Advanced Computer Architecture: Google 2

Jeremy Bradley

8 March 2004

Figure 1: Graph G_1

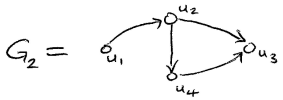

Figure 2: Graph G_2

1. The graph G_1 represents a set of connected web pages. Taking $c = 4/5$, write down:
   (a) The transition matrix P
   (b) The personalisation vector $\vec{p}$
   (c) The modified transition matrix A
   (d) Two iterations of $\vec{x}_{(k+1)} = \vec{x}_{(k)}A$, i.e. $\vec{x}_{(1)}, \vec{x}_{(2)}$ with $\vec{x}_{(0)} = (0, 1, 0)$
   (e) Two iterations of the PageRank algorithm for comparison
   (f) The value $\delta = || \, |\vec{x}_{(k+1)} - \vec{x}_{(k)}| \, ||_1$ for each iteration

   If you have access to a machine you can implement the PageRank algorithm on slide 17 using an ϵ-value of 0.01, instead of doing parts (d) and (e) by hand. How many iterations does it take to converge?

   - **For graph G_1,** $P = \begin{pmatrix} 0 & 1 & 0 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{pmatrix}$
   - **Personalisation vector,** $\vec{p} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}$
   - $A = cP' + (1-c)E = \frac{2}{5}\begin{pmatrix} 0 & 2 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} + \frac{1}{15}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

   - **Two iterations of $\vec{x}_{(k)}A$/PageRank:**

$$\vec{x}_{(1)} = \frac{1}{15}(7, 1, 7); \quad \vec{x}_{(2)} = \frac{1}{225}(63, 141, 21)$$

   - **After one iteration: $\delta = \frac{28}{15}$. After two, $\delta = \frac{252}{225}$**

2. Repeat question 1 for the graph G_2, this time with $\vec{x}_{(0)} = (1, 0, 0, 0)$. How does the answer/convergence vary if you alter c?

   - **For graph G_2,** $P = \frac{1}{2}\begin{pmatrix} 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \end{pmatrix}$
   - **Personalisation vector,** $\vec{p} = \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$
   - $A = cP' + (1-c)E = \frac{1}{5}\begin{pmatrix} 0 & 4 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 4 & 0 \end{pmatrix} + \frac{1}{20}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$
   - **Two iterations of $\vec{x}_{(k)}A$/PageRank:**

$$\vec{x}_{(1)} = \frac{1}{20}(1, 17, 1, 1); \quad \vec{x}_{(2)} = \frac{1}{50}(3, 5, 22, 20)$$

   - **After one iteration: $\delta = \frac{19}{10}$. After two, $\delta = \frac{3}{2}$**

3. Suggest scalable techniques for implementing each of the vector operations below across several processors, where as necessary each processor has the same partitioned set of rows for each vector $\vec{v}_1$ and $\vec{v}_2$
   (a) $v_1 + v_2$
   (b) $||v_1||_1$
   (c) αv_1 for some scalar multiplier, α

   This uses a technique know as row-striping where processor 1 stores the first k entries of the vectors $\vec{v}_1$ and $\vec{v}_2$, processor 2 stores the next k and so on. Where a vector is the result of the distributed calculation, each processor should store the appropriate k entries of the result vector. Where there is a single scalar result, you can assume a central master machine to collate the distributed calculation.

   Processor i holds elements $k(i-1)+1$ to ki of $\vec{v}_1$ and $\vec{v}_2$. Call these $(v_{1_{k(i-1)+1}}, \ldots, v_{1_{ki}})$ and $(v_{2_{k(i-1)+1}}, \ldots, v_{2_{ki}})$.

(a) Processor i calculates and stores $(\vec{v}_1 + \vec{v}_2)_{k(i-1)+j} = v_{1_{k(i-1)+j}} + v_{2_{k(i-1)+j}}$ for $1 \leq j \leq k$.

(b) Processor i calculates $s_i = \sum_{j=1}^{k} |v_{1_{k(i-1)+j}}|$. Master processor collects s_i values and performs $\sum_{i=1}^{n} s_i$ for n processors.

(c) Processor i calculates and stores $(\alpha v_1)_{k(i-1)+j} = \alpha v_{1_{k(i-1)+j}}$ for $1 \leq j \leq k$.