: implications of the obvious. ACM SIGARCH computer architecture news 23, 1 (1995), 20–24.

[32] ZHOU, S., CHELMIS, C., AND PRASANNA, V. K. Optimizing memory performance for fpga implementation of pagerank. In ReConFigurable Computing and FPGAs (ReConFig), 2015 International Conference on (2015), IEEE, pp. 1–6.

[33] ZHOU, S., LAKHOTIA, K., SINGAPURA, S. G., ZENG, H., KANNAN, R., PRASANNA, V. K., FOX, J., KIM, E., GREEN, O., AND BADER, D. A. Design and implementation of parallel pagerank on multicore platforms. In High Performance Extreme Computing Conference (HPEC), 2017 IEEE (2017), IEEE, pp. 1–6.

[34] ZHU, X., HAN, W., AND CHEN, W. Gridgraph: Large-scale graph processing on a single machine using 2-level hierarchical partitioning. In 2015 USENIX Annual Technical Conference (USENIX ATC 15) (2015), USENIX Association, pp. 375–386.