*These papers show how **large-scale graph computations can be performed on a single machine** with good performance. What motivation and techniques are the **same** across at least two of the systems? What motivation and techniques are **different** across at least two of the systems?*

Answer:

For **GraphChi and X-stream**, their **motivations are same**:

1. Transfer edges access(random disk I/O) into sequential disk I/O;
2. Save portion of graph into memory

But the **methods(techniques) of GraphChi and X-stream are different**.

1. GraphChi(vertex-centric): PSW(parallel sliding window), partitions edges into sorted shards.
2. X-Stream(edge-centric): streaming partitions, partitions edges into unsorted streaming partitions

For **FlashGraph**, the motivation is different with GraphChi/X-stream, FlashGraph try to perform many parallel random operations instead of sequential operations that are not needed, and all the edges are saved in SSDs.

The techniques of FlashGrap:

1. Access edges list only required by an application. (Reduce I/O)
2. Conservatively merge I/O requests. (increase sequential I/O when possible)

Questions:

For streaming partitions, disjoint portions of every partition can be done in parallel, and then joint portion will shuffle to each other, seems like merge sort, but I am confused by how does the merge action work between different partitions? Does it just do like BFS scan of the whole edges? And the shuffle expense is cheaper than random access of vertices?