Scalable Computing for Power Law Graphs

experience with parallel PageRank

1. Websearch
Modern websearch engines consist of two stages: an indexing stage and a query stage. At the indexing stage, a web-crawler traverses links between web pages and builds a text database and link database for all pages on the web. These two databases form the core of a search engine. We perform off-line analysis of the link database to statically compute measures like PageRank.

2. PageRank Model
To rank all the pages on the web, PageRank models a random surfer tossing the web and uses the stationary distribution of the associated Markov chain as the ranking score. Assume we are given a web-adjacency matrix, \( A \), a parameter, \( c \), and a prior probability distribution over pages, \( v \).

1. Construct the random walk matrix, \( \overline{P} = D^{-1}A \).
2. Add links from dangling pages, \( \overline{P}' = \overline{P} + c I \).
3. Add random moves, \( \overline{P}' = \overline{P}' (1 - c) + c I \).

The PageRank vector is the stationary distribution of the Markov transition matrix. A unique stationary distribution exists because \( \overline{P}' \) is a strictly positive matrix and the Perron-Frobenius theorem applies.

3. Power Law Data
A power law graph is defined by the property that the number of vertices with degree \( k \) is proportional to \( k^{-\beta} \) for power law exponent \( \beta \). Our graphs have \( \beta \) between 1.4 and 2.0.

4. Computation
PageRank can be computed from an eigensystem to a linear system by way of the following transformation. This transformation gives us a wide-range of computational possibilities.

5. Results
Our goal was to compute the PageRank vector as quickly as possible for a given tolerance. Each plot below shows convergence of all of the methods in terms of number of iterations and time.

6. Scaling
We observed that a tridiagonal distribution of the matrix to processors (balancing only number of rows) displayed oscillatory behavior as we scaled the number of processors. This behavior is smoothed by our heuristic load-balancing distribution. Additionally, the more sophisticated linear solvers (i.e., the KSP methods) showed better scalability than simple power iterations.

7. Host Ordering
The webgraph has the property that pages within a particular host (e.g., echolifighthouse.com) are much more connected than pages between hosts. Because we have the URLs for each page (which includes the host), we can exploit this property to reorder the matrix and improve parallel performance.

8. Conclusions
- The power iteration and Jacobi methods have approximately the same behavior on our graphs.
- The convergence of Krylov methods strongly depends on the graph and is not monotonic.
- Krylov methods converge fastest by number of iterations, but the actual run time may be longer than simple power iterations.
- BICGSTAB and GMRES converge in the smallest number of iterations. GMRES demonstrates more stable behavior.
- The BICGSTAB algorithm scales better than power iterations due to the parallelism in the extra work performed.
- The best method to use is either power iterations or BICGSTAB. The final choice of method is dependent on the time of a parallel matrix-vector multiply compared with the time of the extra work performed in the BICGSTAB algorithm.