Accepted Manuscript

Partitioning big graph with respect to arbitrary proportions in a streaming manner

Ke-kun Hu, Guo-sun Zeng, Huo-wen Jiang, Wei Wang

PII: S0167-739X(17)30342-4
DOI: http://dx.doi.org/10.1016/j.future.2017.06.027
Reference: FUTURE 3525

To appear in: *Future Generation Computer Systems*

Received date : 4 March 2017
Revised date : 5 June 2017
Accepted date : 25 June 2017

Please cite this article as: K. Hu, G. Zeng, H. Jiang, W. Wang, Partitioning big graph with respect to arbitrary proportions in a streaming manner, *Future Generation Computer Systems* (2017), http://dx.doi.org/10.1016/j.future.2017.06.027

Partitioning Big Graph with respect to Arbitrary Proportions in a Streaming Manner

Ke-kun HU, Guo-sun ZENG*, Huo-wen JIANG and Wei WANG
Department of Computer Science and Technology, Tongji University, Shanghai, 201804, China
Tongji Branch, National Engineering & Technology Center of High Performance Computer, Shanghai, 201804, China
hookk@msn.com, {gszeng, hwjiang, wwang}@tongji.edu.cn

Abstract: Using a single commodity computational node to partition big graph is very difficult. This work studies how to partition a big graph with respect to arbitrary proportions in a streaming manner. To meet diverse requirements of big graph partitioning scenarios, we first devise 3 measurement schemes for measuring the graph vertex count, graph workload, and graph processing time, respectively. These schemes are the bases and prerequisites for big graph partitioning. Due to the difficulty in acquiring full big graph information, we then design 8 streaming heuristics to partitioning a big graph during the process of loading its data from external disks into memory. Each of these heuristics decides where to assign every vertex in the stream based on the information calculated by one of the above 3 schemes. At last, we demonstrate the performance and flexibility of our heuristics in partitioning real and synthetic graph datasets on a medium-sized cluster. The characteristics of arbitrary proportions of our approach makes it have a wide range of applications.

Key words: Graph partitioning; arbitrary proportions; measurement scheme; streaming heuristic; metric.

1 Introduction

The size of modern graph datasets generated in many domains is very big. For example, as a popular social media network, the Facebook graph had over 1.79 billion vertices, representing its users, and 138 billion edges, representing friendships between users as of September 2016 [1]. Extracting knowledge from these graphs is an important application that is attracting more and more researchers from both industry and academic. However, running algorithms with even simple operations over these graphs are often time-consuming and beyond the capability of a single computational node, which necessitates a parallel and distributed computing approach involving in many nodes of a cluster. This approach requests to partition graphs first.

Graph partitioning is an essential preprocessing step of big graph analytics. It partitions a big graph into smaller subgraphs according to the cluster architecture on which big graph analytics system deployed through cutting edges or vertices by a certain measurement scheme such as the vertex count, graph workload, and graph processing time. Its goal is often to match each subgraph's intra-subgraph measurement to the computing or storage capability of a node it is assigned to, and/or to minimize the inter-subgraph measurements across subgraphs. Obtaining an optimal graph partition that achieves either or both of the above two goals is a key to improve system performance and utilization. We name it a well-matched partition. However, finding such a partition is NP-hard [2] and faces big challenges. They are summarized as follows: (1) Large graph size. Modern graph datasets often have billions of nodes and trillions of edges [3]. Traditional graph partitioning methods such as spectral method [4], METIS [5] do not scale to these big graphs because their current implementations are time-consuming and require full graph information. Thus, lightweight ones are needed. (2) Diversity in graph workloads and architectures of clusters. Clusters can be either homogeneous or heterogeneous.

* The corresponding author.