International Journal of Engineering Science Invention (IJESI)
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org ||Volume 7 Issue 2|| February 2018 || PP. 01-11

Mining Structural Traits of Hyperlinks to Combat Spamdexing

Dr.S.Sasikala[1], Ms.P.Deepika[2], Ms.S.Saranya[2], Mr.S.Balaji[3]
[1](Asso.Prof, BCA Department, Hindusthan College of Arts and Science, India)
[2](Asst.Prof, BCA Department, Hindusthan College of Arts and Science, India)
[3](Asst.Prof, CSE Department, Mahendra Engineering College, India)
Corresponding Author: Dr.S.Sasikala

ABSTRACT: *The prevalence of link spam loots the search results quality in considerable manner. Graph based methods potentially downgrade the link spam in batch demotion mode. Labeling the nodes as either trusted or distrusted is carried out after analyzing the link (inlink and outlink) flow. The label will be propagated subsequently to the neighbors. Propagation of value (either trust or distrust) from seed set to other nodes is a well adopted maneuver in graph based link spamdexing detection methods. This paper expound ASD_PTDV (Aggregating Seed Discrimination for Propagating Trust and Distrust Value) algorithm for demoting the Web link spam.*

KEYWORDS *Web Graph, Search Engine, Link Spamdexing, Trust Propagation, ASD_PTDV*

Date of Submission: 21-01-2018 *Date of acceptance: 05-02-2018*

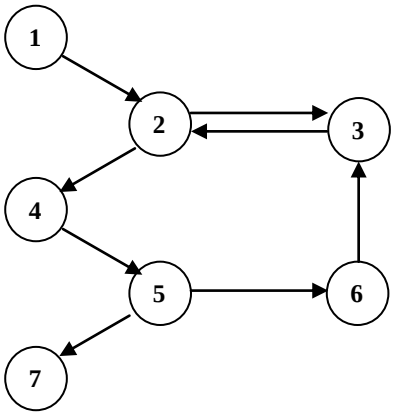
I. Introduction

The Website can be viewed as a graph G = (V, E), where V is the vertices (set of Web pages) and E is the edges (set of hyperlinks). An edge (u, v) ∈ E, iff a page u links to the page v. The inlink to a page, P is referred as E_{IN} and outlink is termed as E_{OUT}.

The total number of inlinks to a page is known indegree, deg^{in} and the total number of outlinks from a page is the outdegree, deg^{out} . The transition matrix for the Webgraph is constructed by assigning values as follows:

$$tm(p, q) = \begin{cases} 0 & \text{if } (q, p) \notin E \\ 1/deg^{out} & \text{if } (q, p) \in E \end{cases}$$

Inverse transition matrix is another important form used in the calculation of the rank in link based ranking algorithms. The inverse transition matrix can be formulated with the strategy "reverse the links". Consider the Webgraph with 7 vertices and 8 edges in Figure 1.



V = {1, 2, 3, 4, 5, 6, 7}
E = {(1, 2), (2, 3), (2, 4), (3, 2), (4, 5), (5, 6), (5, 7), (6, 3)}

Figure 1: Sample Webgraph

Reversing the links the set of edges obtained for the Webgraph in Figure 1 is as follows:

$$E^{-1} = \{(2,1), (3,2), (4,2), (2,3), (5,4), (6,5), (7,5), (3,6)\}$$

The tm and itm are used for depicting the Web graph in a simple manner. This representation helps the link based ranking algorithms for obtaining the structure of the Website. The tm and itm representations are used in the subsequent sections of the work.

II. Related Work

Gyongyi et. al. compare PageRank with TrustRank and concluded that TrustRank performs better than the PageRank. Krishnan and Raj compared Anti-TrustRank with TrustRank and concluded that performance hiked slightly.Krishnan and Raj suggest that the combination of TrustRank and Anti-TrustRank with significant heuristics would yield better performance in fighting the Web spam. This aspect tends to be the motivation behind this work.

Wu et. al. (Wu et.al 2006) proposed the idea of combining the trust and distrust for fighting Web spam. According to them, parent trust score is divided by the number of its outgoing links and each of its children gets an equal share. The child trust score is the sum of shares from all its parents. Two observations are focused on the aforesaid basic idea. First one is, for each parent how to divide its score among its children is termed as splitting step. The other one, how to calculate the overall scores when nodes have the shares from all its parents is accumulation step. For splitting step they provide three choices.

- Equal splitting: a node i with O(i) outgoing links and trust score TR(i) will give $d \,.\frac{TR(i)}{O\,(i)}$ to each child.

d is constant 0 < d <1

- Constant Splitting: a node i with trust score TR(i) will give *d.TR(i)* to each child
- Logarithm Splitting: a node i with O(i) outgoing links and trust score TR(i) will give $d \,.\frac{TR\,(i)}{\log(1+o(i))}$ to each child, where d is the decay factor which determines how much of parents score is propagated to its children

Equal splitting is employed in TrustRank. For accumulation step, choices were offered by the authors.

- Simple Summation: sum the trust values from each parent
- Maximum Share: use the maximum of the trust values sent by the parents
- Maximum Parent: value that never exceed the trust score of the most trusted parent

The simple summation is applied in the PageRank and TrustRank. Using "Constant Splitting" and "Simple Summation", trust score can be calculated using

$$t = (1-\alpha)\,.\,d\,.\,M^{T}.\,t + \alpha\,.\,s \qquad (7)$$

where t is the trust score vector, α is jump probability d is the constant in splitting choices, M is the Web matrix and s is the normalized trust score vector for seed set. Trust score indicates the likelihood of a page not being spam. Distrust score indicates the probability of a page being spam.

Trust flows from parent to child and distrust flows in the reverse manner. Spam sites are used as seed set in distrust propagation and penalizing sites which are linked to the spam sites is the basic idea here. The same idea of splitting and accumulation steps is adopted for distrust propagation as follows:

- Equal Splitting: node i with I(i) incoming links and DISTR(i) will give $d_D \,.\frac{DISTR\,(i)}{I\,(i)}$ to each parent

where $0 < d_D < 1$

- Constant Splitting: a node i with DISTR(i) will give *d.DISTR(i)* to each parent
- Logarithm Splitting: a node i with I(i) incoming links and DISTR(i) will give $d_D.\frac{DISTR\,(i)}{\log(1+I(i)}$ to each parent

For accumulation step, the following choices were offered by the authors.

- Simple Summation: sum the distrust values from each child
- Maximum Share: use the minimum of distrust values sent by the children
- Maximum Parent: sum the distrust values in such a way as to never exceed the distrust score of most distrusted child

For constant splitting and simple summation the following equation is used for distrust score calculation.

$$n = (1-\alpha)\,.\,d_D\,.\,M.n + \alpha\,.\,\pi \qquad (8)$$

where n is distrust score vector, α is jump probability, d is the constant, m is the Web matrix and r is the normalized distrust score vector. Authors combine trust and distrust scores to generate a qualified ranking of pages that is indicating of their trustworthiness. They simply calculate the difference of these two scores and use this value to represent the overall trust worthiness of the Webpage.

$$Total(i) = \eta\,.\,TR(i) - \beta\,.\,DISTR(i) \qquad (9)$$

where η and β ($0 < \eta < 1$, $0 < \beta < 1$) are two coefficients to give different weights to trust and distrust scores. They conducted experiments on datasets and concluded that choices such as "Constant Splitting" or "Logarithm Splitting" in splitting step and "Maximum Parent" in accumulation step for propagating trust can help to demote top ranked spam sites as well as increase the range of trust propagation.

Two other variants of combining trust and distrust is proposed by Wu et. al. and Nie et. al. The combination of trust and distrust again leads to trust score calculation by these two authors. They consolidate the trust and distrust score and finally calculate the total score leading to identification of trustworthiness of a

used page or site. ASP_PTDV algorithm proposed in this research calculates the trust and distrust value in different manner. It doesn't combines the trust and distrust score. The purposes of these two scores are entirely different and if used separately it may give good results along with proper insights retrospectively.

III. Proposed Algorithm

The first paragraph under each heading or subheading should be flush left, and subsequent paragraphs should have a five-space indentation. A colon is inserted before an equation is presented, but there is no punctuation following the equation. All equations are numbered and referred to in the text solely by a number enclosed in a round bracket (i.e., (3) reads as "equation 3"). Ensure that any miscellaneous numbering system you use in your paper cannot be confused with a reference [4] or an equation (3) designation.

TrustRank, Anti-TrustRank and its derivatives has an important concern; the knowledge of either good seeds or bad seeds alone is used in the algorithm. They propagate either trust or distrust regardless of the knowledge of propagated nodes as illustrated in Figure 3.3. The issue addressed above gives two key insights. First one is utilizing either good seed or bad seed alone miss out the valuable information of other kind which may give better results. Second one is considering the grey scale cases mentioned in Figure 3 may yield more accurate results.

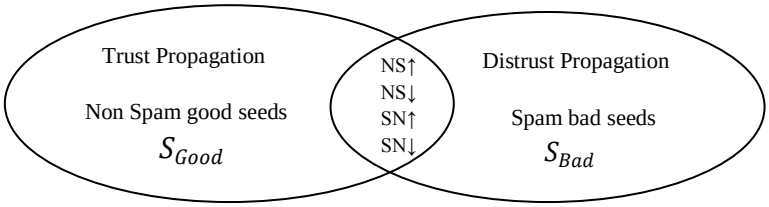


Figure 3: Propagation Coverage

Four cases are considered as grey scale nodes depicted in Figure 3. They are:

- Case 1: NS↑ - Non-Spam pointing to Spam
- Case 2: NS↓ - Non-Spam pointed by Spam
- Case 3: SN↑ - Spam pointing to Non-Spam
- Case 4: SN↓ - Spam pointed by Non-Spam

Case 1 improves the trust score of spam page whereas case 3 decreases the score of non-spam page. Case 2 improves the distrust score of non-spam page and case 4 decreases the distrust score of spam page. In order to rule out these pitfalls and achieve better results the aggregation strategy based on seed discrimination is proposed in this research. Good seeds and bad seeds are segregated initially based on the inverse PageRank (iPR) values. ASD_PTDV algorithm is designed with three Primary motivations.

- Utilizing the knowledge of both good and bad seeds may lead to better results
- Addressing the grey-scale nodes (neither good nor bad) in an effective manner
- Propagating trust and distrust to nodes based on the inferences gained from seed discrimination

These three motivations lead the ASD_PTDV algorithm, which are well explained in the subsequent sections.

3.2 ASD_PTDV Algorithm

The ASD_PTDV algorithm involves the following six phases in its total process:

- *Phase 1:*Seed set selection is done for spam and non–spam nodes based on inverse PageRank (Spam Seed set - pages with least iPR scores and Good Seed set pages with top iPR scores)
- *Phase 2:* Oracle value is assigned for trust and distrust vector (TV and DV), where knowledge of spam and non–spam seeds are known to both TV and DV (In TV, goods seeds =1 and spam seeds =0 and in DV, spam seeds =1 and good seeds =0)
- *Phase 3:* Unevaluated node assessment is carried out based on known spam and non–spam seeds (Unevaluated nodes = N – (spam seeds + non–spam seeds))
- *Phase 4:* ASD_TValue and ASD_DValue are calculated for unevaluated nodes
- *Phase 5:* Line normalized vectors: NDV and NTV are created
- *Phase 6:* ASD_TRank and ASD_DRank are calculated and trust/distrust is propagated

After the six phases, ASD_TRank and ASD_DRank values are compared. When a node shows high ASD_TRank, it will have relatively low ASD_DRank and it will be non–spam node and hence trust will be

propagated. When a node shows high ASD_DRank, it will have relatively low ASD_TRank and it is spam node and distrust is propagated. The algorithm has these insights in its core and designed with meticulous effort to overcome the potential weaknesses of the existing algorithms. The ASD_PTDV algorithm is presented in the subsequent part of this section.

Table I: ASD_PTDV Algorithm

ASD_PTDV: Aggregating Seed Discrimination in Propagating Trust and Distrust Value

Input:

T – Transition Matrix
R – Reverse Transition Matrix
TV – Trust Vector
DV – Distrust Vector
N – Number of nodes
L1, L2 – Limit of Oracle invocations
α – decay factor
M – Number of Iterations
E_{IN}^{S} – Inlink from spam page
E_{IN}^{N} – Inlink from non spam page
E_{IN}^{U} – Inlink from unevaluated page
E_{OUT}^{S} – Outlink to spam page
E_{OUT}^{N} – Outlink to non-spam page

E_{OUT}^{U} – Outlink to unevaluated page

Output:

ASD_TRank – Trust propagation value
ASD_DRank – Distrust propagation value

Steps:

Begin

(1) S_{Good} = SelectSeed()
$\delta = \text{DORank}(\{1 \ldots N\}, s)$
$TV = 0_N$
for i = 1 to L1 do
if $(O(\delta(i)) == 1)$ then
$TV(\delta(i)) = 1$
end if
end for

(2) S_{Spam} = SelectSeed()
$\beta = \text{IORank}(\{1 \ldots N\}, s)$
$DV = 0_N$
for i = 1 to L2 do
if $(O(\beta(i)) == 1)$ then
$DV(\beta(i)) = 1$
$DV(\propto(i)) = 0$
end if
end for

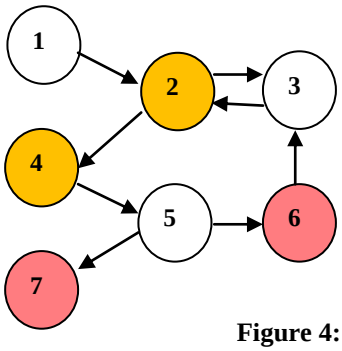
(3) for i = 1 to L1 do
$TV(\beta(i)) = 1$
end for

(4) for i = 1 to N do
if $(\text{cur}(i) \text{ not belongs to } \propto) \cup (\text{cur}(i) \text{ not belongs to } \beta)$ then

$$\text{ASD\_Tvalue}(\text{cur}(i)) = \frac{\sum_{q:q\to p} \text{ASD\_Tvalue}(q)}{\sum E_{IN}^{N} + \sum E_{IN}^{U}}$$

$$\text{ASD\_Dvalue}(\text{cur}(i)) = \frac{\sum_{q:p\to q} \text{ASD\_Dvalue}(q)}{\sum E_{OUT}^{S} + \sum E_{OUT}^{U}}$$

```
            end if
        end for
(5)     update TV and DV // Values of unevaluated nodes are updated in both trust and distrust vector
(6)     NTV = TV/|TV|
(7)     NDV = DV/|DV|
(8)     ASD_TRank_0 = NTV
        while (k ≤ M) do
        ASD_TRank_k = ∝ . (ASD_TRank_k−1 . T) + (1−∝). NTV
(9)     ASD_DRank_0 = NDV
        while (j ≤ M) do
        ASD_DRank_k = ∝ . (ASD_DRank_k−1 . T) + (1−∝). NDV
(10)    return ASD_TRank
                and ASD_DRank
```



V = {1, 2, 3, 4, 5, 6, 7}

E = {(1, 2), (2, 3), (2, 4), (3, 2), (4, 5), (5, 6), (5, 7), (6, 3)}

Good Seed = {2, 4} and Bad Seed = {6, 7}

Figure 4: Exemplification Sample for ASD_PTDV

The ASD_PTDV addresses the four cases mentioned in Figure 4. The unevaluated nodes are assigned a trust and distrust score based on the proposed metric. Both seed sets are updated with the appropriate scores. The seed set is normalized to get NTV (Normalized Trust Vector) and NDV (Normalized Distrust Vector). Good seeds know the bad seeds and bad seeds know the good seeds. The explained strategy is adopted in ASD_PTDV algorithm. The unevaluated nodes 1, 3 and 5 are also assigned value based on the proposed metric.

I. PROPAGATION STRATEGY

The normalized trust vector and distrust vector are used for the rank calculation. The trust-distrust rank is computed based on the biased PageRank versions. The steps involved in the ASD_TRank are as follows:

$ASD\_TRank_0 = NTV$

While ($K \leq M$) do

$ASD\_TRank_k = \alpha \,.\, (ASD\_TRank_{K-1} \,.\, tm) + (1 - \alpha) \,.\, NTV$

The initial value for the ASD_TRank is assigned as NTV. It is evenly spread or propagated in all iterations with (1-α). NTV is constant for all iterations; similarly the ASD_DRank is calculated and propagated with the following steps.

$ASD\_DRank_0 = NDV$

While ($K \leq M$) do

$ASD\_DRank_k = \alpha \,.\, (ASD\_DRank_{K-1} \,.\, tm) + (1 - \alpha) \,.\, NDV$

where α is the decay factor set to 0.85 and tm is the transition matrix. The distrust values in NDV are evenly propagated to all the nodes through iterations. The iterations are terminated after convergence or executed for M, Number of times. The results of the ASD_TRank and ASD_DRank are compared. It tends to show the following observations:

- Page with high ASD_TRank shows relatively low ASD_DRank and vice-versa
- Good page possess high ASD_TRank and low ASD_DRank
- Bad or spam page possess high ASD_DRank and low ASD_TRank
- Unreferenced node (node which has no inlink) shows low ASD_TRank, even if it is good page
- Non-referenced node (node which has no outlinks) will show low ASD_DRank even if it is good page
- High ASD_DRank shows the genuineness of page and high ASD_DRank shows the spamness of a page

Last two observations can be addressed by the comparison of node structure and ranks (ASD_TRank and ASD_DRank). Since trust flows from parent to child and distrust flows from child to parent, this happens.

II. Implementation

The ASD_PTDV algorithm source code is implemented as two modules. First module is seedset selection. Second module calculates the Trust scores, ASD_TRank and distrust scores, ASD_DRank. The program receives the initial trust and distrust score vector as input with seed discrimination. Transition matrix of the Webgraph is given as another input.

The outputs are ASD_TRank and ASD_DRank scores. The code works iteratively to calculate the trust-distrust values with the Jacobi iterative method perception to solve the problem. Experiments are executed on a machine with 2 dual-core 2.33 GHz Pentium IV processors with 4 GB memory. The algorithm is tested in two different datasets. The first dataset is composed from a manually compiled 300 real time samples. It is collected from search results of spammers targeted query "online earning and easy money". Top 300 results were collected from Google search engine. Initially a set of 300 samples with 1860 nodes were evaluated for the proposed ASD_PTDV algorithm. Manual assessment is performed to classify the samples. Among them, 38 samples were spam and 262 were non-spam. The inverse PageRank algorithm is used for the good seed and spam seed selection which is discussed. Based on that, the initial trust and distrust vectors are created. ASD_TRank and ASD_DRank are calculated for the nodes.

Later standard benchmark WEBSPAM-UK2007 dataset (Castillo et al. 2006) with link based features has been used for the ASD_PTDV experiment further. It is a publicly available collection of web pages. The dataset is particularly suited for evaluating machine learning methods for Web spam detection, since it is large and it comes with targets and pre-computed features. It is based on a set of pages obtained from a crawler of the .uk domain. The set includes 77.9 million pages, corresponding to 11402 hosts, among which over 8000 hosts have been labelled as spam, non-spam or borderline. The link based feature set contains originally 3998 instances with 44 attributes. The dataset is divided into 20 buckets with equal number of samples in all buckets based their corresponding PageRank scores. Buckets are indexed with number 1 to 20. The top 50 samples are selected from each bucket as the seeds.

The results of ASD_TRank and ASD_DRank are compared. If a page has high ASD_TRank and low ASD_DRank, it is concluded as trusted page and vice versa. When ASD_TRank equals ASD_DRank then E_{IN} and E_{OUT} are analyzed for its trustworthiness. Based on that, trust or distrust is adopted as follows.

For webpage ,w = ASD_TRank > ASD_DRank: Propagate trust via E_{IN}
ASD_TRank < ASD_DRank: Propagate distrust via E_{OUT}
ASD_TRank = ASD_DRank: Analyze E_{IN} and E_{OUT} for trust and adopt the splitting strategy

The third case doesn't occur often. The possible situation for the case is, when computing score for non-referencing/non-referenced (node with no outlink/no inlink) node. Then analyzing either inlink or outlink may give good insights for demarcating the Webpage.

4.1 Exemplification of ASD_PTDV

The iteration wise result of TrustRank and ASD_TRank for the illustrative sample is presented in Table III, IV, V and VI respectively. The ASD_TRank shows that the result converges from iteration 7 and terminated at iteration 10. In TrustRank, the result converges from iteration 16 onwards. Thus the number of iterations is reduced in ASD_TRank and it leads to the time consumption in computation.

The screenshots of the implemented algorithm is listed in the Figure 5 to 10. Initially the transition matrix is given as the input to the seed set selection module. Later, based on the seeds (Good and Bad) the trust/distrust value is created. Further the propagation is performed with the node assessment.

III. Results

A sample is exemplified for the experiment throughout this paper. Initially the number of iterations is set to 20 for both base and proposed algorithms. After experimenting 40% of samples, it is decided to set the iteration number to 10. Again all the samples are executed with 10 iteration count for ASD_TRank. 21.6% of the samples needs additional iterations to result convergence and again executed with 15 iteration count. ASD_TRank converges in lower number of iterations than the base algorithm for the experimented samples. Handling non-referencing and non-referenced nodes creates the misnomers in results. The time efficiency is achieved and the approach adopted in the proposed ASD_TRank tends to show better results in less number of iterations. Both algorithms rely on the approximate isolation logic and the level of seed usage differs in them. Anti-TrustRank uses the node level seed set and ASD_DRank use the subgraph level seed set. Iteration wise results of TrustRank, ASD_TRank and ASD_DRank are given in Table III, IV, V and VI.

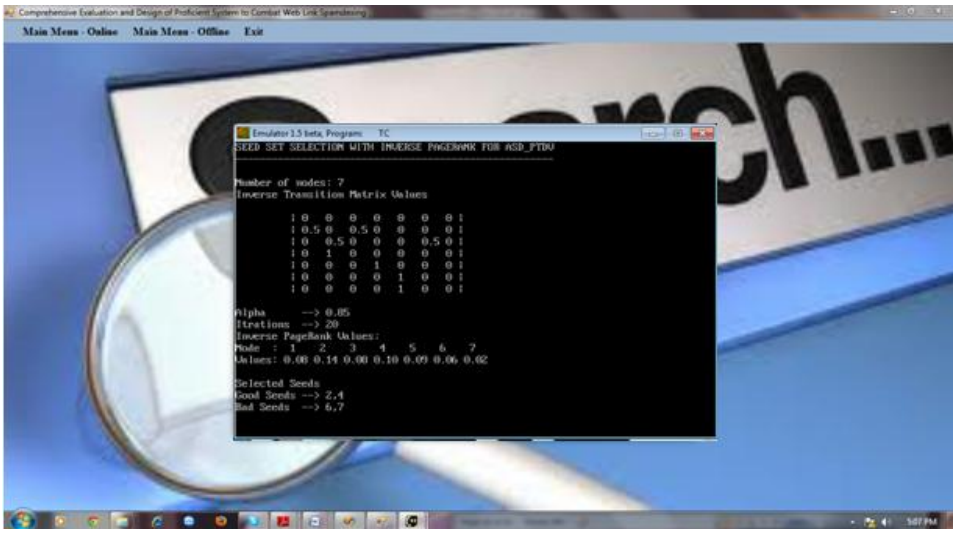


Figure 5: Seed Set Selection with ASD_PTDV

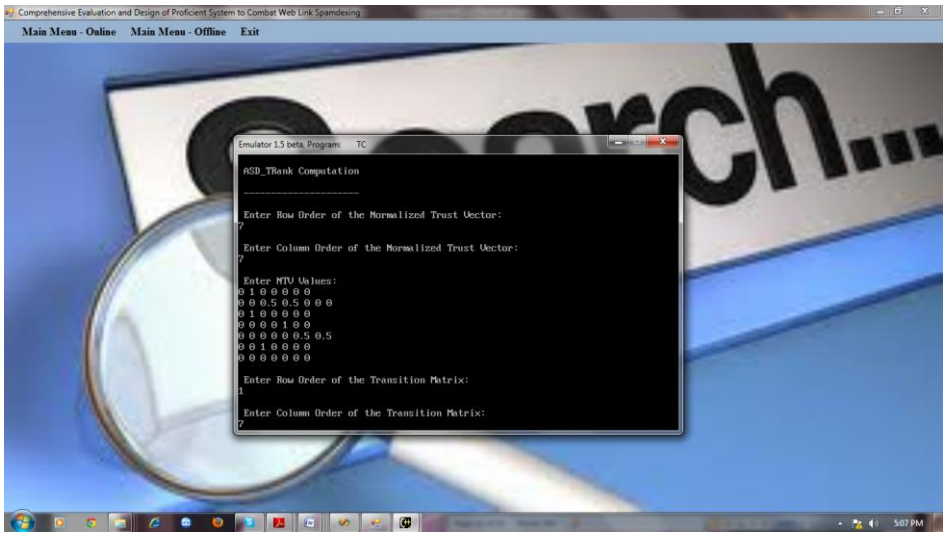
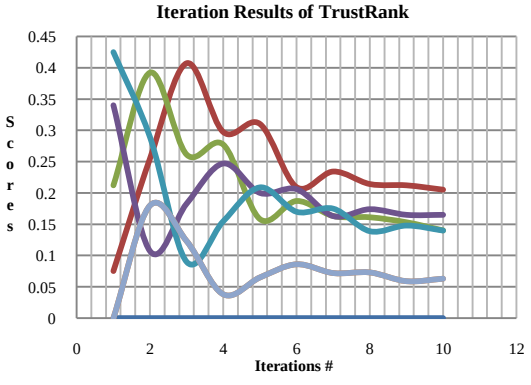
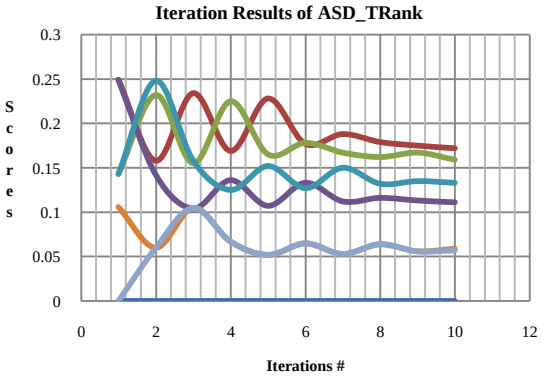


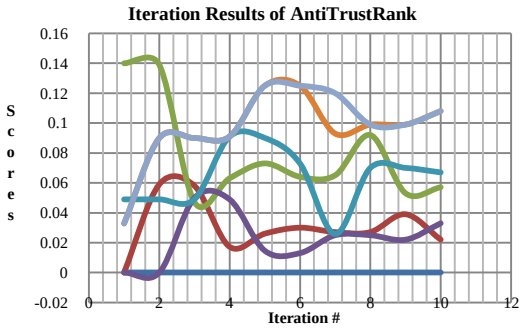
Figure 6: Input Parameter Specification for computing ASD_TRank



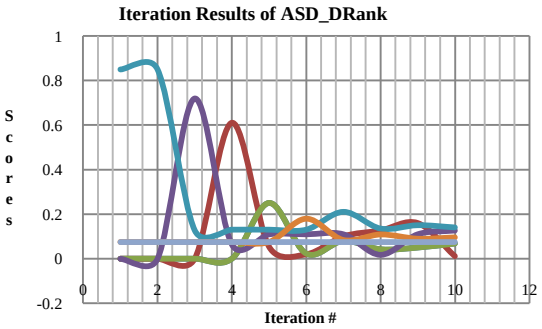
(a)TrustRank values on Iterations



(b) ASD_TRank values on Iterations



(c)Anti-TrustRank values on Iterations



(d)ASD_DRank values on Iterations

Figure 11: Iteration Results Comparison in Base and Proposed Algorithms

Figure 11(a, b, c and d) indicates the iteration wise results comparison of the baseline and the proposed algorithms. In baseline algorithms the M is downsized to 10, in order to show deviation. It is evident from Figure 11 that the proposed algorithms tend to show a gradual convergence from mid-iterations. The ASD_TValue calculation for unevaluated nodes is the task need to be accomplished before the iteration calculations for nodes.

TrustRank uses the node level seed and ASD_TRank use the sub graph level seed for creating the initial trust vector. The same aforesaid scenario is applicable to the Anti-TrustRank vs. ASD_DRank. The Anti-TrustRank results starts converging from 14^{th} iteration onwards whereas the ASD_DRank results maximum nodes converges from 7^{th} iteration onwards. The termination iteration is set to M = 10, ASD_DRank shows better results on 10^{th} iteration itself.

IV. Performance Comparison

Nodes which are manually evaluated kept as a base values to compare. It is verified against the result achieved from the proposed algorithm. The comparison of Precision, Recall and Accuracy for the proposed algorithm is given in Table VII. ASD_TRank is compared with TrustRank and ASD_DRank is compared with Anti-TrustRank. The Precision, Recall, Accuracy and F-Measure are used to assess the performance.

Precision = TP/(TP+FP) (11)

Recall = TP/(TP+FN) (12)

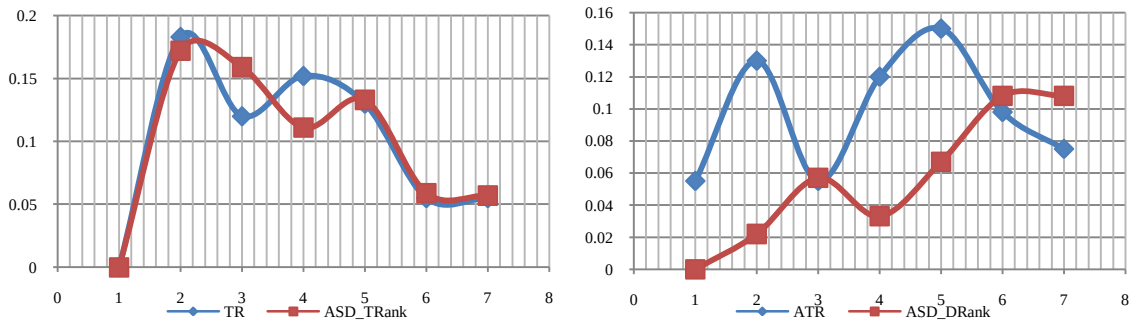
Accuracy = (TP+TN)/(TP+FP+TN+FN) (13)

F-Score = 2.(Precision. Recall)/(Precision+ Recall) (14)

The acronyms used the above metrics are: TP-True Positive, TN-True Negative, FP-False Positive and FN-False Negative. The values of these metrics for base and proposed methods are given in Table VII. Accuracy improves by 4% in ASD_TRank and 1% in ASD_DRank as seen from Table VII. Node results value of the base and proposed algorithm is given Figure 12. It is evident from the figure that bad nodes 6 and 7 have low trust score with high distrust score. For good nodes 2 and 4 ASD_TRank values are higher with lower ASD_DRank values. Node 1 is non-referenced node which possesses low scores for both. The proposed strategy combines the goodness of the TrustRank and Anti-TrustRank and overcomes the potential drawbacks in both algorithms. The grayscale nodes are effectively addressed in ASD_PTDV algorithm. Results are promising in the proposed algorithm.

Table VII: Performance Comparison of the ASD_PTDV

	Precision	**eRecall**	**Accuracy**
ASD_TRank	0.821	0.658	0.800
TrustRank	0.790	0.553	0.760
\ASD_DRank	0.842	0.800	0.837
Anti-TrustRank	0.851	0.632	0.823



(a) TrustRank and ASD_TRank (b) Anti-TrustRank and ASD_DRank

Figure 12: Node Result Comparison

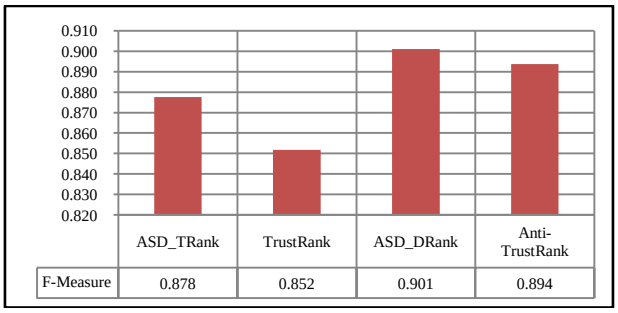


Figure 13: F-Score Comparison of the Base and ASD_PTDV

Results of WEBSPAM-UK2007 datasets are evaluated in two different aspects (Gyongyi et al. 2004). First one is number of spam sites in each bucket. The spam sites must be lower in count in low-indexed buckets and higher number of spam sites must be placed in high-indexed buckets. This assumption may withhold with the ASD_TRank which enhances the TrustRank. For ASD_DRank the assumption would be higher number of genuine sites must be placed in high indexed buckets. In ASD_TRank 11, 13 and 15 buckets have lower spam sites indexed compared to TrustRank. Bucket index after 15 hold higher number of spam sites when compared with the TrustRank (Figure 14). The ASD_TRank performance seems to be improved than TrustRank and PageRank. For ASD_DRank buckets from 11 hold high number of genuine sites compared to Anti-TrustRank and inverse PageRank. The second aspect is total number of spam sites in top-N buckets. If there is lower number of spam sites in top-N buckets, then it means good efficiency. As witnessed from the Figure 15, it is clear that 7, 10, 11, 13 and 18 buckets seems to have lower number of spam sites when compared to other buckets, which have comparatively good performance than the baseline. ASD_TRank address the spam sites in effective manner wheras ASD_DRank should filter the non-spam sites in effective manner. Since in computing the scores, ASD_TRank gives scores to genuine sites and as a consequence spam sites will have low scores and the overturn process is carried out in ASD_DRank. From the Figure 15, it is clear that from bucket 7 onwards the genuine sites are indexed lower in ASD_DRank.

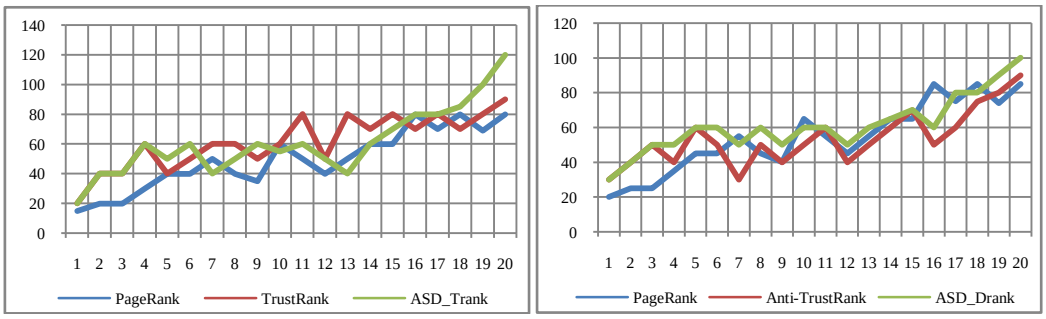


Figure 14: Spam sites in each bucket on WEBSPAM-UK2007

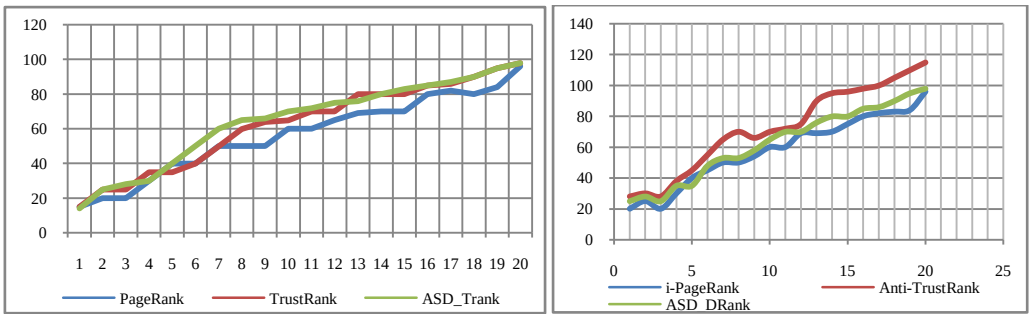


Figure 15: Spam sites in top-N buckets on WEBSPAM-UK2007

The precision value for the WEBSPAM-UK 2007 dataset for the threshold 20 has been given in the Figure 16. It shows a clear inference that the precision seems to be better for the ASD_PDTV comparatively. As a conclusion, the ASD_PDTV seems to be apt for the spamdexing detection.

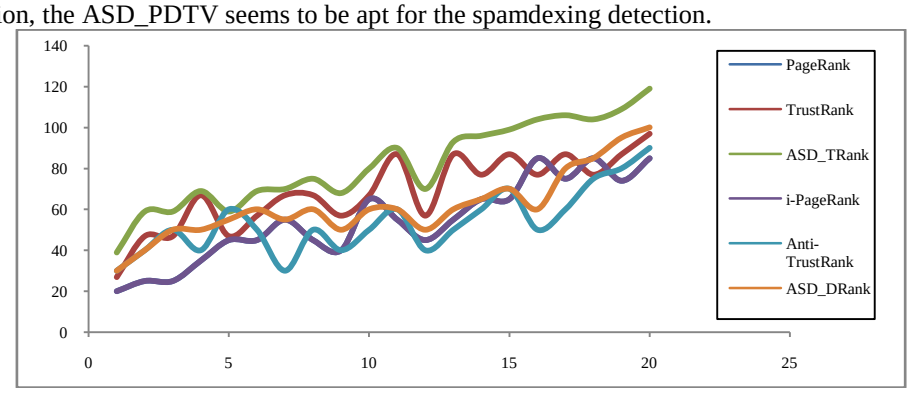


Figure 16: Precision of the WEBSPAM-UK 2007 dataset in different propagation algorithms

V. Computational Complexity

Assume Webgraph G with V, vertices and E, edges. Seed set selection is performed with the inverse PageRank which cost $\mathcal{O}(V+E)$ time, the seeds are moved to the propagation phase, where it requires analysis of all inlinks and outlinks which cost $\mathcal{O}(V+E)$ time. It is evident from the experiments elucidated above the proposed algorithm significantly improves the performance. Convergence of results is achieved in half the number of iterations when compared with base algorithms. This offers noteworthy space complexity for the ASD_PTDV algorithm. In WEBSPAM-UK 2007 dataset, the ASD_PTDV perform well when compared with the other algorithms. The strategy adopted in ASD_PTDV seems to be comparatively good for the spamdexing detection.

VI. Summary And Conclusion

Spam detection and demotion seems to be digital warfare going on for a long while. This paper introduces an algorithm ASD_PTDV which propagates both trust-distrust to nodes. By setting α = 0.85 and good seeds = {2, 4}and bad seeds = {6,7} for the Webgraph in Figure 3.4, the following results are achieved.

ASD_TRank = [0 0.17 0.16 0.11 0.13 0.06 0.06]
ASD_DRank = [0 0.02 0.05 0.03 0.07 0.10 0.11]

When observing the results, it is clearly visible that good nodes have high ASD_TRank and relatively low ASD_DRank. Bad or spam nodes have high ASD_DRank and relatively low ASD_TRank. The ASD_TRank and ASD_DRank computation converges in M = 10 for 78.3% of samples, which is very time consuming when compared with TrustRank and Anti-TrustRank. Remaining 21.66% samples has the nodes either with no inlinks or no outlinks which makes the algorithm feel difficult to understand the nature of nodes connected with them. As a whole, this ASD_PTDV algorithm performs well when compared with the existing algorithms in terms of efficiency.

References

[1]. ***Gyongyi Z, Garcia-Molina H and Pedersen J, 2004,*** *"Combating web spam with TrustRank", VLDB'2004: 30th International conference on Very large data bases, VLDB Endowment pp: 576–587*

[2]. ***Haveliwala T H, 2002,*** *"Topic Sensitive PagRank", 11th International Conference on World Wide Web, ACM, New York, pp: 517-526*

[3]. ***Jeh and widom, 2002,*** *"SimRank: A Measure of Structural-Context similarity", 8th International conference on knowledge discovery and data mining (SIGKDD), pp:538-543*

[4]. ***Krishnan V and Raj R, 2006,*** *"Web spam detection with Anti-TrustRank", 2nd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb), USA, pp: 37-40*

[5]. ***Liang C, Ru L, Zhu X, 2006,*** *"R-SpamRank: A spam detection algorithm based on link analysis", Journal of Computational Information Systems, pp:1705-1712*

[6]. ***Page L, Brin S, Motwani R, and Winograd T, 1998,*** *"The PageRank citation ranking: bringing order to the web", Technical report, Stanford Digital Library Technologies Project*

[7]. ***Qi X, Nie L and Davison B D, 2007, "****Measuring Similarity to Detect Qualified Links, AIRWeb '07, Banff, Alberta, Canada, pp: 49-56*

[8]. ***Qi, C, Song-Nian Y, Sisi C, 2008,*** *"Link Variable TrustRank for Fighting Web Spam" International Conference on Computer Science and Software Engineering, Wuhan, China, pp: 1004-1007*

[9]. ***Wu B, Goel V, and Davison B D, 2006,*** *"Topical TrustRank: Using topicality to combat web spam", 15th International World Wide Web Conference, Edinburgh, Scotland, pp: 63–72(May)*

[10]. ***Yang, Haixuan, King I and Michael R, Lyu, 2007,*** *"Diffusionrank: A Possible Penicillin for Web Spamming", 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, Netherlands, New York*

[11]. ***Zhang L, Zhang Y, Zhang Y and Li X, 2006****, "Exploring both content and link quality for anti-spamming", Sixth IEEE International Conference on Computer and Information Technology, pp. 37-37*

[12]. ***Zhou B and Pei J, 2009****, "Link spam target detection using page farms", ACM Transactions on Knowledge Discovery Data*

International Journal of Engineering Science Invention (IJESI) is UGC approved Journal with Sl. No. 3822, Journal no. 43302.

Dr.S.Sasikala "MINING STRUCTURAL TRAITS OF HYPERLINKS TO COMBAT SPAMDEXING" International Journal of Engineering Science Invention (IJESI), vol. 07, no. 01, 2018, pp. 01–11.