Efficient Distributed Graph Algorithms

Li Shuhe

The Chinese University of Hong Kong

Introduction

Husky is a data-parallel computing system. It's based on shared nothing architecture. Master can assign work to multiple workers. Each worker manages own partitions of objects.

My research is looking for efficient distributed graph algorithm on Husky. Aiming on solving basic graph problems, I finished several implementations, including connected component, shortest path, triangle counting and spanning forest.

Method

Connected component:

I use hash-min algorithm. The idea is broadcasting the smallest vertex ID. Recursively execute sending messages and dealing with messages. When the processing terminates, the connected vertices will be labeled with the smallest vertex's ID.

SSSP:

The algorithm to solve single source shortest path (SSSP) is very similar to the algorithm of connected component. Broadcast the smallest cost from the the source. Recursively execute sending messages and dealing with messages, label the vertex when they reached.

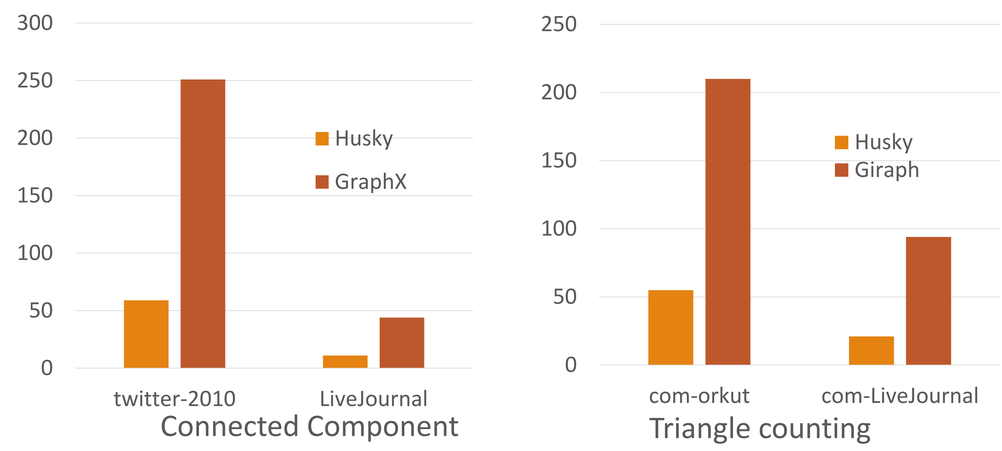
Triangle counting:

There exist many algorithms to solve triangle counting problem [1]. I have tests some of them. The algorithm I used finally shows great performance which will be mentioned at result part. The idea is to make vertex in order and send part of successors as messages to some successors. When the vertex receives the messages, it can get the intersection of messages and successors which is the number of triangle.

MSF:

The algorithm is full of pulling requests and getting respond. The idea is that construct subgraph of MSF and unify them.

Results



Data name	Twitter-2010 [3]	LiveJournal [3]	Com-orkut [3]	Com-LiveJournal [3]
|V|	41,652,230	4,847,571	3,072,441	3,997,962
|E|	1,468,365,182	68,993,773	117,185,083	34,681,189

I tried to keep similar environments when executed different implementations. All of these experiments were executed with one master and 16 workers. The running time of both connected component and triangle counting implementation is much less.

Conclusion and future work

I have introduced my research in summer. Those implementations are not difficult but useful. Connected component, SSSP, triangle counting and MSF can runs faster than other implementations. But speed and memory consumption should keep in balance. Thus, in the future I may try to use Webgraph in distributed system to compress the graph.

References:

[1]. M. N. Kolountzakis, G. L. Miller, R. Peng, and C. E. Tsourakakis. Efficient triangle counting in large graphs via degree-based vertex partitioning. *IM*, 8(1-2):161–185, 2012.

[2]. J. E. Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M. J. Franklin, and I. Stoica. Graphx: Graph processing in a distributed dataflow framework. OSDI, pages 599–613, 2014.

[3]. SNAP, <https://snap.stanford.edu/data>