**Paper Review - [Pregel: A System for Large-Scale Graph Processing](https://xduan7.com/2016/03/01/paper-review-pregel-a-system-for-large-scale-graph-processing/)**

**Summary:** Parallelizing and distributing large graphs is a hard problem due to dependency of millions and trillions of vertices and edges. Pregel offers an intuitive way to handle graph processing by using N number of machines to handle M number of vertices separately, where M is usually greater than N. The system comprises of a single master which is responsible for coordination, aggregation and analysis.

**Introduction:** Nowadays, large scale graphs like the Web graph and social networks, are among the main sources of new computing problems. Efficient processing of large graphs in real world is a challenging task as the algorithms to handle such graphs need to exhibit good performance on various aspects like distribution along with locality of memory access. MapReduce can be a solution but inefficient due to the need of passing the entire state of the graph from one stage to another. Hence, Pregel is introduced by the authors of this paper.

**Literature Review:** There has been a gap in the area of the frameworks for large scale graphs processing that can offer scalability, while being distributed and fault tolerant. The authors have designed Pregel for the Google cluster architecture in which clusters are interconnected and geographically distributed with each one of them having a number of commodity machines. The main contributions by authors are: a) Design of a fault tolerant distributed programming framework for enabling execution of graph algorithms in parallel over a number of machines. b) Provision of API with direct message passing among the vertices, combiners for reducing overhead, aggregators for global communication and monitoring. c) topology mutations by solving conflicting requests.

**Implementation:** Pregel works as a repeated synchronized computation process over vertices. Once the graph has been given as an input, Pregel divides the graph into partition which include a set of vertices and their outgoing edges. These vertices are provided to the worker machines and one of them acts as a master for coordinating the worker machines. The workers then undergo a series of iterations (supersteps). In each superstep, all vertices in each worker machine execute the same user defined function which can: a) receive messages sent during the previous superstep, b) change the state of the vertex and its outgoing edges, c) send messages to be delivered during the next superstep. At the end of each superset a global synchronization point occurs. The vertices can become inactive and the sequence of iterations terminates when all vertices are inactive and there no messages in transit. During the computation, the master also sends ping messages to check for worker failures. The network is only used for sending messages and therefore it significantly reduces the communication overhead becoming more efficient.

**Results:** Developed an efficient way of processing large graphs where distribution and parallelism is carried out with good locality memory access and more work on each vertex using the concept of worker machines.

**Discussion:** Some of the strong points of this work are: a) Fault tolerance is achieved with use of checkpoints, in which the state of the nodes’ partitions is saved to persistent storage that enables nodes to start from the most recent state in times of machine failures, b) Combiners optimize the network traffic and can be enabled manually by users that enables messages to be combined and sent in a single message, reducing the overhead. c) Use of aggregators, a mechanism for global communication and monitoring.

On the other hand, there are some weak points of this work: a) User needs to do a lot of modifications to the Pregel environment for personized needs. From coding to enable combiners to customizing aggregators along with solving conflicting requests to defining handlers, increasing complexity in the system, b) No failure detection is mentioned for the master, making it a single point failure, c) There is no clear comparison with other systems like MapReduce, an experimental comparison would have been interesting.

**Conclusion:** Pregel is a model suitable for large graphs with millions of vertices and various real world Pregel applications are performing with efficiency. API provided Pregel is intuitive, flexible and easy to use. However, techniques can be developed in future to relax the synchronicity of the model to avoid the cost of faster workers having to wait frequently at inter-superstep barriers.