Cascading Citations Indexing Framework algorithm implementation and testing

Eleni Fragkiadaki, Georgios Evangelidis, Nikolaos Samaras
Dept. of Applied Informatics
University of Macedonia
Thessaloniki, Greece 54006
Email: eleni.fra@gmail.com, {gevan,samaras}@uom.gr

Dimitris A.Dervos
Information Technology Dept.
ATEI of Thessaloniki
Sindos, Greece 57400
Email: dad@it.teithe.gr

***Abstract*—The Cascading Citations Indexing Framework has been proposed as a new method for calculating the impact a researcher's work has. The algorithm presented in this paper calculates the total number of citations a distinct (article, author) pair has received, taking into consideration not only the direct citations but also the indirect ones. The algorithm is tested against the CiteSeer bibliographic database. The results presented demonstrate the different ranking of the pairs based on the depth at which one examines the bibliographic references received.**

***Keywords*-citation analysis; citations graph; research evaluation;**

I. Introduction

Citation Analysis is of great importance to the scientific community because it can assist researchers around the world in locating articles relevant to their fields of interest. Many methods have been proposed for ranking the importance of articles, one of which is the Impact Factor. The Impact Factor was proposed by Eugene Garfield [1], [2], [3] and is used for ranking the importance of a journal/conference taking into account the number of direct citations received by the articles published in it.

In order to rank the importance of a specific scientific article one can use the number of direct citations that it received and the Impact Factor of the journal or conference at which it was published.

Apart from this approach, which is more general in nature, others more specific ones that quantify the importance of a researcher's work have been proposed. Such metrics are the h-Index [4] and the g-Index [5]. These metrics use the collection of all the articles a researcher has (co-) authored, plus the sum of all direct citations received in order to calculate one distinct value that quantifies the impact the researcher has made on his scientific field.

The Cascading Citation Analysis approach [6], [7], [8], in a way analogous to that of the Google Page Rank algorithm [9], and to Rousseau's approach [10], evaluates (ranks) the scientific impact of a published article not just by considering the number of direct citations received, but also by taking into consideration the scientific impact of the articles that cite the given (cited) article. More specifically, the scientific impact a given (published) article represents is calculated by considering not only the direct but also the indirect citations received. The Cascading Citations Indexing Framework suggests that citations should be addressed at the (article, author) level for us to be able to rank the contribution of each author's scientific work.

The rest of the paper is organized as follows; In Section II the Cascading Citations Indexing Framework (cc-IF) is presented in order to explain the terminology used in the rest of the paper. Section III presents the methodology used for testing the algorithm, which is presented in detail in Section IV. The results from the execution of the algorithm against the CiteSeer database [11] are shown in Section V, and, finally, the last section presents the conclusions.

II. The Cascading Citations Indexing Framework

Before describing the algorithm it is necessary to present the terminology used in the Cascading Citation Indexing Framework (cc-IF). We are going to use the Citation Graph to depict the relationships that exist among articles and explain the fundamentals of the cc-IF which include the terms citation, self-citation, n-gen citations (citation of rank n), n-chord and n-self-chord (chord/self-chord of rank n).

A. The Citation Graph

The citation graph is a representation of the relationships that exist between scientific articles, based on the references that each article provides. In Figure 1, the articles are presented as the nodes of the directed graph. Therefore in this example 5 articles are included, numbered 1 to 5. For each article we have also included the corresponding authors which appear as labels above each node. A label used more than once implies that this author has authored/co-authored more than one of the articles presented here.

The edges of the graph represent references among articles. For example, the edge leaving node 2 can be interpreted as "Article 2 references article 1". The incoming edges are the direct citations received by a specific article. For article 1 we can state that "Article 1 receives 3 direct citations, one from article 2, one from 3 and one from 5".

B. Cascading Citation Analysis Terminology

According to the cc-IF direct citations like the ones discussed in the previous section are named 1-gen citations.

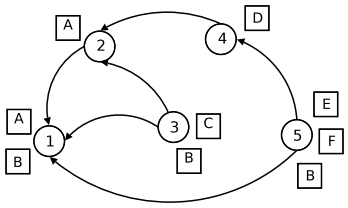

Figure 1. Citation Graph

If we carefully examine the graph we are going to find that article 1 also receives an indirect citation from article 4 through article 2. This is considered to be a 2-gen citation. In general an n-gen citation exists between a source article S and the target article T if there is a directed path in the citation graph from node S to node T. In the example presented, the greatest gen citation is of rank 3, from article 5 to article 1, through the citation path $5 \rightarrow 4 \rightarrow 2 \rightarrow 1$.

If we now consider not only the citations from article to article but also the authors of each article we are able to define the self citations of rank n. For example, article 1 receives a 1-gen citation from article 2. Article 1 is written by authors A and B, and article 2 is written by author A. In such a case the pair (1, B) receives a 1-gen citation from article 2 whereas the pair (1, A) receives a 1-gen-self-citation from the same article. We consider a citation to be a self-citation for a pair (article, author) if the author of the article is also found in the author list of the citing article.

Apart from the information listed above we are able to detect even more relations among the articles included in the example. By carefully examining the citation paths presented in Figure 1, we are able to detect more than one paths that have the same source article and the same target article, differentiated only by the fact that they are of different rank. For example, consider articles 3 and 1. The citations provided from article 3 to article 1 are two; the 1-gen citation through the path $3 \rightarrow 1$ and the 2-gen citation through the path $3 \rightarrow 2 \rightarrow 1$. In this case the 2-gen citation is called a 2-gen-chord. In other words a n-gen chord (n>1) exists between a source article S and a target article T if apart from the n-path between S and T a direct 1-gen citation exists between the two articles. Chords are also considered at the (article, author) level, thus, we define the n-gen-chords and the n-gen-self-chords, where $n > 1$.

According to the cc-IF, the citations that a (article, author) pair receives can be calculated up to depth n, thus, producing a number of distinct values. If, for example, we choose to consider the citations up-to depth 2 then the following values are going to be calculated: 1-gen citations, 1-gen-self citations, 2-gen citations, 2-gen-self citations, 2-gen chords and 2-gen-self chords. These values are stored in a table named Medal Standings Output (MSO). This table can then be used in order to examine the importance of a specific article or author.

```
<record>
<header>
<identifier>oai:CiteSeerPSU:number#</identifier>
</header>
<metadata>
<dc:title>The Title</dc:title>
<oai_citeseer:author name="authorName"> </oai_citeseer:author>
<oai_citeseer:relation type="References">
<oai_citeseer:uri>oai:CiteSeerPSU:number#</oai_citeseer:uri>
</oai_citeseer:relation>
<oai_citeseer:relation type="References">
<oai_citeseer:uri>oai:CiteSeerPSU:number#</oai_citeseer:uri>
</oai_citeseer:relation>
</oai_citeseer:oai_citeseer>
</metadata>
</record>
```

Figure 2. CiteSeer Record

III. Methodology

The algorithm we present calculates and stores all the citations present in a bibliographic database, producing the MSO table for all the distinct (article, author) pairs stored up-to depth 3. In order to test the algorithm we used the CiteSeer bibliographic database. Due to the large amount of data that need to be stored, accessed and retrieved, a relational database was used. The database is used in order to store part of the original data from CiteSeer as well as to keep the calculated values, the 1-gen, 2-gen, 3-gen citations (plus all individual citation paths) and the MSO table.

A. Data

CiteSeer provides the bibliographic data using the Open Access Initiative (OAI) format, which is XML based. A sample record is shown in Figure 2. For simplicity, only the identifiers that are used by the algorithm are listed.

Each article is identified by a unique identifier generated by CiteSeer, defined by the <*identifier*> tag, as shown in Figure 2. Other fields required by the algorithm are the title, defined by <*dc:title*> tag, the authors, defined by the tag <*oai_citeseer:author*> and the list of references included in each article, defined by the <*oai_citeseer:relation*> tag.

B. Database Structure

We used a relational database schema that includes the following relations:

- ***article****(identifier, title)*: Information about the articles.
- ***authors****(authorid, name)*: Information about the authors.
- ***art_has_authors****(identifier, authorid)*: Maps the articles with their authors.
- ***citations****(identifier, isrefby)*: The citations that each article has received.
- ***gen1****(id, identifier, authorid, fromid, self)*: The 1-gen citations for all (article, author) pairs. The citation path is fromid → identifier where fromid is the source article and identifier is the target article. One extra field is used (self) that is assigned the value of 1 in the case of a self citation.

- ***gen2****(id, identifier, authorid, fromid, throughid01, chord, self)*: The 2-gen citations for all (article, author) pairs and the citation path. The citation path is fromid → throughid01 → identifier. The chord field is assigned the value of 1 in the case of a 2-gen chord.
- ***gen3****(id, identifier, authorid, fromid, throughid01, throughid02, chord, self)*: The 3-gen citations for all (article, author) pairs and the citation paths. The citation path is fromid → throughid01 → throughid02 → identifier. The chord field is assigned the value of 1 in the case of a 3-gen chord.
- ***mso****(identifier, authorid, g1, g2, g3, sg1, sg2, sg3, cg2, cg3, scg2, scg3)*: The MSO table keeps summarized information about all types of citations received by each (article, author) pair.

C. Parse Algorithm

The CiteSeer database consists of 72 files, each holding 10.000 articles with their corresponding bibliographic details. Articles appearing in the list of references of a particular article, are also part of the CiteSeer database. In order to retrieve the necessary information for the execution of the algorithm and to store it in the relational database presented at Section III-B we developed a parsing algorithm.

The algorithm parses the files and stores all necessary information in the first four relations of the database schema of Section III-B. We should mention though that certain errors occurred during this process, like articles with no information about the authors and records which included special characters that we could not process. The former records were excluded because they lack the completeness required by the algorithm and the latter in order to simplify the procedure.

We should also note that the authors are identified only by their name, and since a person's name can be written in many different formats, the name alone does not guarantee the uniqueness of the author. For the purpose of testing our algorithm each name is considered to uniquely identify an author. We might encounter mismatching problems, like two authors with the same name that are considered to be the same person or a person whose name appears in many formats and, thus, the algorithm considers him to be not one but multiple authors.

IV. CC-IF ALGORITHM

In previous work ([6], [7]) we introduced an algorithm that calculated the direct and indirect citations received by a (article, author) pair up to depth 3.

The algorithm presented in this paper considers the citations at the (article, author) level and calculates all values presented in Section II-B up to depth 3. That is, it calculates the 1-gen, 2-gen, 3-gen, 1-gen-self, 2-gen-self, 3-gen-self, 2-gen-chord, 3-gen-chord, 2-gen-self-chord, and 3-gen-self-chord citations.

Algorithm 1 cc-IF

```
1 // R, T, L are articles
2 // A is author
3 // through_info is the collection of columns
4 // that contain the path information
5 cc-IF(depth, I, ADC, AA)
6 if depth = 1 then
7   foreach R in I do
8     foreach T in ADC[R] do
9       foreach A in AA[R] do
10        S = check_self(AA[R], AA[T])
11        insert_gen{1}(autoid, R, A, T, [], S)
12 else
13   cc-IF(depth-1, I, ADC, AA)
14   prev_gen = data from table gen{depth-1}
15   foreach row in prev_gen
16     R = row[identifier]
17     A = row[authorid]
18     T = row[fromid]
19     foreach L in ADC[T] do
20       S = check_self(AA[L], A)
21       C = check_chord(R, L)
22       TI = make_row[row[through_info], S]
23       insert_gen{depth}(autoid, R, A, TI, S)
24 calculate_mso()
```

In addition it not only creates the MSO table but for each citation received by a distinct (article, author) pair it calculates and stores in the database the corresponding citation path.

A. Pre-processing stage

Three data structures are necessary for the execution of the algorithm: the ***Article Direct Citations (ADC),*** the ***Article Authors (AA),*** and the list of the articles ***I*** that need to be processed. These structures are created based on the tables articles, art_has_authors and citations.

We denote an article by R_x. Let the list of all articles that need to be processed be $\boldsymbol{I}$=[R_1, R_2, R_3, ..., R_m] that is, the database contains m articles. Let CR_x denote the list of articles that reference R_x. Thus, CR_x is a subset of $\boldsymbol{I}$ and the Article Direct Citations (ADC) data structure is $\boldsymbol{ADC}$=[CR_1, CR_2 ,CR_3 , ... , CR_m].

Let the list of all authors of R_x be AR_x=[A, B, C,...] where A, B, C are the distinct co-authors which are different for each article. The Article Authors (AA) data structure is $\boldsymbol{AA}$=[AR_1, AR_2, AR_3, ..., AR_m].

B. The algorithm

The algorithm presented is recursive and the number of iterations is equal to the depth at which we want to examine the citations. It receives as input the desired *depth*, the list of identifiers (*I*) to be processed, the *Article Direct Citations (ADC)* data structure and the *Article Authors (AA)* data structure presented in Section IV-A. The output of the algorithm is the full list of all citation paths up-to the desired depth, and the characterization of each path at the (article, author) level based on the terms presented in Section II-B.

The algorithm recursively executes until the value of depth is equal to 1. At this point (line 6) the if condition is met and

the algorithm begins the calculations for 1-gen citations. For each article R in the list of articles I, the algorithm iterates through all the citations that this article has received (line 7). This information is found in the ADC structure. Then, for each such referencing article, identified by T, and for each author A of the article in question, the algorithm checks whether the specific author also exists in the list of authors of T (line 10). If the check_self function returns TRUE then this is a self citation. Finally, a new record is inserted in table gen1 that consists of information about the article, the author, the referencing article, the "through" information and the indication of whether this is a self-citation or not (line 11).

As soon as it checks all articles in list I, the algorithm returns to the incomplete recursive calls starting with the one where the value of depth equals two. For each recursive call the algorithm re-uses the information it calculated in the exact previous recursive call in order to calculated the new citations of higher rank. In other words, in order to calculate the 2-gen citations the algorithm retrieves all (article, author) 1-gen pair citations and for each pair calculates the 2-gen citations (lines 14-23). For each record present in the previous level citations table we retrieve information for the article (R), the author (A) and the source article of the citation (T). All direct references that the source article has received are considered n-gen citations for the target article R. We then check whether the citation is a chord and moreover if it is a self citation or not (lines 20-21). Finally, the desired values along with the intermediate edges are stored in the database (line 23).

After the calculation of all gen citations up-to the defined depth we summarize the results for each (article, author) pair, thus, producing the MSO table (line 24). A function named calculate_mso() is used for this purpose, that counts the total number of citations for each distinct value that needs to be stored in the MSO table.

V. Results of the algorithm

The algorithm was programmed using the Python programming language and MySQL was used for the creation and manipulation of the relational database described at III-B. The database that was created with the data provided by CiteSeer consisted of 658.045 articles, with 410.945 identified authors and 1.643.057 direct references among the articles. After the execution of the algorithm the following records where stored in the database:

- **Table gen1:** Approximately 4 million records.
- **Table gen2:** Approximately 20 million records.
- **Table gen3:** Approximately 103 million records.
- **Table mso:** Approximately 450 thousand records.

These records represent the number of distinct (article, author) pairs located in the database that receive at least one 1-gen citation/self-citation.

	Title	Author	1-gen
1	Graph-Based Algorithms for Boolean ...	Randal E. Bryant	1.600
2	Optimization by Simulated Annealing	M. P. Vecchi	1.339
3	Optimization by Simulated Annealing	C. D. Gelatt	1.339
4	Optimization by Simulated Annealing	S. Kirkpatrick	1.339
5	A Method for Obtaining Digital ...	R. L. Rivest	1.219
6	A Method for Obtaining Digital ...	A. Shamir	1.219
7	A Method for Obtaining Digital ...	L. Adleman	1.219
8	Congestion Avoidance and Control	Michael J. Karels	1.123
9	Congestion Avoidance and Control	Van Jacobson	1.123
10	Statecharts: A Visual ...	Comm. A. Pnueli	1.043

Table I
Top-10 pairs based on 1-gen citations

By sorting the MSO table we can demonstrate the results of the algorithm based on the 1-gen, 2-gen and 3-gen citations received by the individual (article, author) pairs. We observe that the listing of the pairs that received greater number of citations differentiates based on the sorting that we apply to the results.

For example, if we choose to sort the results based on the 1-gen citations received by each pair then the dominant pair is (*"Graph-Based Algorithms for Boolean Function Manipulation"*, Randal E. Bryant) which receives 1.600 references, as shown in Table I.

If we now choose to present the top 10 (article, author) pairs based on the 2-gen citations received then we get the results of Table II which are different from those in Table I. Here we can see that the article *"Congestion Avoidance and Control"* by Michael J. Karels and Van Jacobson has moved to the first positions in the list with 13.444 2-gen citations for Michael J. Karels and 13.424 citations for Van Jacobson. The difference in the number of citations between the two authors is due to the self citations that Van Jacobson received by citing at future works this article. Such a difference can also be noted for the *"Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism"* article and its three authors each of which receives a different number of 2-gen citations. If we search for the position in the listing of the (article, author) pair receiving the most 1-gen citations we are going to find that it has moved down to the 11th position due to the fact that even though it collected 1.600 1-gen citations it received 6.574 2-gen citations.

If we sort based on the 3-gen citations (Table III) that the pairs received we are going to find that the article *"Congestion Avoidance and Control"* remains in the first two positions with Michael J. Karels receiving a total of 132.977 3-gen citations and Van Jacobson a total of 132.877 citations (100 less than Michael J.Karels). In this case the *"Graph-Based Algorithms for Boolean Function Manipulation"* by Randal E. Bryant received 46.478 citations, thus, moving down to the 32nd position in the listing.

	Title	Author	2-gen
1	Congestion Avoidance and Control	Michael J. Karels	13.444
2	Congestion Avoidance and Control	Van Jacobson	13.424
3	A Method for Obtaining Digital ...	A. Shamir	8.996
4	A Method for Obtaining Digital ...	L. Adleman	8.996
5	A Method for Obtaining Digital ...	R. L. Rivest	8.996
6	Supporting Real-Time Applications ...	David D. Clark	8.526
7	Supporting Real-Time Applications ...	Lixia Zhang	8.497
8	Supporting Real-Time Applications ...	Scott Shenker	8.434
9	Random Early Detection ...	Van Jacobson	6.630
10	Tcl and the Tk Toolkit	John K. Ousterhout	6.583

Table II
TOP-10 PAIRS BASED ON 2-GEN CITATIONS

	Title	Author	2-gen
1	Congestion Avoidance and Control	Michael J. Karels	132.977
2	Congestion Avoidance and Control	Van Jacobson	132.877
3	Supporting Real-Time Applications ...	David D. Clark	83.380
4	Supporting Real-Time Applications ...	Lixia Zhang	83.188
5	Supporting Real-Time Applications ...	Scott Shenker	82.601
6	A Scheme for Real-Time Channel ...	Dinesh C. Verma	73.663
7	A Scheme for Real-Time Channel ...	Domenico Ferrari	73.560
8	A Method for Obtaining Digital ...	A. Shamir	69.845
9	A Method for Obtaining Digital ...	L. Adleman	69.845
10	A Method for Obtaining Digital ...	R. L. Rivest	69.845

Table III
TOP-10 PAIRS BASED ON 3-GEN CITATIONS

It is also very interesting to notice the number of chords that each of the three dominant pairs has received, as shown in Table IV. In general, the fact that *"Congestion Avoidance and Control"* received the greatest number of chords (for both authors) could imply that this article is of great importance in the specific field of research. In order to safely draw a conclusion, these numbers have to be examined in accordance with the overall number of citations received by each pair and possibly taking into account the number of research articles in the specific scientific area.

VI. CONCLUSIONS

Based on the Cascading Citation Indexing Framework a new algorithm has been developed that calculates the citations received by a (article, author) pair taking into account not only the direct (1-gen) citations it receives but also the indirect citations (2-gen, 3-gen) it receives through articles that directly cited it. Other calculated values are the self citations, the chords, and the self chords at all levels.

	Title	Author	2-gen chords	3-gen chords
1	Congestion Avoidance ...	Michael J. Karels	4429	31881
2	Congestion Avoidance ...	Van Jacobson	4383	32018
3	Graph-Based Algorithms ...	Randal E. Bryant	3347	11055

Table IV
NUMBER OF CHORDS FOR 3 DOMINANT PAIRS

Through the presentation of the results, the different aspects of the n-gen citations were demonstrated and it was noted that if, in addition to the direct citations we also consider the indirect ones, the sorting of the results can vary.

Future work on this field will: (a) test the algorithm for its scalability, (b) improve the algorithm to operate on dynamic article collections, and, (c) evaluate its performance in terms of memory and CPU usage. Further research is also going to assist us in identifying relations between the calculated values and, if possible, in calculating a unique value based on some criteria for the measurement of the scientific value of a (article, author) pair in the context of the research field it belongs to.

REFERENCES

[1] E. Garfield, "Citation indexes for science. a new dimension in documentation through association of ideas," *Science*, vol. 122, pp. 1123–1127, 1955.

[2] ——, "Journal impact factor: a brief review," *CMAJ*, vol. 161, no. 8, pp. 979–980, October 1999. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/10551195

[3] ——, "The agony and the ecstasy - the history and meaning of the journal impact factor." International Congress on Peer Review And Biomedical Publication, sep 2005.

[4] J. Hirsch, "An index to quantify an individual's scientific research output," in *Proceedings of the National Academy of Sciences*, vol. 102, no. 46. National Acad Sciences, 2005, pp. 16 569–16 572.

[5] L. Egghe, "Theory and practise of the g-index," *Scientometrics*, vol. 69, no. 1, pp. 131–152, 2006.

[6] D. Dervos and T. Kalkanis, "cc-IFF: A Cascading Citations Impact Factor Framework for the Automatic Ranking of Research Publications," *3rd IEEE International Workshop on Intelligent Data Acquisition and Advanced Computer Systems: Technology and Applications (IDAACS 2005), Sofia, Bulgaria*, September 2005. [Online]. Available: http://dlist.sir.arizona.edu/1105/

[7] D. Dervos, N. Samaras, G. Evangelidis, and T. Folias, "A new framework for the citation indexing paradigm," in *Proc. of the ASSIST 2006 Annual Meeting*, November 2006. [Online]. Available: http://eprints.rclis.org/archive/00008405/

[8] D. Dervos and L. Klimis, "Exploiting cascading citations for retrieval," in *Proc. of the ASSIST 2008 Annual Meeting*, October 2008.

[9] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks and ISDN Systems*, pp. 107–117, 1998.

[10] R. Rousseau, "The gozinto theorem: Using citations to determine influences on a scientific publication," *Scientometrics*, vol. 11, no. 3-4, pp. 217–229, 1987.

[11] CiteSeer, "http://citeseer.ist.psu.edu/."