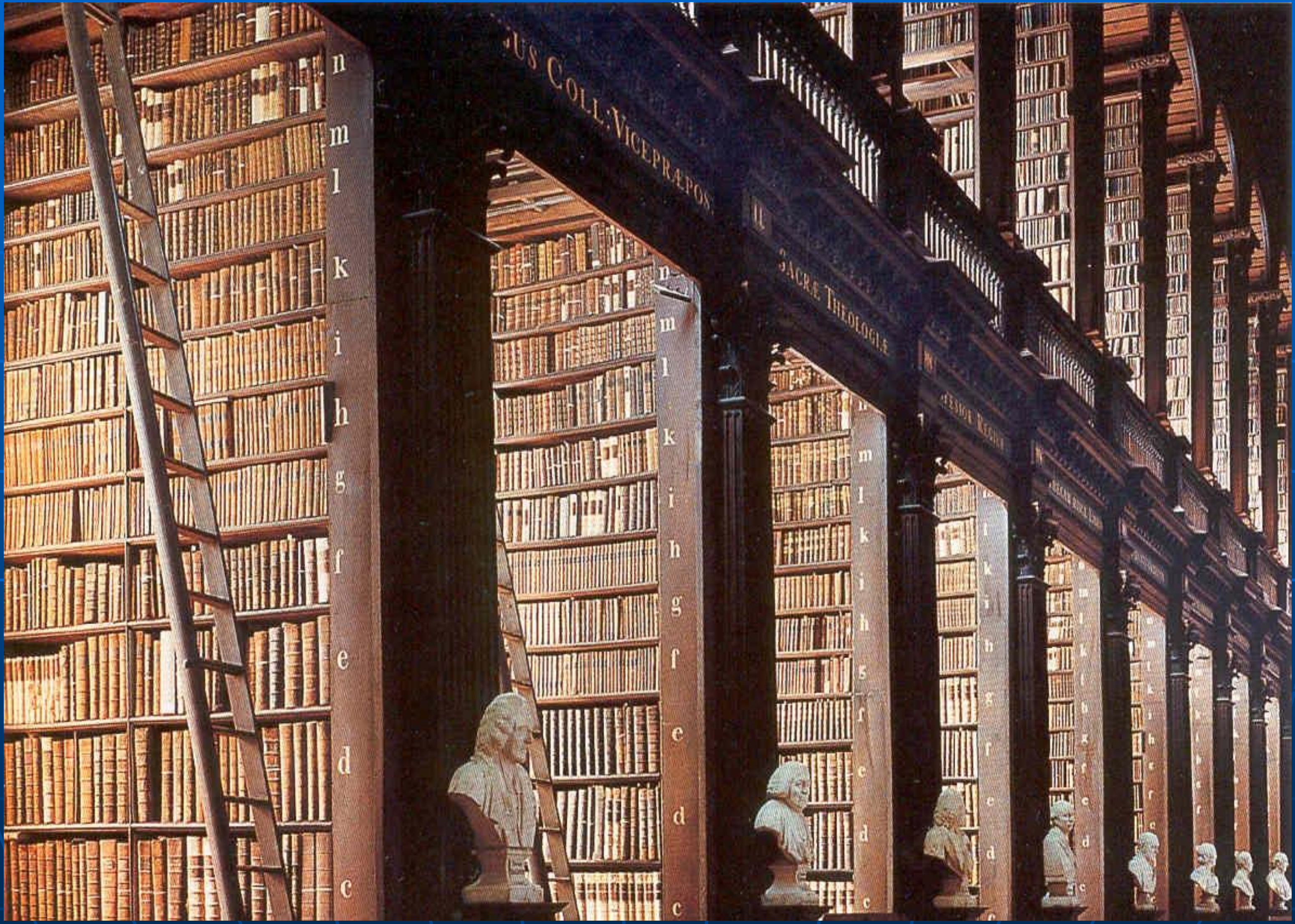
Accelerating the Pagerank Algorithm

M. Campbell
Missouri State University REU

The Information Retrieval Problem



Actually two…

Given a finite set of Documents *D* and a query *q*…

I. Which elements of D are relevant to q?

II. Of the relevant documents, which are *most* relevant?

Exploiting the Structure of the document set

Scholarly papers:

Papers cite other papers.

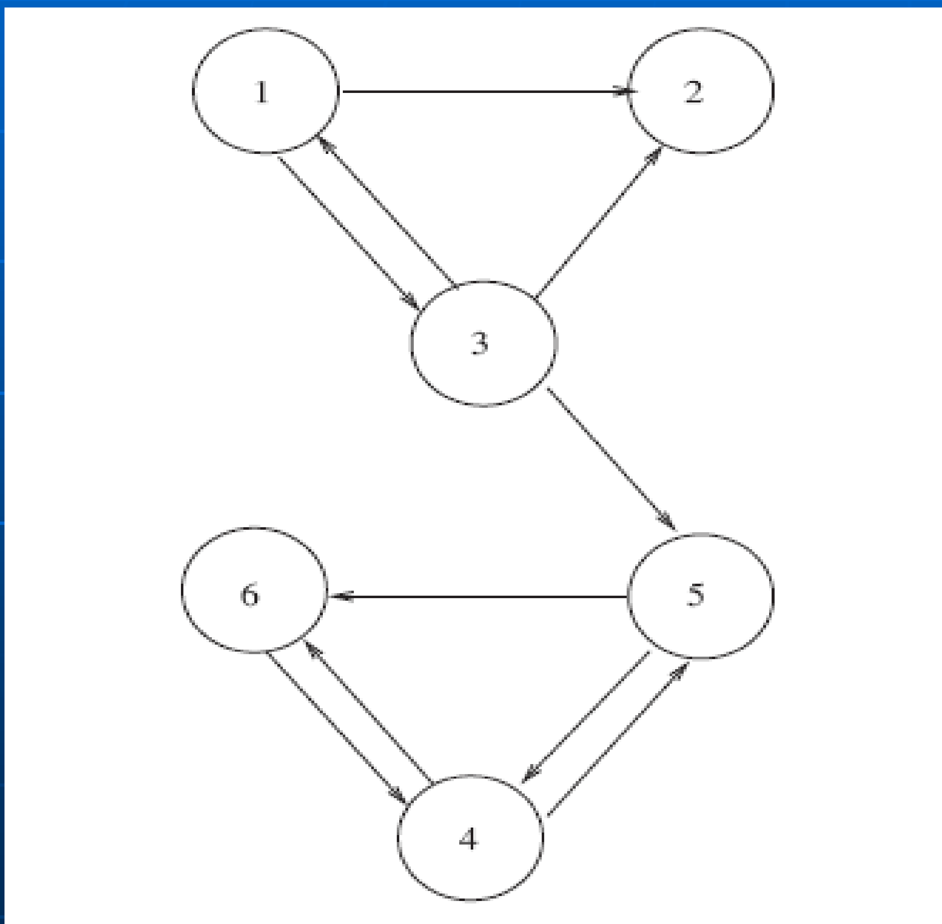
Papers which are cited the most are likely to be very important in their field. Additionally, the papers cited by important papers gain in relative importance.

What about the internet?

The internet has a similar structure due to hyperlinking.

Pages which are very important get linked to by many pages, and pages linked to by important pages will likely be deemed to be more important than others.

Looking abstractly at the link structure of the web



$$
\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{pmatrix}
0 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
\end{array}
$$

The Pagerank Equation

$$r(P) = \sum_{Q \in B_P} \frac{r(Q)}{|Q|}$$

The Iterative Pagerank Equation

$$r_k(P_i) = \sum_{Q \in B_{P_i}} \frac{r_{k-1}(Q)}{|Q|}, \quad k = 1, 2, \ldots$$

Determining the Pagerank Vector by the Power Method

Denote $v_k = [r_k(P_1) \quad r_k(P_2) \quad \cdots \quad r_k(P_n)]^T$

$$v_k = H\, v_{k-1}, \qquad \text{with} \quad H_{ij} = \begin{cases} \frac{1}{|P_j|} & \text{if } P_i \text{ has a link from } P_j \\ 0 & \text{otherwise.} \end{cases}$$

This is the power method, where we are computing the eigenvector of H associated to the eigenvalue of 1

Fixing the Link matrix to ensure the pagerank vector exists

I. Dangling nodes

$$B = H + u\,a^T,$$

$$a_i = \begin{cases} 1 & \text{if page } i \text{ is a dangling node} \\ 0 & \text{otherwise} \end{cases}$$

$$u = e/n$$

Fixing The link matrix

II. Reducibility (dangling webs)

$$G = \alpha B + (1-\alpha)E$$

$$E = u\,e^T$$

U is a probabilistic (entries add to one) “personalization” vector

The Google matrix

$$
\begin{aligned}
G\,v_k &= \alpha\,Bv_k + (1-\alpha)Ev_k \\
&= \alpha H v_k + \alpha\,u\,a^T v_k + (1-\alpha)u\,e^T v_k \\
&= \alpha H v_k + \alpha\,u\,a^T v_k + (1-\alpha)u.
\end{aligned}
$$

An alternate Method

$$G\,v \;=\; v$$

$$G \;=\; \alpha\,(H + u\,a^T) + (1-\alpha)\,u\,e^T$$

$$\alpha\,(H + u\,a^T)\,v \;+\; (1-\alpha)\,u\,e^T v \;=\; v$$

$$(I - \alpha H - \alpha u a^T)v = (1-\alpha)u$$

The linear system

Letting $R = I - \alpha H$

It has been shown that ***v*** = ***rx*** for some scalar ***r*** where ***x*** is the solution of the system

$$Rx = u$$

Options for solving the system

There are many options for solving this system. I focused on three.

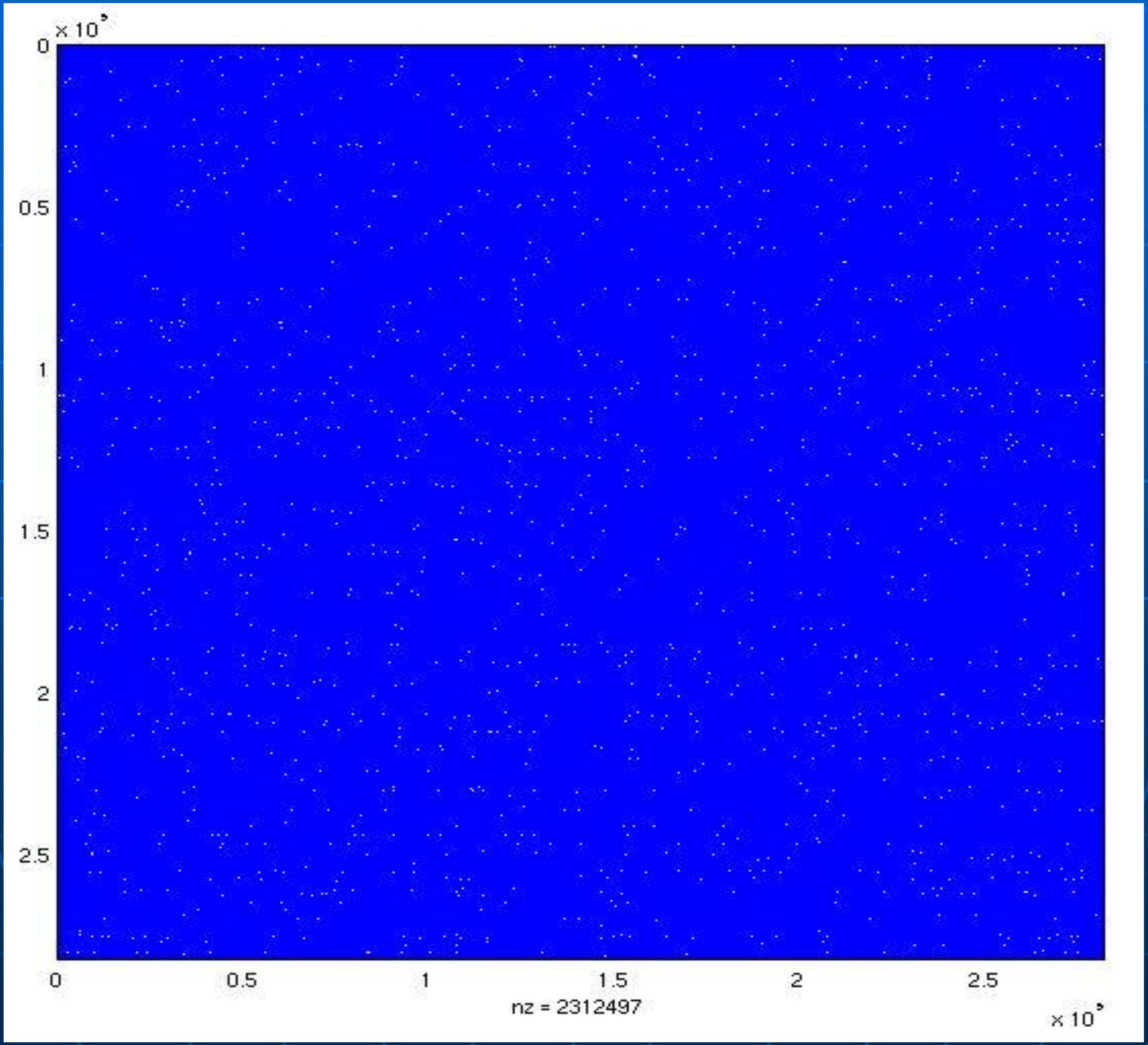
I. Jacobi

II. Gauss-Seidel

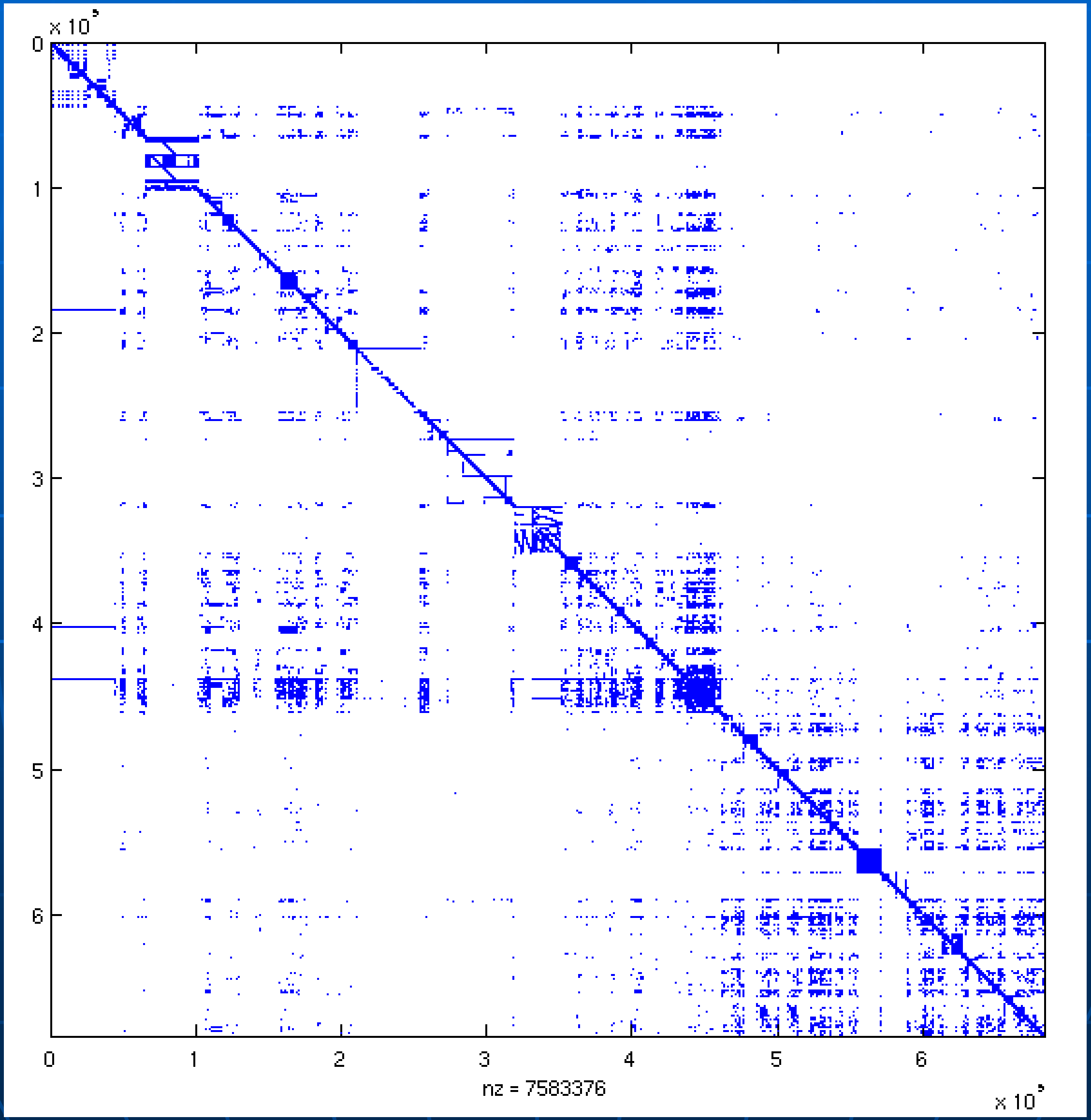
III. Successive Over Relaxation(SOR)

But first we study reorderings of the matrix to make it “nice” for the solver

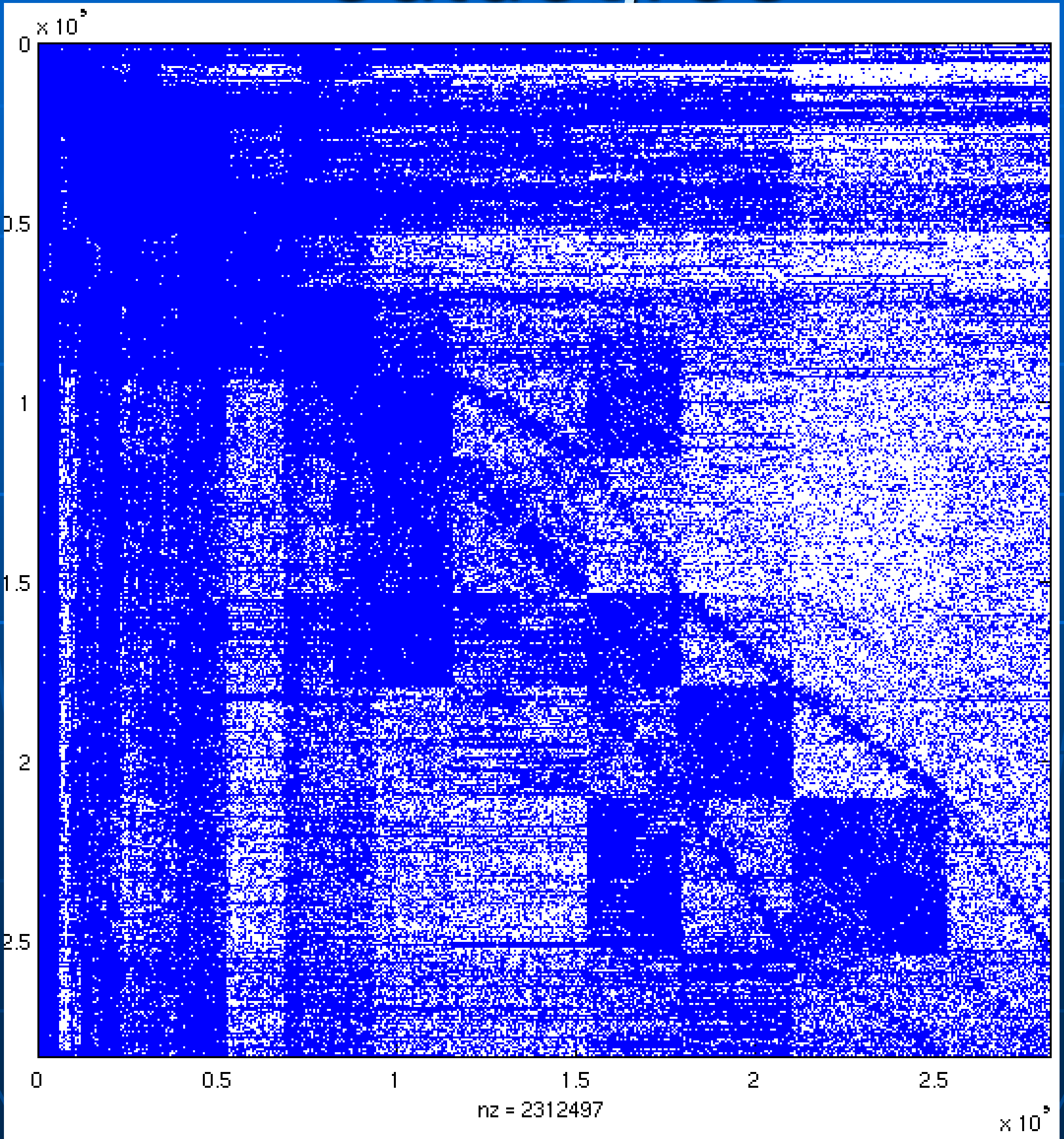
Stanford.edu web



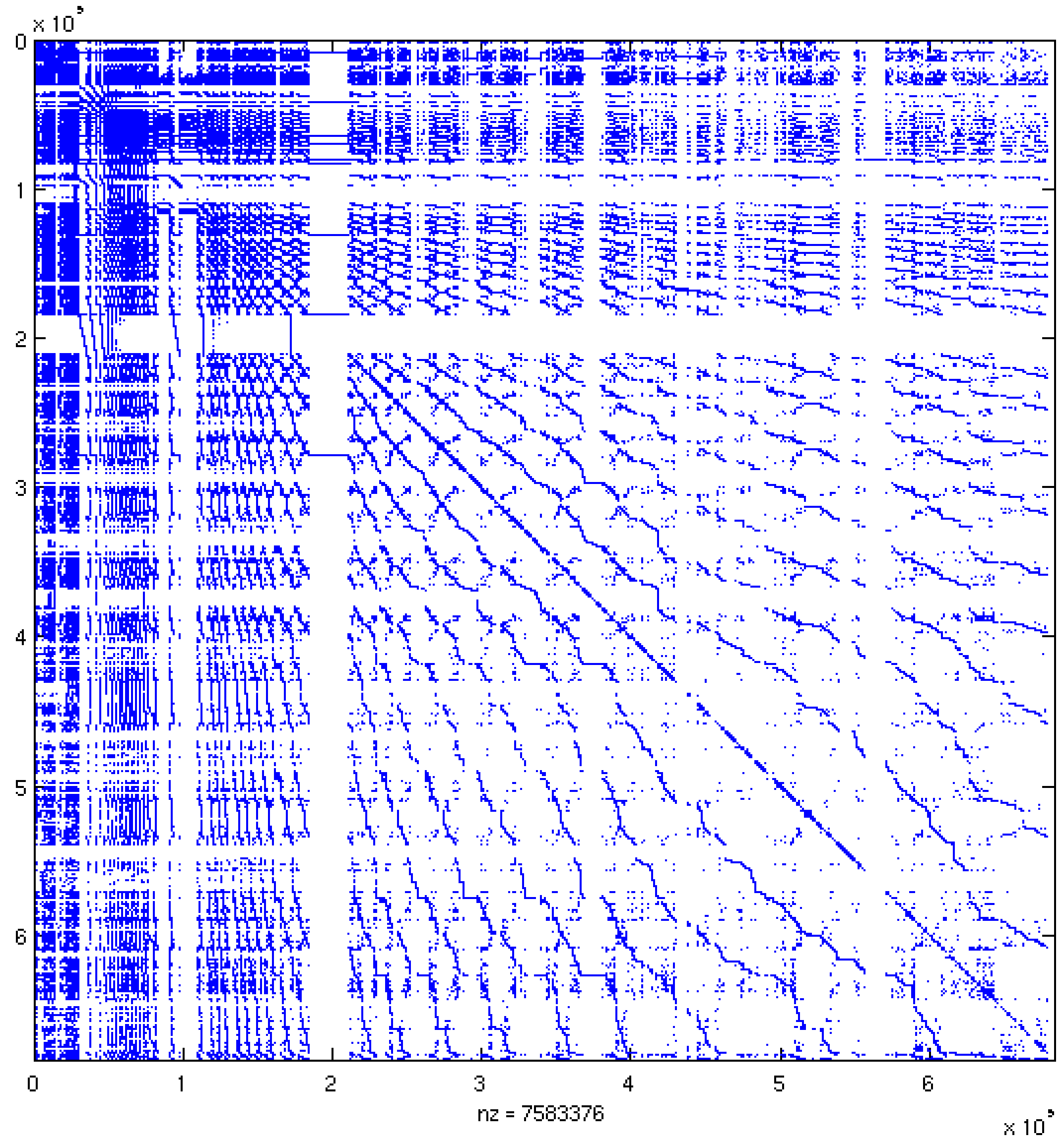
stanford.edu/berkley.edu web



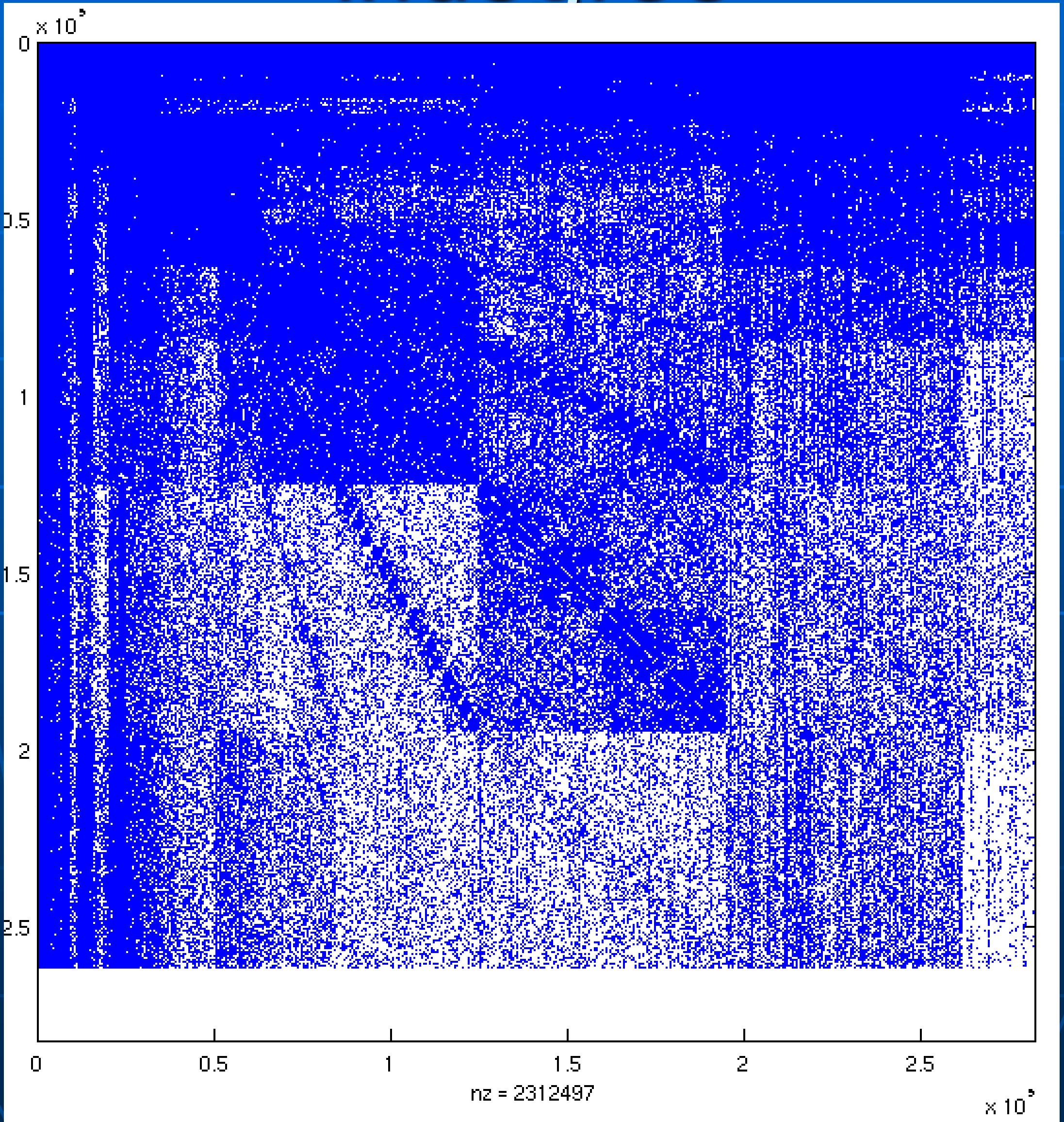
Stanford Reordered by descending outdegree



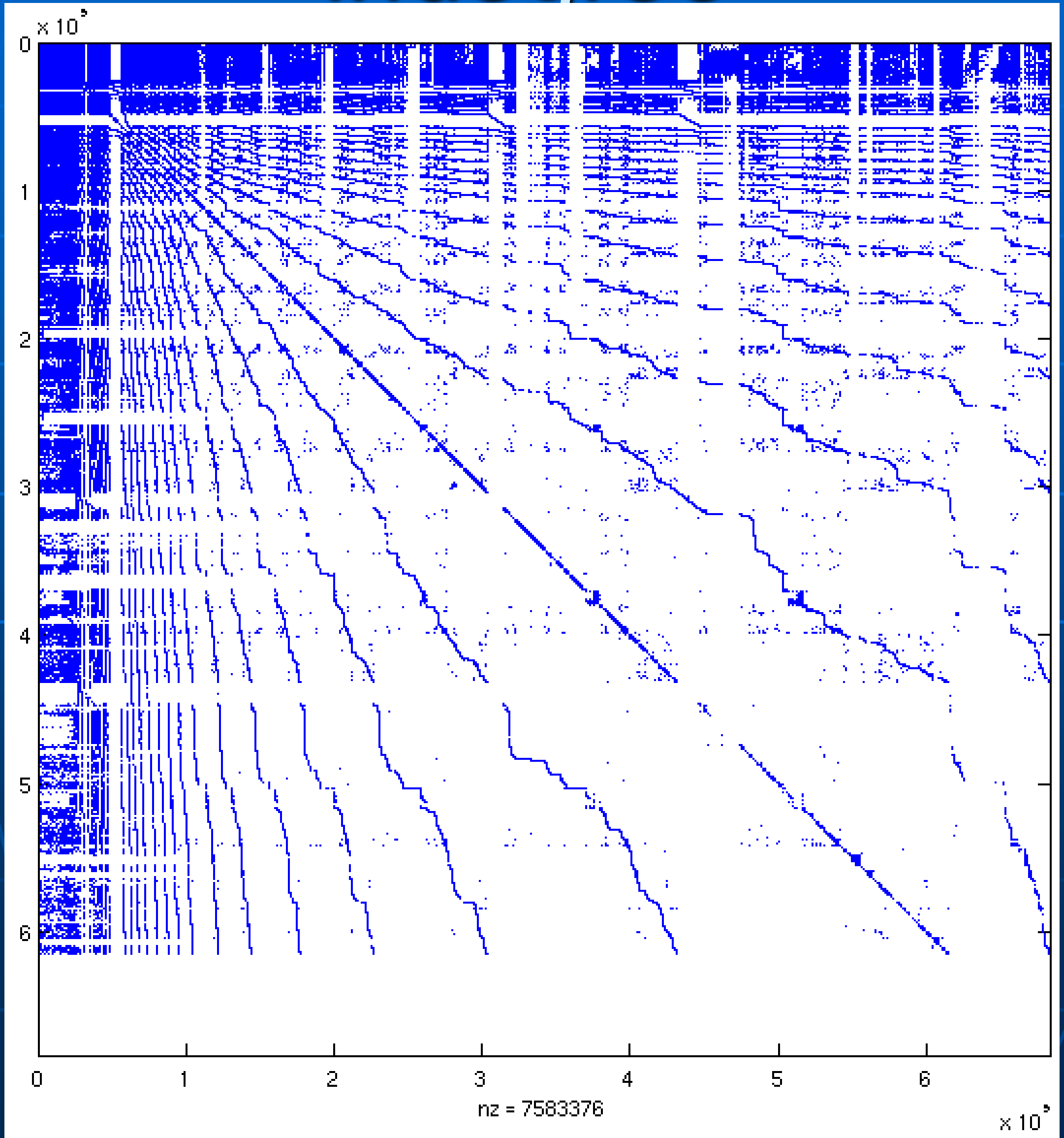
SB Reordered by descending outdegree



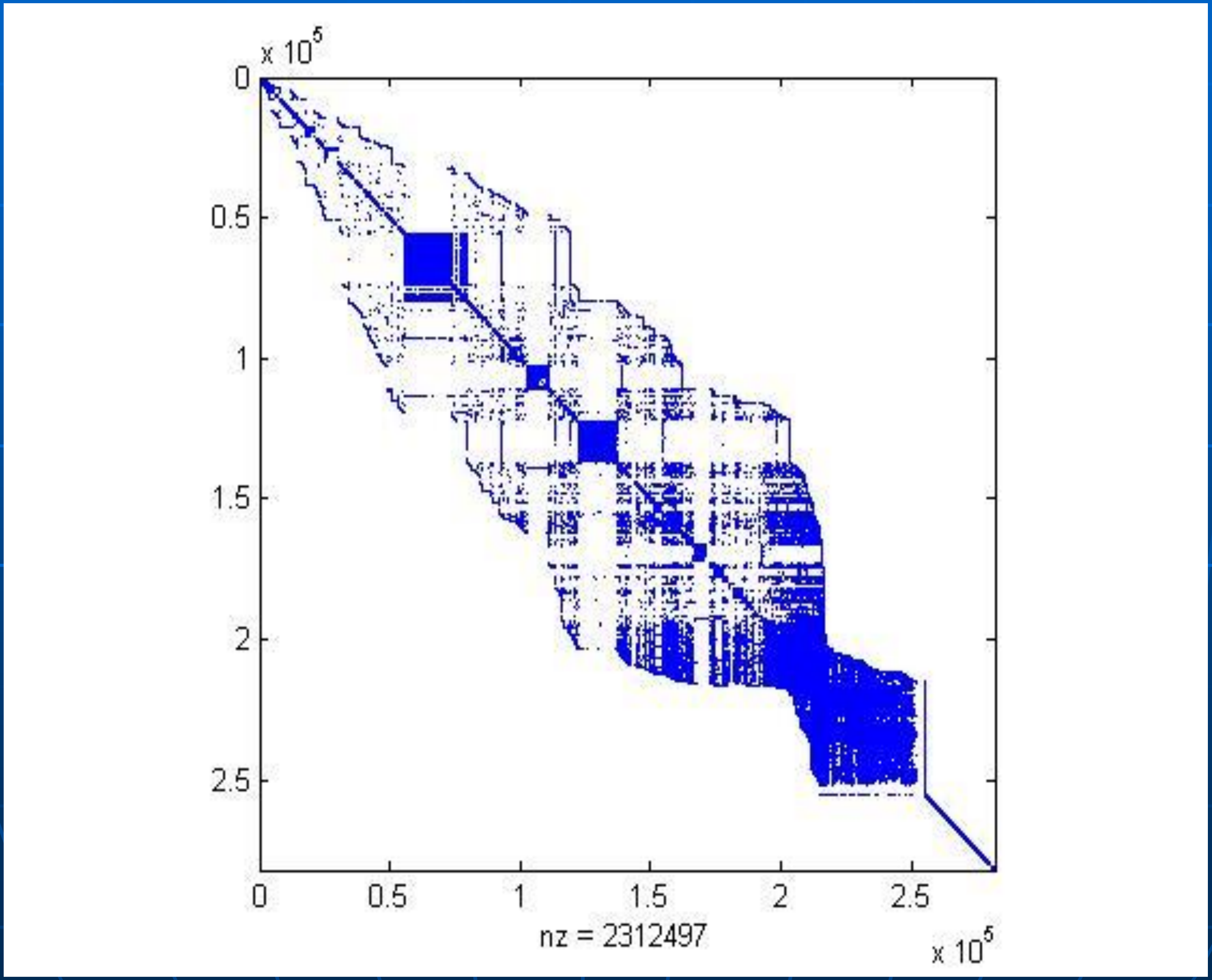
Stanford Reordered by descending indegree



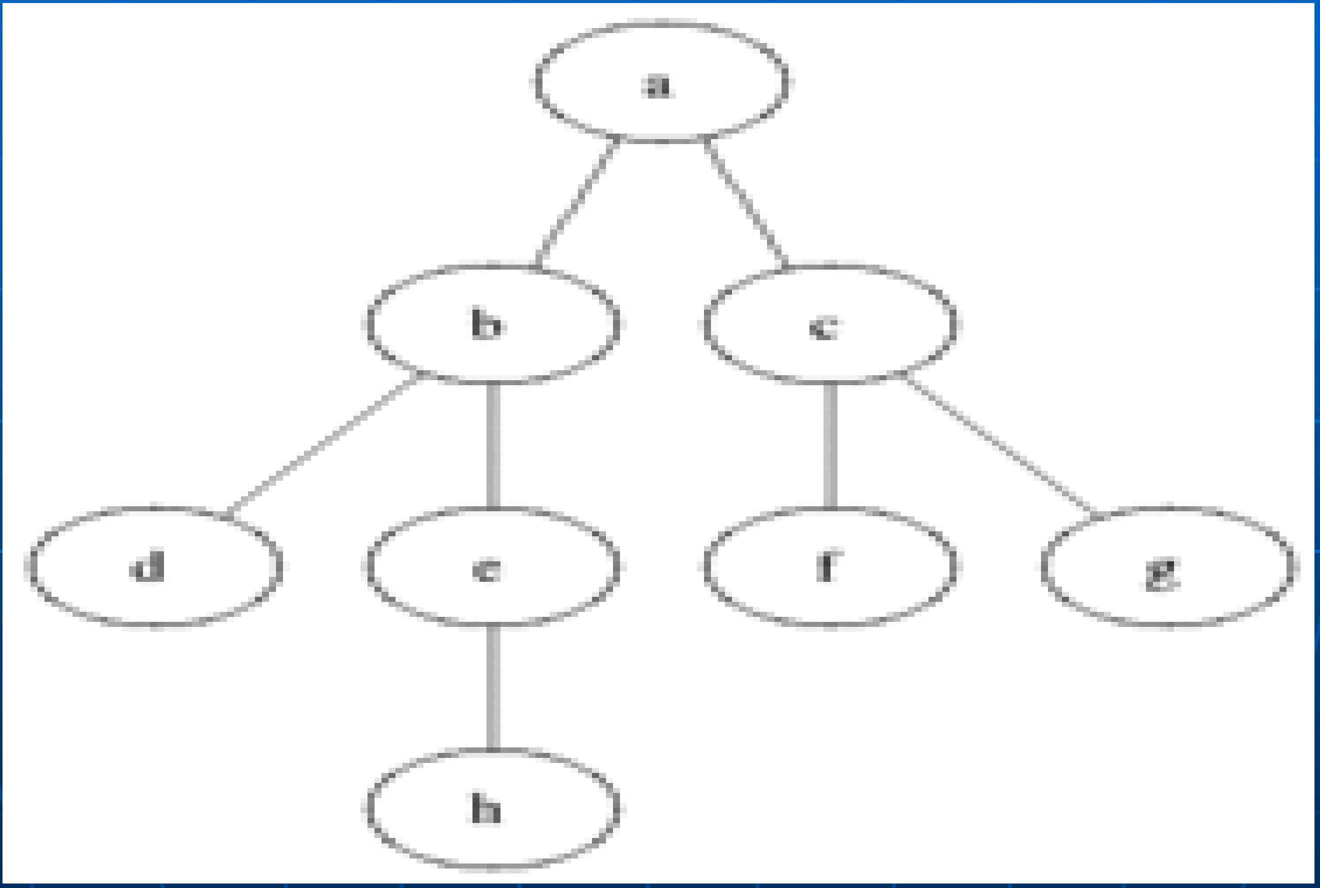
SB Reordered by descending indegree



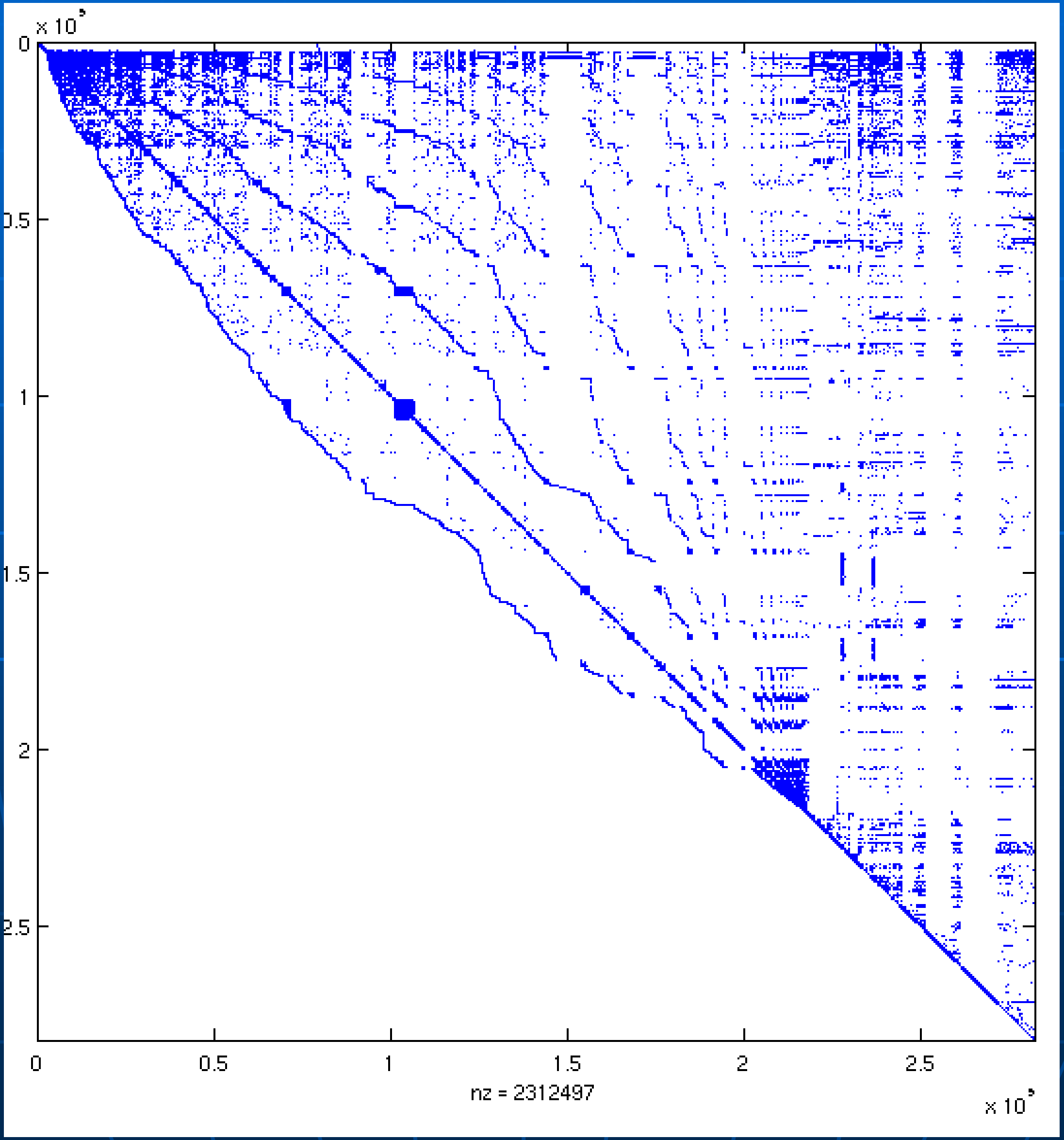
Reverse Cuthill Mckee



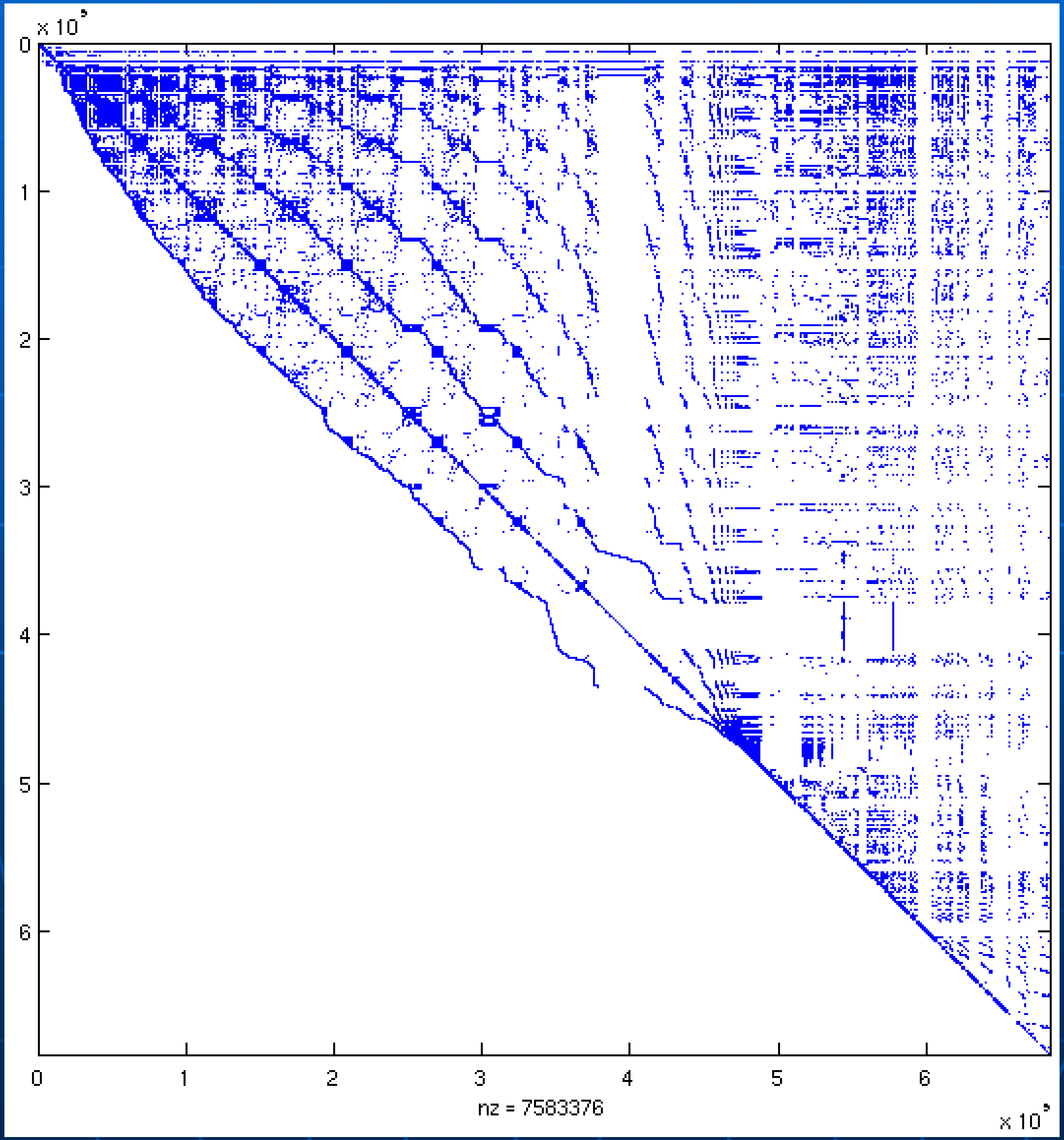
The Breadth first search



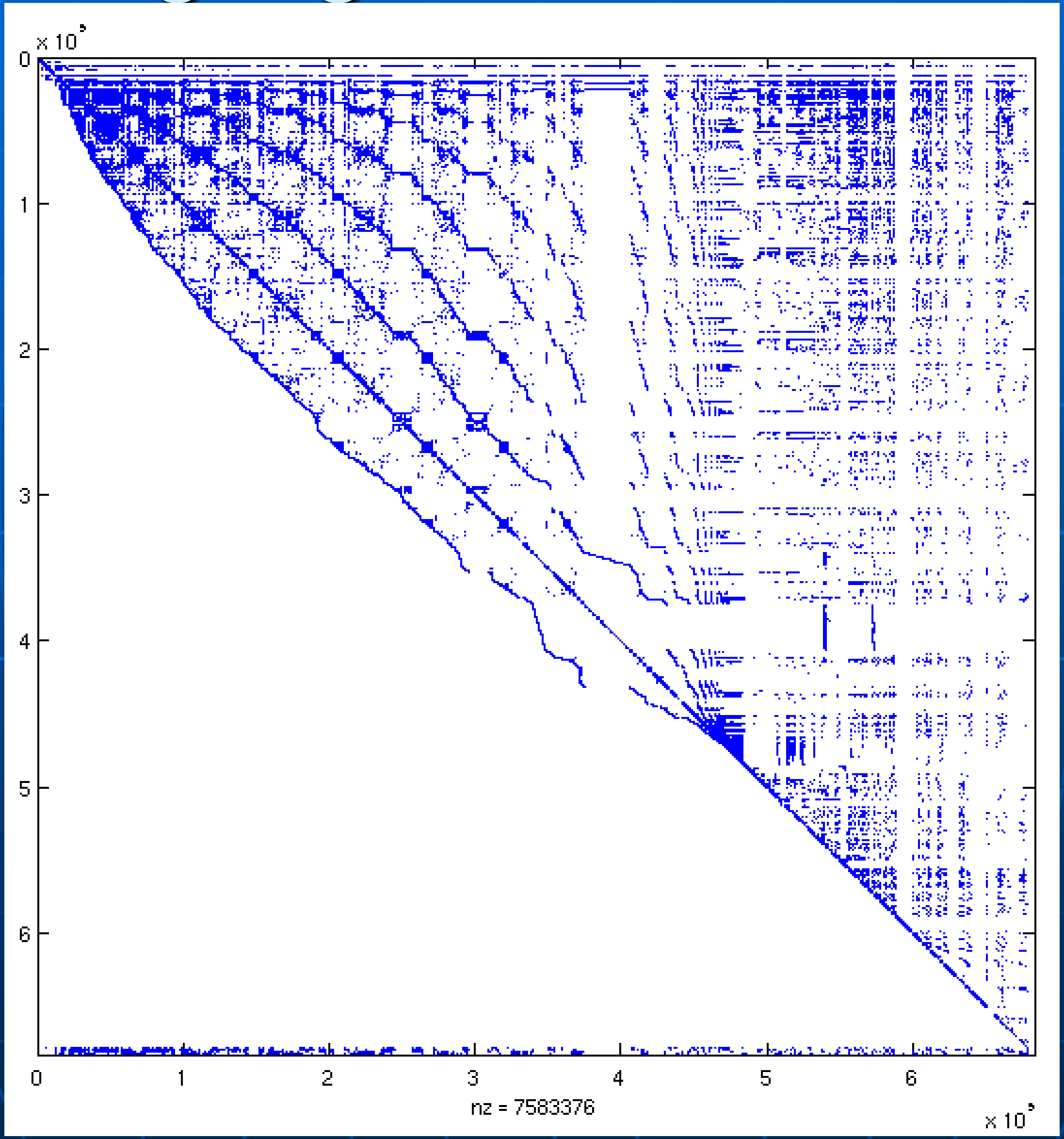
BFS reorder on Stanford web



BFS on Stanford/Berkley



The dangling node/BFS reordering



Solving the BFS/Dangling system

$$\begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} - \alpha \begin{pmatrix} H_{11} & 0 \\ H_{21} & 0 \end{pmatrix} = \begin{pmatrix} I - \alpha H_{11} & 0 \\ -\alpha H_{21} & I \end{pmatrix}$$

$$\begin{pmatrix} I - \alpha H_{11} & 0 \\ -\alpha H_{21} & I \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

$$\begin{cases} \left(I - \alpha H_{11}\right) x_1 = u_1 \\ x_2 = u_1 + \alpha H_{21} x_1 \end{cases}$$

Comparative Results

Web:	Stanford			Stanford/Berkley		
	Time(s)	N.Iter.	residual	Time(s)	N. Iter.	residual
Power	10.32	132	8×10^{-12}	28.9	134	7×10^{-12}
Jacobi	5.8186	146	9×10^{-12}	17.42	144	1×10^{-11}
GS	11.289	68	5×10^{-11}	29.441	70	3×10^{-11}
SOR	10.7	64	6×10^{-11}	29	68	4×10^{-11}

Further studies

I. Preconditioning

II. Optimal implementation of Gauss-Siedel/SOR Algorithm

III. Markov Chain Updating Problem with Linear Solving

IV. Using Kendall-tau measure for convergence criterion.