PageRank

Ng Wing Hin

Introduction

Here's the topics we are going to explore :

- PageRank
- Topic-sensitive PageRank
- Binning and Vertex-centric Gather-Apply-Scatter (BVGAS)
- Partition-centric Processing Methodology

Definition

For page p, we define the following:

1. $pa(p)$ - Set of pages have out-links pointing to p
2. o_q - number of out-links in q
3. $0 < \beta < 1$ - Damping factor, probability of teleporting to a random page at any step. It is generally set around 0.85.
4. N - Number of of pages.

PageRank of page p, $PR(p)$, is giving by the following:

$$PR(p) = \beta \sum_{q \in pa(p)} \frac{PR(q)}{o_q} + \frac{(1-\beta)}{N}$$

PageRank in matrix form and Power iteration

Definition

A column stochastic matrix $\mathbf{M}$: (i.e. Each columns sum to 1)
Let page p_1 have o_i number of out-links.

$$\mathbf{M}_{p_2 p_1} = \begin{cases} \frac{1}{|o_i|} & \text{if there exists link from } p_1 \text{ to } p_2. \\ 0 & \text{otherwise.} \end{cases}$$

Score vector $\mathbf{r}$, that contain score for each page:
Let $\mathbf{A} = \beta \mathbf{M}$,

$$\mathbf{r}^{(t+1)} = \mathbf{A} \cdot \mathbf{r}^{(t)} + [\tfrac{1-\beta}{N}]_N$$

Power iteration:
We first set score for each page to $\frac{1}{N}$, $\mathbf{r}^{(0)} = [\frac{1}{N}, \ldots, \frac{1}{N}]^T$
We then iterate using this dynamic form equation above.
The iteration stops when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 <$ a small threshold.

Problems of PageRank

Problem

Spider traps :
All out-links are inside a group of pages. Eventually this trap will absorb all PageRank scores.
Solution is Random Teleport. At each iteration, using damping factor β as probability, it might follow out-link of the trap or it will teleport out of the trap.

Problem

Dangling page :
Pages with no out-links. PageRank scores will leak out of the system.
Solution is Always Teleport. We set the probability of random teleport to 1 for all dangling pages.

Properties and drawbacks of PageRank

Property

Sum of scores of all pages is equal to 1.

Property

PageRank computation is inexpensive.

Drawback

PageRank measure general popularity of a page.
A better solution : Topic-Sensitive PageRank.

Topic-Sensitive PageRank

Idea of Topic-Sensitive PageRank is that we bias the Random Walk from original PageRank. Allowing queries be more personal and of more interest to the user.

A topic-sensitive set of pages related to a topic is called Teleport Set, S.

Topic-Sensitive PageRank

Definition

Bias the Random Walk by updating the matrix **A** from original PageRank as following:

$$A_{ij} = \begin{cases} \beta M_{ij} + \frac{(1-\beta)}{|S|} & \text{if } i \in S \\ \beta M_{ij} & \text{otherwise.} \end{cases}$$

Weighting of pages in S is equal in the above equation. However, we can even give different weights to pages in S. We can then compute score using original PageRank for each teleport set S

Overall Algorithm

Algorithm

Preprocessing:
We will create set of topics. We can uses DMOZ(Open Directory Project) top-level categories.
For each set of topic t_i, we will evaluate PageRank ranking with respect to i^{th} topic.
Query-time processing:
Given a query q and context, we compute a Composite Link Score for a page d:
$s_d = \sum_i w_i r_i[d]$.

Binning with Vertex-centric GAS

BVGAS is a two phase algorithm, Scatter phase and Gather phase. The alogrithm scatter each update values from PageRank into bins to reduce the traffic in main memory. Then, gather phase accumulate the updates for evaluating the PageRank score.

Scattering phase of BVGAS

Algorithm

Let $m =$ number of nodes in a partition, P be set of partitions.
Initialize score of each page $= \frac{1}{N}$.
for *number of iteration* **do**
  foreach *page v in the graph* **do**
    $PR[v] = \frac{PR[v]}{|\text{out-link of } v|}$;
    foreach *page* $u \in$ *out-link of v* **do** insert $(PR[v], u)$ into $bins[\lceil \frac{u}{m} \rceil]$;
  end
  Set all PageRank score to 0.
  $\vdots$
end

Gathering phase of BVGAS

Algorithm

for *number of iteration* **do**

$\vdots$

foreach *partitions* $p \in P$ **do**

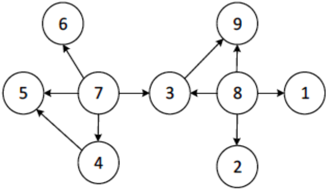
foreach $(update, dest)$ *in* $bins[p]$ **do** $PR[dest] += update$;

end

foreach *page* v **do** $PR[v] = \beta \times PR[v] + \frac{1-\beta}{N}$;

end

foreach *page* v **do** $PR[v] = PR[v] \times |\text{out-links of } v|$;



Partitions

P_1	P_2	P_3
1	4	7
2	5	8
3	6	9

(a) Example graph and node partitions

Bin 1

Updates	Dest. ID
PR[7]	3
PR[8]	1
PR[8]	2
PR[8]	3

Bin 2

Updates	Dest. ID
PR[4]	5
PR[7]	4
PR[7]	5
PR[7]	6

Bin 3

Updates	Dest. ID
PR[3]	9
PR[8]	9

(b) Bins store (PageRank, destination node ID) pairs [1]

[1] Kartik Lakhotia; Rajgopal Kannan; Viktor Prasanna. *Accelerating PageRank using Partition-Centric Processing*, arXiv sept 2017

Advantages and Drawbacks of BVGAS

Advantage

During Scatter phase, the size of partition is kept small therefore everything fits in the cache, the spatial locality[a] is thus improved. Also, during Gathering phase it process one bin at a time, the temporal locality[b] is then improved. Moreover, Gathering phase reads updates and destination ID squentially from a bin, so it also have good spatial locality.

[a]When reading a specific storage location, the nearby memory locations might also be used very soon. Therefore, we can prepare faster access to that particular location area.

[b]Special case of spatial locality, that is current memory location have the same data as the prospective location. Therefore, we can store that particular data for faster access.

Drawback

There are redundant update reads and writes.

Partition-centric Processing Methodology

PCPM separate the bin from BVGAS into two separate bins, *update_bins* and *destID_bins*. During the scattering of PCPM, we only store the update $PR[v]$ once.
PCPM changes the Most Significant Bit of IDs inside *destID_bins* to indicate the range of destination ID that use the same update values. The last destination ID using the same update values will have its MSB set to 1. The MSB will then be masked in order to generate the true identifier of destination.

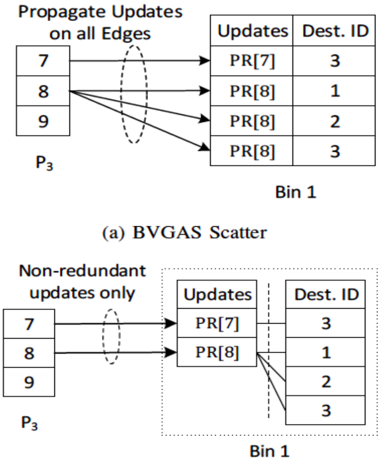
Scattering phase of PCPM

Algorithm

Let m = number of nodes in a partition, P be set of partitions.
Initialize score of each page $= \frac{1}{N}$.

for *partition* $p \in P$ **do**
    foreach *vertex* $v \in p$ **do**
        $prev\_bin = \infty$;
        foreach *vertex u pointed from v* **do**
            if $\lfloor \frac{u}{m} \rfloor \neq prev\_bin$ **then**
                insert $(PR[v])$ into $update\_bins[\lceil \frac{u}{m} \rceil]$;
                insert u to $destID\_bins[\lceil \frac{u}{m} \rceil]$, $prev\_bin = \lfloor \frac{u}{m} \rfloor$;
            else
                insert u to previous destination bin;
            end
        end
    end
end

Different between PCPM and BVGAS in binning



(a) BVGAS Scatter

(b) PCPM Scatter [2]

---

[2] Kartik Lakhotia; Rajgopal Kannan; Viktor Prasanna. *Accelerating PageRank using Partition-Centric Processing*, arXiv sept 2017

Gathering phase of PCPM

Algorithm

Initialize score of each page = 0.

for *partition* $p \in P$ **do**

  while $updateBins[p] \neq \emptyset$ **do**

    pop *update* from $update\_bins[p]$;

    pop *id* from $destID\_bins[p]$;

    while $MSB(id) \neq 1$ **do**

      $PR[id] = PR[id] + update$;

      pop *id* from $destID\_bins[p]$;

    end

    $id = id \& bitmask$;

    $PR[id] = PR[id] + update$;

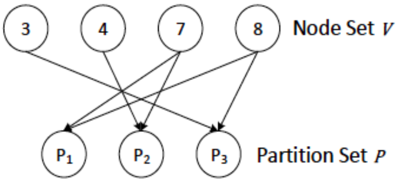
  end

end

foreach $v \in V$ **do** $PR[v] = \frac{\frac{1-d}{|V|} + d \times PR[v]}{|N_o(v)|}$;

Bipartite Partition-Node Graph

We only draw edges from node to partition set if that node point to one of the nodes in the partition set (Compression Step). This layout transform the original graph with less edges.



3

3 Kartik Lakhotia; Rajgopal Kannan; Viktor Prasanna. *Accelerating PageRank using Partition-Centric Processing*, arXiv sept 2017

PCPM Scattering Algorithm using PNG layout

Algorithm

Let PNG $G^{'}(P, V, E^{'})$, $N_{p_i}(p^{'})$ be in-neighbours of partition $p^{'}$ in bipartite graph of partition p

foreach $p \in P$ **do**
  foreach $p^{'} \in P$ **do**
    foreach $u \in N_{p_i}(p^{'})$ **do** insert $(PR[u])$ into $update\_bins[p^{'}]$;
  end
end

Branch Avoidance

This mechanism uses the modified identifiers of the *destID_bins*.
We will let *destID_ptr* and *update_ptr* be the pointers in *destID_bins*[p] and *update_bins*[p].
pop operator uses pointers that increment after reading each entry from respective bin.
Recall that **pop** operation is used everytime for *destID_bins* but **pop** operation is used for *update_bins* when $MSB(id) = 1$.

PCPM Gathering algorithm with Branch Avoidance

Algorithm

Let each score be 0.

foreach *partition* $p \in P$ **do**

$destID\_ptr = update\_ptr = 0$;

while $destID\_ptr < size(destID\_bins[p])$ **do**

id = destID_bins[p][destID_ptr++];

update = update_bins[p][update_ptr];

update_ptr = update_ptr + MSB(id);

PR[id & bitmask] = PR[id & bitmask] + update;

end

end

foreach $v \in V$ **do** $PR[v] = \frac{\frac{1-d}{|V|} + d \times PR[v]}{|N_o(v)|}$;

Comparison with BVGAS and PCPM

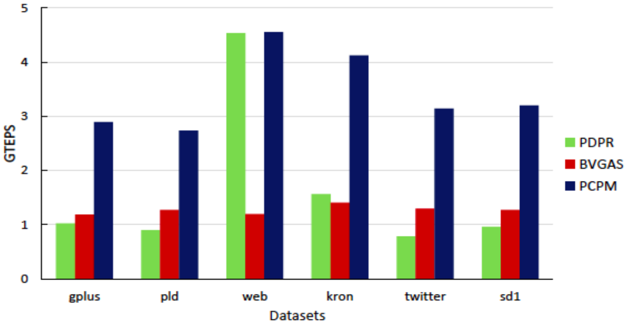
System of experimental setup : Intel Xeon E5-2650 v2 Ivy-bridge, 16 cores. Running Ubuntu 14.04 operating system.

Dataset :

Dataset	Description	# of Nodes(M)	# of Edges(M)	Degree
gplus	Google Plus	28.94	462.99	16
pld	Pay-Level-Domain	42.89	6 23.06	14.53
web	Webbase-2001	118.14	992.84	8.4
kron	Synthetic graph	33.5	1047.93	31.28
twitter	Follower network	61.58	1468.36	23.84
sd1	Subdomain graph	94.95	1937.49	20.4

Result

Uses GTEPS as performance evaluation metric. It computed as ratio of giga edges in the graph to runtime of single PageRank iteration. PCPM is faster than BVGAS by at least 2.1× and at most 3.8×.

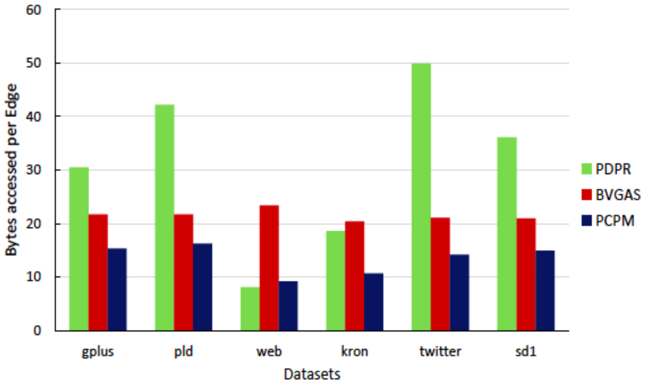


[4]

[4] Kartik Lakhotia; Rajgopal Kannan; Viktor Prasanna. *Accelerating PageRank using Partition-Centric Processing*, arXiv sept 2017

Result

From the following graph, we can see the average communication of PCPM is 1.7× less than BVGAS.



[5]

[5] Kartik Lakhotia; Rajgopal Kannan; Viktor Prasanna. *Accelerating PageRank using Partition-Centric Processing*, arXiv sept 2017

Conclusion

- Topic-sensitive PageRank gives result more related to query's context.
- BVGAS is more efficient than the Pull Direction PageRank.
- PCPM gives reduction in both excution time and DRAM communication than BVGAS. It reduce redundant reads and writes to reduce main memory traffic.