DATA EXTRACTION FROM SERVERS BY THE INTERNET ROBOT

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

_____

Nam Pham

Certificate of approval:

_____	_____
Hulya Kirkici Professor Electrical and Computer Engineering	Bogdan M. Wilamowski, Chair Professor Electrical and Computer Engineering
_____	_____
Lloyd S. Riggs Professor Electrical and Computer Engineering	Vitaly J. Vodyanoy Professor Anatomy, Physiology, and Pharmacology

_____

George T. Flowers
Dean
Graduate School

DATA EXTRACTION FROM SERVERS BY THE INTERNET ROBOT

Nam Pham

A Thesis

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama

August 10th, 2009

DATA EXTRACTION FROM SERVERS BY THE INTERNET ROBOT

Nam Pham

_____
Signature of Author

08/10/2009
Date of Graduation

THESIS ABSTRACT

DATA EXTRACTION FROM SERVERS BY THE INTERNET ROBOT

Nam Pham

Master of Science, August 10, 2009
(B.S., Auburn University, May 2008)

103 Typed Pages

Directed by Bogdan. M. Wilamowski

Data extraction from internet is a way to download and extract the required data automatically from web servers. In this thesis, we present a method called the Internet Robot to extract the data directly from a web server by using Perl scripting language with the powerful regular expressions. The regular expressions are widely used in this method to reduce the complexity of the program code as well as increase up the downloading and extracting speed. The Internet Robot in this thesis is a process of three steps: data collection, data filtering and processing and data presentation. The final result of this process will be the html files- with all required data in the typical format that is presented under different links of a webpage. The accuracy and speed make this method become unique in processing and extracting data not only from the internet but also from an available database.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my advisor, Professor B. M. Wilamowski. Without his patience and guidance I would not be in the position I am today. From him I have learned a multitude of things, of which, engineering is only the tip of the iceberg.

Style manual or journal used Journal of Approximation Theory (together with the style known as “aums”). Bibliography follows van Leuen’s *A Handbook of Scholars*.

Computer software used The document preparation package MS Word together with the departmental style-file aums.sty.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

THE FUNDAMENTALS OF THE INTERNET ROBOT

1.1 Introduction.

With a tremendous growth in available information to the masses [1] the question is how users can search and extract the useful information they need in the shortest amount of time. In other words, making use of consolidated information requires such substantial efforts since the web pages are generated for visualization not for data exchange [2]. This requires methods developed to optimize users' searching process. This thesis introduces a method known as the Internet Robot (IR) to process and extract data from the web servers by using Perl scripting language. The Internet Robot can be understood quickly as a special program that serves for a certain purpose of users, particularly in processing and extracting data automatically from a web server with a structured template. The programs that perform the task of Information Extracting (IE) are referred to as Extractors and Wrapper [3]. This method doesn't use browsers to handle the web but it does that by using modules such as *LWP* (the extensive lib www-

perl library) [4]. This aspect turns the Internet Robot into an effective solution to extract data with an accelerated speed [1].

The executing procedure of the Internet Robot can be divided into three steps. 1: *Data collection*: to extract all information from web pages into database to analyze. 2: *Data filtering and processing*: the useful information needs to be extracted from the database with a random mass of information. 3: *Data presentation*: all extracted information is processed and sorted in the format which is convenient for users.

To have a closer look about how the Internet Robot really works, this chapter will describe more details with a typical application of how it extracts all information about authors' name, titles, volume numbers, issue numbers, page numbers, issue date etc automatically from IEEE Xplore and rearranges them in the typical format that readers can search and read papers in the fastest way as well as know all details about authors, dates, citations etc in the easiest way. Overview of the Internet Robot is given in Chapter1, Chapter 2 explains some applications of the Internet Robot to download and extract useful data with illustrating figures from servers of IEEE Xplore, particularly IEEE Transactions on Industrial Electronics. Chapter 3 will explain details of Perl code.

1.2 Overview of the Internet Robot.

Execution of the Internet Robot is very simple. Programmers only need Perl scripting language which is an interpreted language. It is parsed and executed at runtime instead of being compiled into binary form and then run [4]. Moreover, this is an open-source software for users with the standard modules which also comes with Perl [5]. Like the built-in functions, these modules provide users with hundreds of prewritten resources.

A module is made up of Perl code written in a way to conform to certain conventions so users can access that code from their program. Perl modules provide users with a great deal of prewritten code and are stored in files with the extension *.pm*. Users can load such modules into their code by using the *use* statement [4].

1.2.1 Perl scripting language.

Perl which stands for “Practical Extraction and Report Language” is a way to make the report processing easier. It is a great language for doing data manipulation tasks. It is fast, portable and accessible. Perl is the programming language which excels at handling textual information. It has been used with good success for systems administration tasks on Unix systems, acting as glue between different computer systems, World Wide Web, CGI programming, as well as customizing the output from other programs [4]. Modern Perl interpreters are in fact not interpreters but compilers that pre-compile the whole script before running it [6].

Perl is the most prominent web programming language because of its text processing features and development in usability, feature, and/or execution speed. Handling HTML forms is made simple by the CGI.pm module, a part of Perl’s standard distribution. Perl has a powerful regular expression engine built directly into its syntax. A regular expression is a syntax that increases ease of operations that involve complex string comparisons, selections, replacements and hence, parsing [1].

1.2.2 Web browser.

As we said before in this chapter, the Internet Robot doesn't use browsers but it plays that role by using modules. These modules have capability to download a web page. In other words, they can emulate as a browser. There are a number of modules supporting programmers to do that in Perl such as *LWP::Simple* or *LWP::UserAgent etc* [4] Let's look at a typical example to download the main FAQ index at CPAN by using *get* function of *LWP::Simple* module and store that web page into the html file-name.html.

```
use LWP::Simple;
 $add="http://www.cpan.org/doc/FAQs/index.html";
$content=get("$add");
open FILEHANDLE,">name.html";
print FILEHANDLE $content;
close FILEHANDLE;
```

An address is called into the subroutine "&content", this subroutine uses get function of *LWP::Simple* to copy web source into the variable "$content". And then the content of this web page is stored into the file name.html. The final result will be a file having the similar content as the web page http://www.cpan.org/doc/FAQs/index.html .

In this application, instead of using LWP::Simple to emulate as a browser, we substitute it by the written *Wget.exe* [7] which is available and familiar with all users. GNU Wget.exe is a free software package for retrieving files using HTTP, HTTPS and FTP, the most widely-used Internet protocols. In another words, its main function is to link to a certain webpage with an address input and copy all web sources of that webpage

into a file. Because of this reason, Wget.exe is considered as non-interactive. It allows users to disconnect from the system and let Wget.exe finish the job without logging in. This is extremely convenient and time-saving when users have to transfer a lot of data from the different web-links. In contrast, other web browsers always require users' constant presence. Generally, the Wget.exe works almost the same as the browser built from LWP::Simple module. However, it is more powerful because it gives users more options.

Below is the basic structure to execute Wget.exe from the Perl script. *"–O"* means that the documents will not be written to appropriate files but all will be concatenated together and written to a file while *"–q"* turns off Wget.exe output on the prompt screen.

```
$add= address of webpage;
$fname= "filename";
system("wget.exe", "-q", "-O", $fname,$add);
```

By assigning value of variables $add and $fname by an address of a webpage and a name of a file which will be generated respectively and using structure *"system"* as above, the new html file will contain almost content of that web link except Java script. Once the new html file contains all data that users want, they can open and copy this data into an array. From here, users can extract all information they need by using Perl commands, especially the regular expressions which is the power of Perl language and makes it different from other languages.

CHAPTER 2

APPLICATIONS OF THE INTERNET ROBOT

2.1 Application of the Internet Robot in extracting article data from IEEE.

Our goal is to download desired information of papers from 1988 until now from IEEE Transactions on Industrial Electronics with all details about authors' name, titles, volume numbers, page numbers, date issues, content of abstracts, Abstract links and Full Text links which will be active if you are a subscriber of IEEE Xplore as Fig. 1.

IEEE IEEE Transactions on Industrial Electronics ies

Volume 37, Number 1, Jan 1990 Access to the journal on IEEE XPLORE IE Transactions Home Page

37.1.1 N. Hamada, K. Bekki, T. Yokota, "VLSI logic design with logic programming and knowledge-base technology," *IEEE Trans. on Industrial Electronics*, vol. 37, no. 1, pp. 1-5, Jan 1990. Abstract Link Full Text

Abstract: An approach to VLSI logic design using partial and general structural specifications in addition to behavioral specifications is developed. This approach requires a new style of programming technique, especially if a universal solution procedure for all types of architectures is needed. Knowledge of the design process involves unification of the heterogeneous (i.e. behavior and structure) information between a system and its parts, as well as representation of functional modules in order to ensure their reusability in an efficient manner. Following these strategies, a logic synthesis expert system, ProLogic, is developed, and the system is evaluated using MPU-type VLSIs. It is found that the universal connecting procedure for any compound functional module that unifies the behavioral and structural specifications between a total module and its parts improves logic design efficiency by a factor of 2 and that logic programming, object-oriented frames, and rule bases implemented in ProLogic improve software productivity by a factor of 5

37.1.2 S. Komada, K. Ohnishi, "Force feedback control of robot manipulator by the acceleration tracing orientation method," *IEEE Trans. on Industrial Electronics*, vol. 37, no. 1, pp. 6-12, Jan 1990. Abstract Link Full Text

Abstract: The authors propose a novel approach to force and compliance control of multi-degree-of-freedom (DOF) robot manipulators. The acceleration tracing orientation method (ATOM) is applied to both controllers. The control law is described in the Cartesian space; however, the final command is the acceleration in the joint space. The interactive terms in each joint disturb and deteriorate the joint motion. The disturbance observer cancels out the total sum of these terms and enables each joint to trace the acceleration command. As a result, a robust control is possible in the force task. The testing of the proposed system in a three-DOF robot manipulator is discussed

37.1.3 H.-G. Yeh, "Real-time implementation of a narrow-band Kalman filter with a floating-point processor DSP32," *IEEE Trans. on Industrial Electronics*, vol. 37, no. 1, pp. 13-18, Jan 1990. Abstract Link Full Text

Fig. 1. Format of a html file.

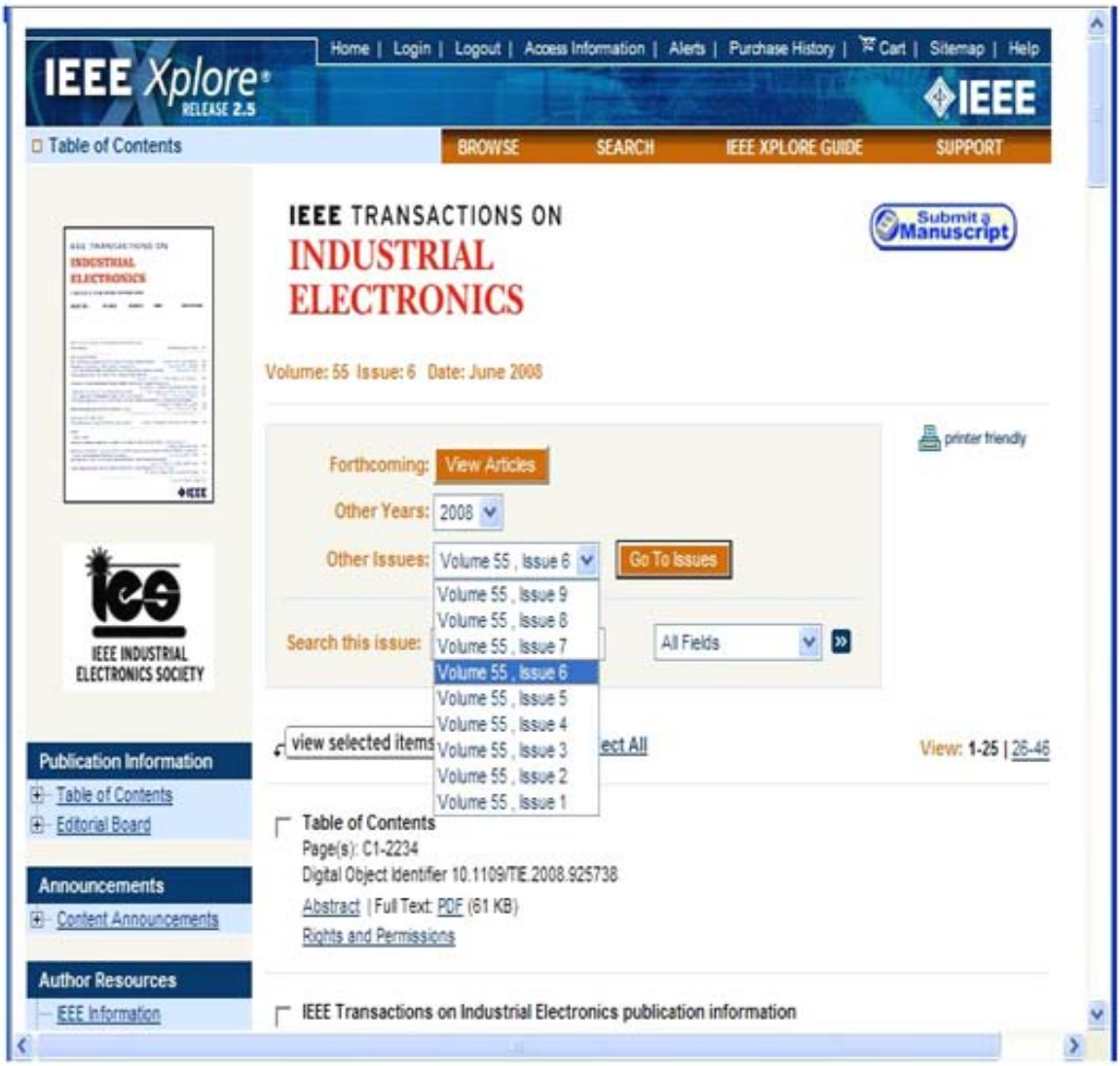


Fig. 2. IEEE Xplore webpage.

With this typical presentation, readers can find out what paper they are interested in the easiest and fastest way by reading information about abstracts, authors, titles…Once readers want to read or download the full text of a paper in PDF format, they can click to the Full Text link without logging their password in IEEE Xplore in case the data base is available. By using the same concept according with “Publish and Perish”

software, we can create a tool to let the users know more information about total number of papers, total number of citations per year, number of citations per paper, the most cited papers and the most downloaded papers.

2.2 Procedure of execution of the Internet Robot for this application.

All magazines or journals' issue level of IEEE have the same format link (OPAC) [8].

"http://ieeexplore.ieee.org/servlet/opac?punumber=X&isvol=Y&isno=Z"

X: code number of a magazine or a journal.

Y: volume number of one year

Z: number of issues in one volume.

In our application, IEEE Transactions on Industrial Electronics has its code number X=41. If users want to know all information about the papers of the issue number 6 in the volume 55, they only need to type in: http://ieeexplore.ieee.org/servlet/opac?punumber=41&isvol=55&isno=6 and this link will lead users to the papers in that issue (Fig. 2.). Every volume has many different issues and every issue has many papers which consist of 11 subject categories.

For convenience to follow how the Internet Robot works, a flowchart (Fig. 3) is depicted to present all its three steps: data collection, data filtering and processing, data presentation on web. (see more in APPENDIX A)

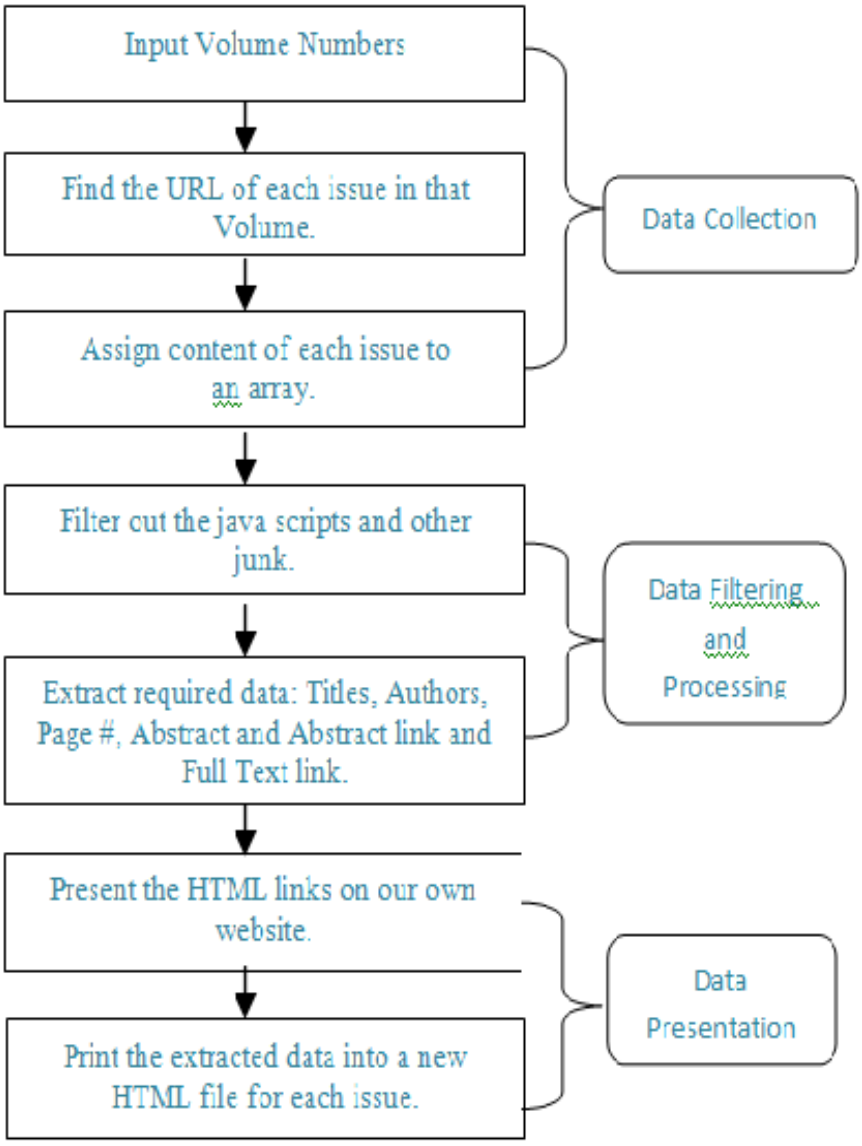


Fig. 3. Flowchart to describe all steps of Internet Robot.

2.2.1 Data collection.

There is one particular feature of Perl is that programmers can use outputs of other programs as their input. This is the background to build the control text file. When we input enough data as code number, volume number and issue number etc to refer to a

web link of IEEE Transactions in the control text file, the input data will be read into an array of Perl program. By assigning the code number, the volume number, and the issue number for variables, these variables will receive the same values from the input. This option will give programmers more flexibility without changing the generality of the codes. Fig. 4 is a sample of the control text file in this example (see more in APPENDIX G).

```
code=41
MAXissue=12
FROMissue=1
FROMyear=1988
TOissue=10
TOyear=2008
startYear=1953
directory=C:/Industrial Electronics on Trans/
journal=IEEE Transactions on Industrial Electronics
shortcut=IEEE Trans. on Industrial Electronics
```

Fig. 4. Control text file.

Moreover, instead of using the standard array in C or C^{++}, programmers can use *hash* in Perl which doesn't depend on the order of the input data by searching it text string keys, not numeric indexes. It is very handful when users can easily type some unexpected errors. In order to extract all information of titles, pager numbers, authors' name of all papers in this issue, at first the program has to call a sub-routine &sub

get_content, this sub-routine will copy source content of the above web page into an array @lines and return it back into the array @get_issue of the main program for example. By using another sub-routine &sub add_issue, all content of the array @get_issue will be copied into the ISS_no.html file. However, this html file only contains the web source of the 25 first papers, it still misses the web sources of other papers from 26 to 46 in this particular issue. By using the regular expressions we can fetch links of other papers in the array @get_issue. After these links are fetched, their source contents are copied and then appended into ISS_no.html file by the sub-routine &add_issue. For simplicity, assuming an array @P1 and another array @P2 are two returned values from our fetching process. At this point, @P=@P1.@P2 will contain the web sources of whole papers in this issue, no matter how many papers and how many links they are separated into, their content will be copied into the ISS_no.html file automatically. Traditionally, the approach of most researchers for the above example is to extract URLs from web pages and then use these extracted URLs to retrieve next pages via HTTP request [9].

```
my @get_issue=&get_content($addr);
my $temp=&add_issue(@get_issue);

sub-routines:

sub get_content
  {
    $add=$_[0];  # all incoming input
    $fname= "med.htm";
    system("wget.exe", "-q", "-O", $fname,$add);
    $file=$mydir."med.htm";
    open(r1,"<$file");
    @lines = <r1>;
    close(r1);
```

```
    unlink($fname); # delete temporary file
    return(@lines);
  }



sub add_issue
  {
    $file=$mydir."ISS_No.htm";
    open(K,">>$file");
    print K "@_";
    close(K);
    return 0;
  }
```

2.2.2 Data Processing.

In this step, all raw data that we need is saved into ISS_no.html file. So this file should be opened and copied into an array. The remaining process will base on this array to extract. This array will include in all details about authors' name, titles, page number etc mixed with a lot of junks that stays miscellaneously, disorderly that we don't need to extract. Using regular expressions according with some other built-in commands as *for* loop, *Index* etc we can fetch line by line of this array to extract our desired data.

2.2.2.1 Extract the titles.

From the webpage of IEEE Transactions on Industrial Electronics, we see every paper has all its information arranged orderly from title, authors' name, page number, abstract link and full text link. So whenever content of papers of this issue is copied into an array, its elements will contain all that information in the same order respectively. In other words, the starting point is to search the titles first. However, in our desired

information there is a lot of undesired information so called junks that we don’t need, therefore we have to filter them out.

The array element which contains the title has the following format.

<td class="bodyCopyBlackLargeSpaced">TITLE

To extract the title we need to fetch the element which contains the title. This is the unchanged procedure and will be applied through our remaining process. The array element that has information of the title stores some unique key words and has different format comparing with other elements, therefore it is the premise to define its position in the array by using the conditional statements and the regular expressions. Once the title is fetched we can use other commands such as index and substring…to discard the junky words so called key-words above that we don’t really need by allocating the starting point and the ending point of a title. The result will be a string having the title of a paper and stored into a variable which will be used later. Particularly in this case, the junky words as: <td class="bodyCopyBlack LargeSpaced"> and
 will be avoided.

2.2.2.2 Extract the authors’ name and the page numbers.

Once the title of a paper is fetched, the next two elements of the array will be two strings that contain information of author’s name and page number. This aspect depends on the structured template of web pages as well as the desired data. However, the elements which store the author’s name and the page numbers have some similar key words as other junky elements in this array, therefore it will be very difficult to fetch this information by using the regular expressions because the returned result will be

something different from the one we want. Possibly, the returned result is the last element having the same key words as in the regular expressions. However, the position of the array element containing the title has been defined already in the previous step, so it can be used as a reference to search for other elements of the author's name and the page numbers. Still repeat the same steps, fetching the elements that have information of the authors' name and the page numbers first and then extracting the author's name and the page numbers. The results will be two strings, one string of the author's name and one string of the page numbers and saved into two different variables. This procedure is kind of similar to the procedure of extracting titles.

2.2.2.3 Extract the abstracts, the abstract links and the full-text links.

Two strings below are the typical format of the array elements that store information of Abstract link and Full Text link, respectively.

Abstract.

href="/iel5/41/4505400/04454446.pdf?isnumber=4505400&prod=JNL&arnumber=4454446&arSt=1893&ared=1909&arAuthor=Levi%2C+E."class="bodyCopySpaced"> PDF (526 KB)

To get the Abstract link and the Full Text link of a paper we need to fetch the elements in the array that store the Abstract link and the Full Text link [9]. Although these fetched elements store information of the abstract link and the full text link but they are not the full links that can be used in html of the data presentation part. However, this

information is not useless to create the full Abstract link and Full Text link that can really work. First we should filter to get the necessary information and then concatenate it with the partial links that we create by assigning them into variables $b and $c. For example, assuming the variable $abstract="=4505400&arnumber=4454446&count=43&index=3" is a substring that we extracted from a long string above and then concatenating this string with another newly created variable which is also a string $b=" http://ieeexplore.ieee.org/xpls" ($link_abst= $b.$abstract;), the final string will be the abstract link of that paper.

The results of this process will be some respective links like these for Abstract link and Full Text link.

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?isnumber=4505400&arnumber=4454446&count=43&index=3

http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/41/4505400/04454446.pdf?tp=&isnumber=4505400&arnumber=4454446

When we had the abstract link, we can use a sub-routine to call that address and copy all its content into an array by using Wget.exe. From this array we can fetch the array element storing a full description of the paper abstract by another sub-routine &get_abstractspan. Once this has been done, the full description of the abstract will be returned and assigned into the variable $each_abst.

```
my @get_abstract= &get_content_ab($link_abst);
my$abst_content= &get_abstractspan(@get_abstract);
$each_abst[$jj]= $abst_content;
```

2.2.3 Data Presentation.

Our desired data about Author's name, volume number, issue number, page number, abstract link, full text link, abstract description of a paper is fetched, extracted and stored in the variables. Combining html with these variables in Perl script, these data will be presented in the favorite format which is easy and convenient for readers. Still fetching the remaining elements in the ISS_no.htm file, all authors' names, titles, abstracts etc of an issue will be extracted, presented and copied into a file name 55_6.htm having similar format as Fig. 1. While the variables $volume_ number and $issue_number are used to track number of an issue in the volume so that the program will automatically update a new name for the new issue, not overwrite on the old issue, and @add_link is an array consists of all information of the authors' names, the title etc that has been already presented in html format.

```
$volume="$volume_number"."_"."$issue_number";
$file=$mydir."$volume.htm";
open(Kvolume,">$file");
print Kvolume "$head <table>";
print Kvolume "@add_link";
print Kvolume "</table>";
close(Kvolume);
```

Expanding this concept and using "for loop", all issues of all volumes can be extracted at one time. Every issue will be generated into every html file, and all issues in one volume as well as many volumes will be grouped and generated in another html file. The obtained files are then released into the World Wide Web by presenting them as the links on a website. A sample website which contains the links of all desired data can be accessed from

http://tie.ieee-ies.org/tie/abs/index.htm (Fig. 5.)

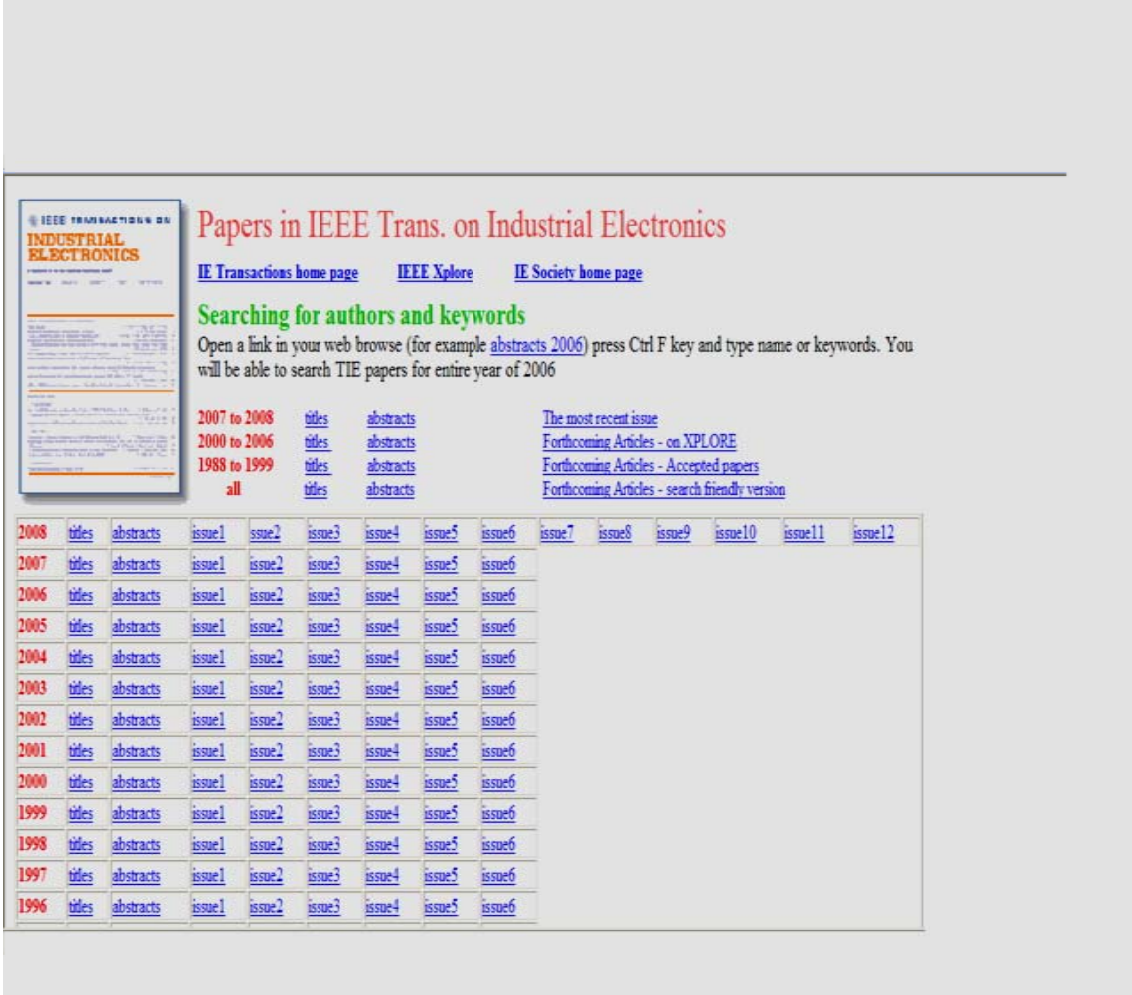


Fig. 5. Resultant webpage.

2.3 Other applications.

2.3.1 Combine the Internet Robot with "Publish and Perish" to extract the citations.

"*Publish and Perish"* is also a soft-ware written in Perl script to extract number of citations of papers. By inputting enough information of journal's name, year, field etc and running, this software will generate a text file including in the authors' name, the

titles, and number of citations of papers etc. If this text file is saved and used as an input of our another program, this program will compare each paper of each volume that we downloaded from IEEE Xplore with the papers from the text file to create a new html file with number of citations added (Fig. 6.) (see more in APPENDIX B).

MOST CITED TIE 2007 PAPERS

(SOURCE : GOOGLE SCHOLAR AS OF Sept 23, 2008)

389 papers were published in 2007 and they were cited 1146 times

1. E. Monmasson, M.N. Cirstea, "FPGA Design Methodology for Industrial Control Systems—A Review," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 4, pp. 1824-1842, April 2007. Abstract Link Full Text No. of Citations = 19
2. G. Escobar, P. Mattavelli, A.M. Stankovic, A.A. Valdez, J. Leyva-Ramos, "An Adaptive Control for UPS to Compensate Unbalance and Harmonic Distortion Using a Combined Capacitor/Load Current Sensing," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 2, pp. 839-847, Feb 2007. Abstract Link Full Text No. of Citations = 15
3. G. Escobar, A. A. Valdez, J. Leyva-Ramos, P. Mattavelli, "Repetitive-Based Controller for a UPS Inverter to Compensate Unbalance and Harmonic Distortion," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 1, pp. 504-510, Jan 2007. Abstract Link Full Text No. of Citations = 14
4. J.J. Rodriguez-Andina, M.J. Moure, M.D. Valdes, "Features, Design Tools, and Application Domains of FPGAs," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 4, pp. 1810-1823, April 2007. Abstract Link Full Text No. of Citations = 14
5. Seul Jung, Sung su Kim, "Hardware Implementation of a Real-Time Neural Network Controller With a DSP and an FPGA for Nonlinear Systems," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 1, pp. 265-271, Jan 2007. Abstract Link Full Text No. of Citations = 14
6. J. Pou, J. Zaragoza, P. Rodriguez, S. Ceballos, V.M. Sala, R.P. Burgos, D. Boroyevich, "Fast-Processing Modulation Strategy for the Neutral-Point-Clamped Converter With Total Elimination of Low-Frequency Voltage Oscillations in the Neutral Point," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 4, pp. 2288-2294, April 2007. Abstract Link Full Text No. of Citations = 13
7. S. Katsura, K. Ohnishi, K. Ohishi, "Transmission of Force Sensation by Environment Quarrier Based on Multilateral Control," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 2, pp. 898-906, Feb 2007. Abstract Link Full Text No. of Citations = 13
8. Sung-Sae Lee, Seong-Wook Choi, Gun-Woo Moon, "High-Efficiency Active-Clamp Forward Converter With Transient Current Build-Up (TCB) ZVS Technique," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 1, pp. 310-318, Jan 2007. Abstract Link Full Text No. of Citations = 13
9. C. Rech, J.R. Pinheiro, "Hybrid Multilevel Converters: Unified Analysis and Design Considerations," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 2, pp. 1092-1104, Feb 2007. Abstract Link Full Text No. of Citations = 13
10. Y. Maeno, Y. Ohsawa, "Human–Computer Interactive Annealing for Discovering Invisible Dark Events," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 2, pp. 1184-1192, Feb 2007. Abstract Link Full Text No. of Citations = 13
11. S. Katsura, K. Ohishi, "Modal System Design of Multirobot Systems by Interaction Mode Control," *IEEE Trans. on Industrial Electronics*, vol. 54, no. 3, pp. 1537-1546, March 2007. Abstract Link Full Text No. of Citations = 13

Fig. 6. Volume papers with a number of citations.

2.3.2 Apply the Internet Robot to extract the citations from "ISI Web of knowledge".

Instead of using "Publish and Perish" to extract citations from Google Scholar, we can use the Internet Robot to do this directly from the "ISI web of knowledge" server. When we have this data we can generate the similar html files as above (Fig. 6). The reason we need to extract citations from ISI web of knowledge because its database is different from Google Scholar. (see more in APPENDIX E).

2.3.3 Apply the Internet Robot to extract the forthcoming papers in different modes.

Using the Internet Robot, we can download the forthcoming papers weekly or monthly and save into two different files (see more in APPENDIX F). Nevertheless, there are some old papers removed and some new papers added. Therefore, how we know which papers were removed, which papers are still and which papers have just been added. By comparing these two files, the Internet Robot will generate new three files containing those papers with three different states. For example, there are 101 forthcoming papers on September 21th and 94 forthcoming papers on October 25th, after comparison, the result shows that there are 21 old papers removed (Fig. 8), 14 papers have just been added (Fig. 7.) and 80 common papers are still (Fig. 9.). In this case because the Internet Robot doesn't have to access to extract data directly from IEEE Xplore server but access directly to available database, so its speed is accelerated extremely faster. (see more in APPENDIX C).

IEEE Transactions on Industrial Electronics

14 new forthcoming articles as of Oct 25 2008 Access to the journal on IEEE XPLORE IE Transactions Home Page

1 H. Tanaka, K. Ohnishi, H. Nishi, T. Kawai, Y. Morikawa, S. Ozawa, T. Furukawa, "Implementation of Bilateral Control System Based on Acceleration Control Using FPGA for Multi-DOF Haptic Endoscopic Surgery Robot," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 K. Zhou, D. Wang, B. Zhang, Y. Wang, "Plug-In Dual Mode Structure Repetitive Controller for CVCF PWM Inverters," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 B. Wang, X. Yu, X. Li, "ZOH Discretization Effect on High-Order Sliding Mode Control Systems," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

4 K. Erbatur, O. Kurt, "Natural ZMP Trajectories for Biped Robot Reference Generation," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

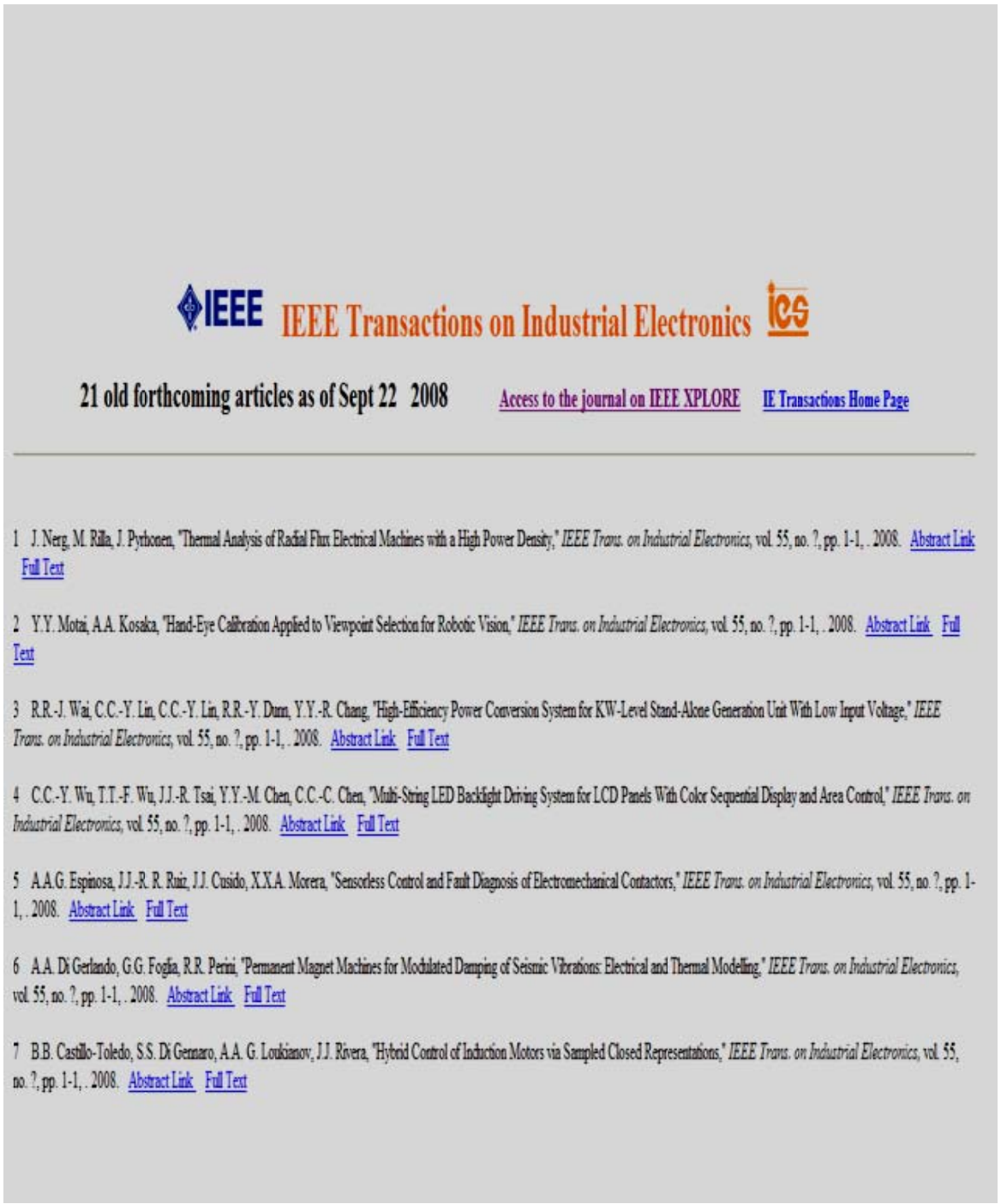
5 C. Kral, F. Pirker, G. Pascoli, H. Kapeller, "Robust Rotor Fault Detection by Means of the Vienna Monitoring Method and a Parameter Tracking Technique," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 T. Chen, L. Chen, L. Sun, X. Li, "Design and Fabrication of a Four-Arms Structure MEMS Gripper," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 J.P. V. S. Cunha, R.R. Costa, L. Hsu, "Design of First Order Approximation Filters for Sliding Mode Control of Uncertain Systems," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

8 J. Zaragoza, J. Pou, S. Ceballos, E. Robles, P. Ibanez, J.L. Villate, "A Comprehensive Study of a Hybrid Modulation Technique for the Neutral-Point-Clamped Converter," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 7. New papers.



IEEE Transactions on Industrial Electronics

21 old forthcoming articles as of Sept 22 2008 Access to the journal on IEEE XPLORE IE Transactions Home Page

1 J. Nerg, M. Rilla, J. Pyrhonen, "Thermal Analysis of Radial Flux Electrical Machines with a High Power Density," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 Y.Y. Motai, A.A. Kosaka, "Hand-Eye Calibration Applied to Viewpoint Selection for Robotic Vision," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 R.R.-J. Wai, C.C.-Y. Lin, C.C.-Y. Lin, R.R.-Y. Duan, Y.Y.-R. Chang, "High-Efficiency Power Conversion System for KW-Level Stand-Alone Generation Unit With Low Input Voltage," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

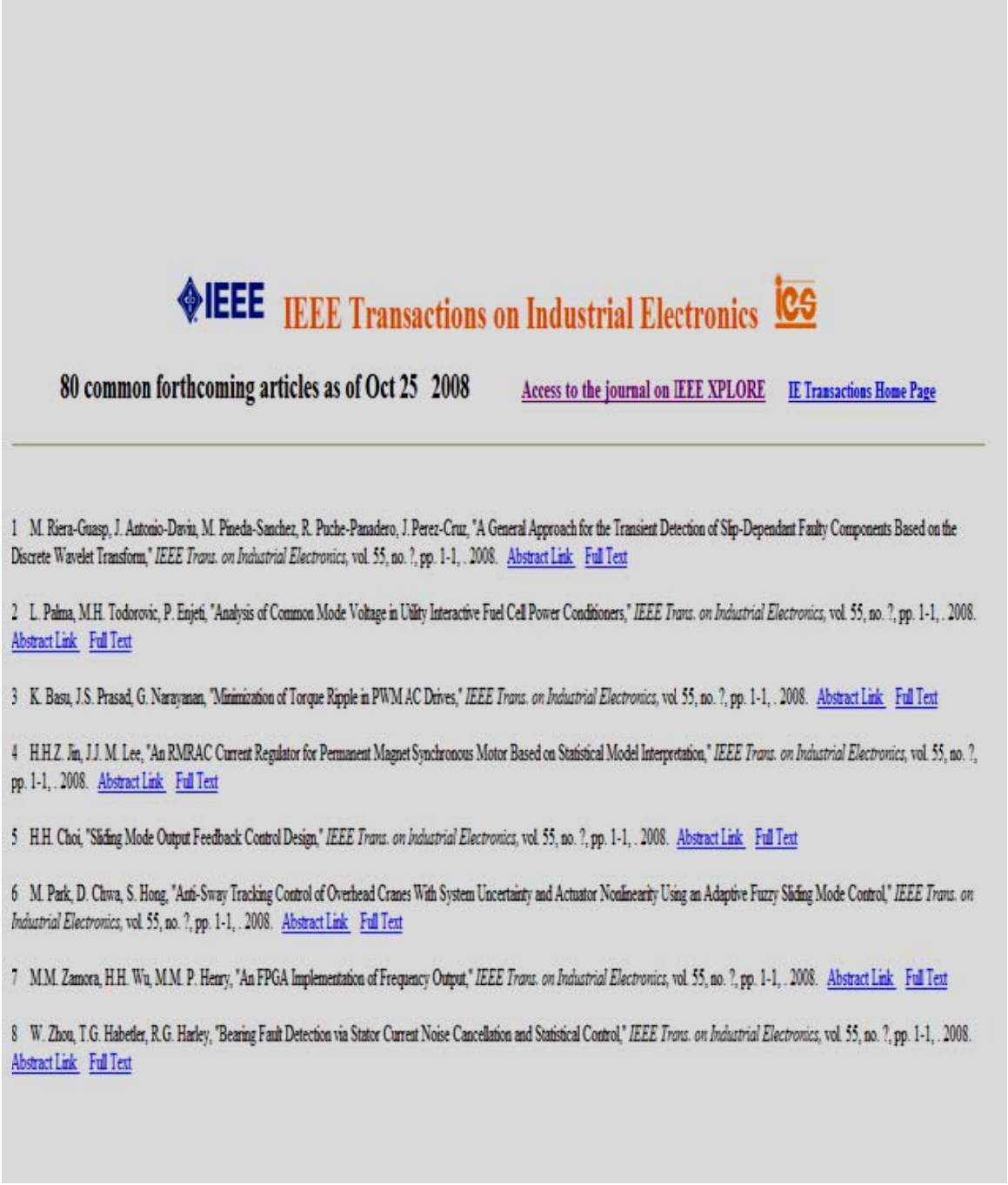
4 C.C.-Y. Wu, T.T.-F. Wu, J.J.-R. Tsai, Y.Y.-M. Chen, C.C.-C. Chen, "Multi-String LED Backlight Driving System for LCD Panels With Color Sequential Display and Area Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

5 A.A.G. Espinosa, J.J.-R. R. Ruiz, J.J. Cusido, X.X.A. Morera, "Sensorless Control and Fault Diagnosis of Electromechanical Contactors," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 A.A. Di Gerlando, G.G. Foglia, R.R. Perini, "Permanent Magnet Machines for Modulated Damping of Seismic Vibrations: Electrical and Thermal Modelling," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 B.B. Castillo-Toledo, S.S. Di Gennaro, A.A. G. Loukianov, J.J. Rivera, "Hybrid Control of Induction Motors via Sampled Closed Representations," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 8. Old papers.



IEEE Transactions on Industrial Electronics

80 common forthcoming articles as of Oct 25 2008 Access to the journal on IEEE XPLORE IE Transactions Home Page

1 M. Riera-Guasp, J. Antonio-Daviu, M. Pineda-Sanchez, R. Puche-Panadero, J. Perez-Cruz, "A General Approach for the Transient Detection of Slip-Dependant Faulty Components Based on the Discrete Wavelet Transform," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 L. Palma, M.H. Todorovic, P. Enjeti, "Analysis of Common Mode Voltage in Utility Interactive Fuel Cell Power Conditioners," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 K. Basu, J.S. Prasad, G. Narayanan, "Minimization of Torque Ripple in PWM AC Drives," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

4 H.H.Z. Jin, J.J. M. Lee, "An RMRAC Current Regulator for Permanent Magnet Synchronous Motor Based on Statistical Model Interpretation," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

5 H.H. Choi, "Sliding Mode Output Feedback Control Design," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 M. Park, D. Chwa, S. Hong, "Anti-Sway Tracking Control of Overhead Cranes With System Uncertainty and Actuator Nonlinearity Using an Adaptive Fuzzy Sliding Mode Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 M.M. Zamora, H.H. Wu, M.M. P. Henry, "An FPGA Implementation of Frequency Output," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

8 W. Zhou, T.G. Habetler, R.G. Harley, "Bearing Fault Detection via Stator Current Noise Cancellation and Statistical Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 9. Common papers.

2.3.4 Apply the Internet robot to replace Microsoft-front page software.

In our webpage (Fig. 5), there are many different links displaying the collected paper in different groups of year such as from 1988 to 1999, from 2000 to 2006 or form

2007 to 2008, in order to manage these database more efficiently in this particular case we can use the Internet Robot to replace for “Microsoft Front-page” software which you have to append those files together manually. This procedure of the internet robot will save a lot of time. Specially, when there are some errors need to be corrected in those files. (see more in APPENDIX D).

CHAPTER 3

IMPLEMENTATION OF THE INTERNET ROBOT IN PERL

Chapter 2 introduced the overall view of the Internet Robot in extracting data from IEEE Xplore as well as other applications of the Internet Robot we develop so far (see more detail in appendixes). So now we will get in more details to see how it exactly works in this chapter by analyzing one sample Robot.

3.1 The control text file.

All our programs in extracting paper information and designing this webpage start with the control text file as figure 4 (chapter 2). The control text file includes in all necessary information to download an issue. Concretely, journals have different code numbers, Maxissue is modified as what the maximum issue number is in one volume, the other parameters such as FROMissue, FROMyear, TOissue, TOyear relate to how many issues of the volumes that we want to download. Such as in this case, the program will download from the issue number 1 of year 1988 to the issue number 10 of year 2008. Other parameters involve in where html files will be generated and what kind of presentation we want as well.

Whenever these parameters will be changed in this control text file we can download paper information of all issues of all Journals or Magazines from IEEE Xplore as we want.

```
$textfile="Loop_issue.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
  chop $array[$v];
  $leg=length($array[$v]);

   for($u=0;$u<50;++$u)
     {
      $index=rindex($array[$v],$blank);
      if (($index==-1)||($index!=$leg-1))
        {
         last;
        }
      $array[$v]=substr($array[$v],0,$index);
      $leg=length($array[$v]);
     }

  }
```

The parameters in the control text file will be read into the array @array. However, these input parameters are not supposed in the same format in every time the users type in. Because of this reason these inputs need to be kept in the fixed form independently on the users. This is very important when we want to keep our presentation in the same form for all issues as well as we can use these generated files to compare all

issues together to extract the duplicated papers for our another purpose. For example, code=41 without any blanks after it should be the desired format but for some reason it is typed in code=41 . Therefore, we have to delete all the blanks after number 41. This is an easy case that our program can realize the blanks and delete them until no blanks are left and the array @array is modified again with the new format. But in other parameters like dir, shortcut and journal…there are many blanks that are not only at the end of the parameters but also between words of the parameter, so our program has to be able to distinguish what type of blank between these two cases. In other words, if every time it runs and recognizes a blank after a parameter, the blank will be cut off. Once there is not any blank left after the parameter it will automatically jump to the next parameter and execute the same procedure again. We know that all the blanks after the parameters always have a blank or a character in front of it but the blanks in the parameter are bounded by two characters. And the above program does exactly as what have just been described here.

```
@z1=split(/=/,$array[0]);
@z2=split(/=/,$array[1]);
@z3=split(/=/,$array[2]);
@z4=split(/=/,$array[3]);
@z5=split(/=/,$array[4]);
@z6=split(/=/,$array[5]);
@z7=split(/=/,$array[6]);
@z8=split(/=/,$array[7]);
@z9=split(/=/,$array[8]);
@z10=split(/=/,$array[9]);

%hash=($z1[0] => $z1[1],
       $z2[0] => $z2[1],
       $z3[0] => $z3[1],
       $z4[0] => $z4[1],
```

```
        $z5[0] => $z5[1],
        $z6[0] => $z6[1],
        $z7[0] => $z7[1],
        $z8[0] => $z8[1],
        $z9[0] => $z9[1],
        $z10[0] => $z10[1],);

$Tyear=$hash{TOyear};
$Fyear=$hash{FROMyear};
$IS=$hash{TOissue};
$beginIS=$hash{FROMissue};
$mydir=$hash{directory};
$pnum=$hash{code};
$startYear=$hash{startYear};
$journal=$hash{journal};
$shortcut=$hash{shortcut};
$Maxissue=$hash{MAXissue};
```

The parameter format is equal to “string=string”. The first string is just an unchanged name to define what kind of input, only the second string that has information is needed for our purpose. The “split” function is used to extract the second string. By using this function, two strings will be copied into the first and second element of an array. In this case there are ten input parameters, so there will be ten arrays from z1 to z10. At this point, the elements of these arrays can be valid to execute the program if they are arranged in the fixed order so that their values can be assigned for the variables correctly. In other words, these arrays’ content will depend on the order of the input parameters. If this order is changed, the values of the variables will change and the program can’t execute. However, using “hash” can help us get rid of this problem. Hash can be considered as an array that has relationship between its elements. If we know the first element we can know what the second element is by defining their connection. For example, assume that $z1[0]=”code”, $z1[1]=”41” and %hash=($z1[0] => $z1[1],), we

know that $z1[0] is a fixed string while $z1[1] is a changing number which depends on what journal or magazine is. By assigning $pnum=$hash{code}; the returned value of $pnum is always equal to an integer value of $z1[1] no matter where that string is put in the control text file.

3.2 Checking the errors.

```
$long_dir=length($mydir);
$mark="/";
$check_mark=rindex($mydir,$mark);
if($check_mark!=$long_dir-1){print("you forgot the forward slash (/) of the directory\n");}
$long_startYear=length($startYear);

for($i_pnum=0;$i_pnum<1;++$i_pnum)
 {
  if($pnum =~ m/(\D+)/ )
    { print("you need to put a code in numbers not in words\n");  last; }

  if(($Maxissue<$beginIS)||($Maxissue<$IS))
    { print("Please check the maximum number of issue in textfile");  last;}

  if($Tyear<$Fyear)
    { print("please check number of years you want to extract\n");  last; }

  if($long_startYear!=4)
    { print("please check startyear input\n"); last; }
….
        ….
 }
```

Users sometimes have difficult time to find out why the program doesn't work correctly even though it does not display any syntax error. There are some common errors that users often make such as inputting the wrong code with the combination of the integers and the characters or the wrong directory without a forwarding slash, years are

not a 4 digit number etc. Because of these reasons, the program should be able to check these common errors before execution. Once the program executes and meets one of these errors, it will automatically pop up a message to tell users why it has this error and stops running. This is the fastest way to correct the errors without any confusion. Otherwise, the users have to take a lot of time but don't know why, especially the users don't know much about Perl.

After checking all the input parameters and they are fine, the program will start to execute. In this particular case of IEEE Transactions on Industrial Electronics, there are maximum twelve issues in one volume and published monthly, but some volumes have only 4 or 5 or 6 issues in a year. That is the reason Maxissue is included in the control text file. Let's say we want to download all papers from the first issue no. 1 of year 1988 to the last issue no. 10 of year 2008 as in the control text file above. In the program below, the outer for loop is used to fetch volume by volume while inner for loop is used to fetch issue by issue in a volume. Whenever there are enough inputs as the code number, the volume number and the issue number, the variable $addr with the full link of an issue will be called by a subroutine &get_content. As explained in the Chapter 1, this subroutine will link to that address and copy its web source except java script into the array @lines due to the support of Wget.exe. This subroutine will return the new array @lines back into the main program and assign it for another array @get_issue. As mentioned in the beginning of this paragraph, some volumes do not have enough 12 issues so there are some invalid issue numbers that refer the variable $addr to the wrong addresses. To discard this case, there should be a checking condition by comparing the length of the returned value from the subroutine &get_content with 150 (

$Nget_issue=@get_issue; if ($Nget_issue<150){last;}). If this length is shorter than 150, it means that this is the invalid address and jumps out the loop to continue searching for the next issue. Additionally, this is a flexible program that the users can download a single issue or many issues in the same volume or in the different volumes. Its function will depend on the way users define the input parameters.

```
 my $V= $Tyear - $startYear;   # volume
$a="http://ieeexplore.ieee.org/xpl"; # partial link of other articles in an issue
$b="http://ieeexplore.ieee.org/xpls"; #partial link of abstract
$c="http://ieeexplore.ieee.org/Xplore/login.jsp?url="; #partial link of PDF

 @split_array=("","","","",);
 $countyear=0;

 for ($startV=$Fyear-$startYear;$startV<=$V;++$startV)
   {
      if ($startV==$V){$Maxissue=$IS;}
      $startyear=$Fyear+$countyear;
      $countyear=$countyear+1;

      for ($startIS=$beginIS;$startIS<=$Maxissue;++$startIS)
       {
          if($startIS==1) {$month="Jan";}
          if($startIS==2) {$month="Feb";}
          if($startIS==3) {$month="March";}
          if($startIS==4) {$month="April";}
          if($startIS==5) {$month="May";}
          if($startIS==6) {$month="June";}
          if($startIS==7) {$month="July";}
          if($startIS==8) {$month="August";}
          if($startIS==9) {$month="Sept";}
          if($startIS==10){$month="Oct";}
          if($startIS==11){$month="Nov";}
          elsif($startIS==12){$month="Dec";}
           $addr=
          "http://ieeexplore.ieee.org/servlet/opac?punumber=$pnum&isvol=$startV
          &isno=$startIS"; #full link of an issue
          my @get_issue=&get_content($addr); #get content of page of issue
          $Nget_issue=@get_issue;
```

```
            if ($Nget_issue<150){last;}
           ….
        }
    ….
  }
```

Sub routines

```
sub get_content
{
   $add=$_[0];  # all incoming input
   $fname= "med.htm";
   system("wget.exe", "-q", "-O", $fname,$add);
   $file=$mydir."med.htm";
   open(r1,"<$file");
   @lines = <r1>;
   close(r1);
   unlink($fname); # delete temporary file
   return(@lines);
}
```

3.3 Copy web source into an array.

```
my @get_issue=&get_content($addr); #get content of page of issue
$Nget_issue=@get_issue;
if ($Nget_issue<150){last;}
my $temp=&add_issue(@get_issue);  #add content of all pages in the same file
my @clean_file=&get_href(@get_issue); #clean to have file start with <a HREF
$p_issue=@clean_file;
$page_issue=$p_issue/2;

for($i=0;$i<$page_issue;++$i)
  {
    @filter=split(/ class/,$clean_file[$i]);
    @filter=(@filter,@split_array);
    @filter1=split(/xpl/,$filter[0]);
    @filter1=(@filter1,@split_array);
    $link=$a . $filter1[1];
    my @get_issue=&get_content($link);
    my $temp1=&add_issue(@get_issue);
  }
```

```
$file=$mydir."ISS_No.htm";
open(K1,"<$file");
@Lines=<K1>;
close(K1);
```

Subroutines:

```
sub add_issue  #add content of all pages
{
   $file=$mydir."ISS_No.htm";
   open(K,">>$file");
   print K "@_";
   close(K);
   return 0;
}


sub get_href
{
  $N=@_;
  @new=("");
  $j=0;
  for ($i=0;$i<$N;++$i)
   {
    if ($_[$i]=~ m/<a HREF=/)
      {
       $new[$j]=$_[$i];
       $j=$j+1;
      }
   }
  return(@new);
}
```

For simplicity, from here we can consider how the Internet Robot works with one issue of a certain volume because other issues of other volumes almost work in the same way. From the above program we know that content of the array @get_issue is the web source of the address $addr but there are many papers of one issue staying under different links, so it is a good idea that we should create a html file ISS_No.html to store this web

source. Once we get the content of other papers in this issue we just append it into this ISS_No.html file. As just pointed out, we only have some first papers of an issue and still miss the rest of them. In order to search other papers we have to fetch their address links first and the procedure to do it is to transfer the array $get_issue into the subroutine &get_href. This subroutine is written with a for loop to search every element in this array. Every time the expression in the array element contains the key word "<a HREF=" by using the regular expressions, that expression will be stored into the new array @new. The reason we have to compare the expression with "<a HREF=" because those are the only ones containing the addresses of other papers. After the for loop is done, the array @new with some elements having the key words"<a HREF=" will be returned back into the main program. Because every expression in the array @new has its own duplicate but we only need to use a first half of this array to extract the links of other papers. That is the reason why we have to initialize $page_issue = $p_issue/2. By using another for loop according with the split function and concatenation function we can fetch the links of the rest paper. Once we have these links, using the subroutines &get_content and &add_issue to copy and append the web source into the ISS_No.html file. Up to this point, we can say all our necessary information stays in this html file.

To extract our desired information, the ISS_No.html should be opened and copied into the array @Lines. Because of the template of the IEEE links, the titles, the authors, the page numbers, the abstract links, the full text links and the abstract contents will be extracted respectively. The only procedure of extracting this information is to fetch the element in the array @Lines which contains our desired information first by using the logical conditions and the regular expressions, whenever this step is done we will apply

some functions in Perl Script to extract the specific details that is needed for our purpose. Because the authors' name of some papers in IEEE Transactions on Industrial Electronics are not posted, this changes the template and therefore we have to change our program a little bit to match with but it basically works in the same way. To simplify it, we don't concern it right now and you can see more details in the appendix.

```
$file=$mydir."ISS_No.htm";
open(K1,"<$file");
@Lines=<K1>;
close(K1);
$N1=@Lines;
@add_link="";
@each_abst="";
$jj=0;
$str_title="strong";

for ($j=0;$j<$N1;++$j)
        {
         if (($Lines[$j]=~ m/strong><br/)&&($Lines[$j]=~ m/<td class=/))
                                        # search the title
          {
            $start= index($Lines[$j],$str_title);
            $ad=$start+7;
            $length_string= length($Lines[$j]);
            $title= substr($Lines[$j],$ad,$length_string-14-$ad);

             if ($Lines[$j+2]!~ m/Page(\D+)s(\D+): /) #search author
              {
               $length_author=length($Lines[$j+2]);
               $aut=substr($Lines[$j+2],13,$length_author-18);
               $aut=~ s/(\; , .)//gi;
               @seperate_aut=split(/;/,$aut);
               $N_a=@seperate_aut;
               $author="";
               $#connect=-1;

                for($i_a=0;$i_a<$N_a;++$i_a)
                {
                   @f=split(/,/,$seperate_aut[$i_a]);
                   @f=(@f,@split_array);
```

```
            if($i_a>0)
              {
                $long_f=length($f[0]);
                $f[0]=substr($f[0],1,$long_f);
              }
            $connect[$i_a]=$f[1]." "."$f[0],";
            $author=$author . $connect[$i_a];
          }

          }

        if ($Lines[$j+2]=~ m/Page(\D+)s(\D+): /) #search page
          {
           $length_page=length($Lines[$j+2]);
           $page=substr($Lines[$j+2],23,$length_page-28);
          }
....
....
}
```

3.4 Extract the titles.

The array element $Lines[$j] containing the title is searched by comparing it with the key words “<td class= “and “
”. The element that has these key words is the element we are looking for. The expression structure below is unchanged except the title. Using the “index” function to define the position of the first “strong” (there are two words “strong” in the expression) and $ad=$start+7 lets us know the starting point of the title as well as we know the unchanged length of string “
” which is equal to 14 in this particular case. By subtracting the length of whole expression by $ad+14 we can define the ending point of the title. Then substring the original string from the starting point to the ending point of the tile, the title information will be extracted and saved into the variable $title.

$Lines[$j]=”<td class="bodyCopyBlackLargeSpaced">**Title**
”;

```
$start= index($Lines[$j],$str_title);
$ad=$start+7;
$length_string= length($Lines[$j]);
$title= substr($Lines[$j],$ad,$length_string-14-$ad);
$title=Title
```

3.5 Extract the authors.

In chapter one, we discussed about our strategy how to extract the rest of desired information by referring to the title that we will not repeat it again. When the title expression is fetched, the second expression from it will contain the author information. Still redoing the same procedure is to fetch first and to extract later we can extract full information of the authors of papers. Assume that the variable $Lines[$j+2] which is the expression having the following content.

Ex: $Lines[$j+2]=” Nerg, J.; Rilla, M.; Pyrhonen, J. ;,.
”;

Using the substring function, the authors can be extracted easily and marks of ;,. should be replaced by the blanks to keep the authors’ name in the good format. Nevertheless, the format of IEEE Transactions on Industrial Electronics is different from the one we want that users can cite papers directly through our webpage. We know that the names are separated by the semicolons, so @seperate_aut=split(/;/,$aut) will split different names by semicolons and store them into the array @seperate_aut. But we have

not finished yet we have to invert between the last name and the first name of each author before rearranging them. By reusing split function but now we split the last name and the first name by a comma @f=split(/,/,$seperate_aut[$i_a]);. At this time, the last name and the first name will be the first element and second element of the array @f, respectively and recombined by $connect[$i_a]=$f[1]." "."$f[0],"; for example if an author's name is Nerg, J. now it is invert into J. Nerg. A for loop with the length is equal to how many authors is applied to invert the last name and the first name of an author, then will recombine them together. The final string will be as following J. Nerg, M. Rilla, J. Pyrhonen,

```
if ($Lines[$j+2]!~ m/Page(\D+)s(\D+): /) #search author
      {
       $length_author=length($Lines[$j+2]);
       $aut=substr($Lines[$j+2],13,$length_author-18);
       $aut=~ s/(\; , .)//gi;
       @seperate_aut=split(/;/,$aut);
       $N_a=@seperate_aut;
       $author="";
       $#connect=-1;

        for($i_a=0;$i_a<$N_a;++$i_a)
        {
           @f=split(/,/,$seperate_aut[$i_a]);
           @f=(@f,@split_array);
           if($i_a>0)
             {
               $long_f=length($f[0]);
               $f[0]=substr($f[0],1,$long_f);
             }
           $connect[$i_a]=$f[1]." "."$f[0],";
           $author=$author . $connect[$i_a];
        }

       }
```

3.6 Extract the page numbers.

If the element in the array containing information of the authors is allocated from the previous step, the second element from it will be the expression storing information of the page numbers. It depends on whether the author names are posted or not the order of this expression will change comparing with the expression of the titles as you can see below. This is not important as we assumed in the beginning that full information of papers is posted so we don't have to worry about its general format.

```
if ($Lines[$j+2]=~ m/Page(\D+)s(\D+): /) #search page
{
 $length_page=length($Lines[$j+2]);
 $page=substr($Lines[$j+2],23,$length_page-28);
}
elsif ($Lines[$j+4]=~ m/Page(\D+)s(\D+): /)
{
 $length_page=length($Lines[$j+4]);
 $page=substr($Lines[$j+4],23,$length_page-28);
}
```

For example, $Lines[$j+4]= " Page(s): C1-3497
". This expression is fetched by comparing its content with Page(s): by the regular expressions. Because all the key words in this expression don't change and have a fixed length, so we can substring the page numbers as we did before. By hand calculation, the index numbers of the starting point and the ending point of the page number are known. Once we know the index numbers we can filter out other unnecessary words. The final result will be $page=" C1-3497".

3.7 Extract the abstract links and the abstract description.

```
if ($Lines[$j]=~ m/>Abstract(\D+)a>/)  # search abstract link of each paper
     {
      $length_abst=length($Lines[$j]);
      $abstract=substr($Lines[$j],31,$length_abst-69);
      $link_abst=$b . $abstract;
      my @get_abstract= &get_content_ab($link_abst);
      my $abst_content= &get_abstractspan(@get_abstract);
      $each_abst[$jj]= $abst_content;
     }
```

Subroutine

```
 sub get_content_ab
 {
  $jjj=20;
  do{
    $add=$_[0];  # all incoming input
    $fname= "med.htm";
    system("wget.exe", "-q", "-O", $fname,$add);
    $file=$mydir."med.htm";
    open(med1,"<$file");
    @lines = <med1>;
    close(med1);
    unlink($fname); # delete temporary file
    $jjj=$jjj-1;
    $N=@lines;

    for ($i=0;$i<$N;++$i)
      {
       if ($lines[$i]=~ m/>Abstract<(\D)span>/) {last; }
      }

  }while (($lines[$i+1]!~ m/(\w)/)&&($jjj>1))

  return(@lines);
 }

 sub get_abstractspan
  {
   $N=@_;
   for ($i=0;$i<$N;++$i)
```

```
{
 if ($_[$i]=~ m/>Abstract<(\D)span>/)
   {
    $abstrvar=$i;
    last;  #break
   }
}
$feedback=" ";

for ($th=1;$th<100;++$th)
   {
    if ($_[$abstrvar+$th] !~ m/<(\D)td>/){ $feedback=$feedback .
$_[$abstrvar+$th];}
    elsif ($_[$abstrvar+$th] =~ m/<(\D)td>/) { last;}
   }

  return ($feedback);
}
```

The element containing the abstract link has the unique key word Abstract, in order to fetch it we don't have to concern about its order relative to other elements. Although this element has the necessary information of the abstract link but it is not a full link that can be used directly for the presentation. Therefore we have to do two things. The first thing is to extract the necessary information. Once we have this, we can concatenate it with the partial link $b="http://ieeexplore.ieee.org/xpls" that we create by ourselves to have $link_abst=$b.$abstract. However our webpage needs to have not only the abstract link but also the abstract content. The subroutine &get_content_ab is called with the $link_abst input to copy all the web sources and return them back into the array @get_abstract. In reality, we face an uncomfortable problem that the abstract content of some papers are not copied into @get_abstract by Wget.exe. This cause can be explained by the internet traffic to IEEE Xplore. Assume that if this problem is not fixed, we have

to run the program many times and then copy and paste the abstract content manually into the papers which miss them. To get rid of this trouble, a do-while loop is added into the subroutine &get_content_ab to check whether the abstract content is available or not by searching the key words Abstract<\span>. If the array @lines has this key word, it means that the abstract content is copied by Wget.exe and the subroutine returns its value right away at that point. Otherwise, the process of the subroutine will be repeated maximum 20 times until this key word can be recognized. On the other hands, after 20 times if this key word isn't found out yet, the process of the subroutine &get_content_ab will stop and return its value. By using this simple trick, we can come over the traffic jam to IEEE Xplore.

To search the abstract content of a paper, another subroutine &get_abstractspan is called. This subroutine fetches the paper abstract and returns it back into the variable $abst_content of the main program by fetching the next element after the element containing the expression "Abstract
. However, the paper abstract is a long paragraph and will be copied into different elements of the array, so we have to concatenate them to have a full description. This is done by using a for loop as we saw above. The for loop will concatenate element by element and stop when it recognizes the element containing key words </td> which is the ending point of the paper abstract.

…… Abstract

<para> paper abstract</para>

</td>

3.8 Extract the full-text links.

```
if ($Lines[$j]=~ m/>PDF(\D+)a>/)    # search PDF link
  {
   @cut=split(/PDF/,$Lines[$j]);
   @cut=(@cut,@split_array);
   $length_pdf= length($cut[0]);
   $pdf=substr($cut[0],46,$length_pdf-71);
   $link_pdf=$c . $pdf;
   $link_pdf=~ s/Xplore(\W)login.jsp(\W)url(\W+)//;
   $link_pdf=~ s/pdf(\?)isnumber/pdf?tp=&isnumber/;
   $link_pdf=~ s/prod=JNL&//;
   $link_pdf=~ s/&arSt(.+)//;
  }
```

The way we extract the full text link is almost the same as the way we did with the abstract link. First, we fetch the element having information of the full text link by searching for the key words “>PDF</a”, once this element is fetched we use the split function and the substring function to extract our desired data. Finally, we concatenate this data with the partial link $c= "http://ieeexplore.ieee.org/Xplore/login.jsp?url=" that we created to have the PDF link $link_pdf. However, this link is not active yet, but we can use the regular expressions to make this link become active if you are a subscriber of IEEE Xplore by substituting the undesired expressions by the desired expressions.

The information of the authors’ name, the title, the page number, the abstract link, the abstract content and the abstract link is saved into the variables $author, $title, $page, $link_abst, $link_pdf and $each_abst respectively. These variables are combined to present in the html and its result will be stored into the arrays @add_link and @no_abstract. The process continues until information of the last paper is extracted.

Once the process is finished, the presentation of each paper will be stored into each element of the arrays @add_link and @no_abstract while @add_link has the abstract description but @no_abstract doesn't have.

3.9 Presentation.

Up to this point we extracted and presented all necessary information of the papers of an issue in the arrays @add_link and @no_abstract. While each element of these arrays will include in information of each paper presented in HTML -Hypertext Markup Language-format as Fig1.

```
$add_link[$jj]=
"<html><tr>
<td
valign=\"top\">$startV.$startIS.$count   </td><td>$author&n
bsp;\"$title,\"<i> $shortcut,</i> vol. $startV, no. $startIS, pp. $page, $month
$startyear.  
<a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
<a href=$link_pdf><font color=\"blue\"> Full Text</font></a><br><br>
<font size=\"3\"
color=\"blue\"><b>Abstract:</b>$each_abst[$jj]</font><br><br>
</td></tr>
</html>";

$no_abstract[$jj]=
"<html><tr>
<td
valign=\"top\">$startV.$startIS.$count   </td><td>$author&n
bsp;\"$title,\"<i> $shortcut,</i> vol. $startV, no. $startIS, pp. $page, $month
$startyear.  
<a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
<a href=$link_pdf><font color=\"blue\"> Full Text</font></a><br><br>
</td></tr>
```

</html>”;

These arrays are added and appended into three different html files while one file for an issue and two other files for a volume with the abstract content and without the abstract content which you can append when a new file of the new issue is generated. In order to have a good web page, we have to combine these data with HTML which is the commonly used for network programming that can be seen with more details in the appendix.

```
$volume=”$startV”.”_”.”$startIS”;
$file=$mydir.”$volume.htm”;# abstract
open(Kvolume,”>$file”);
print Kvolume “$head <table>”;
print Kvolume “@add_link”;
print Kvolume “</table>”;
close(Kvolume);


$file=$mydir.”$startV”.”s.htm”;  #append no abstract
open(Kstart,”>>$file”);
print Kstart “$head <table>”;
print Kstart “@no_abstract”;
print Kstart “</table>”;
close(Kstart);

$file=$mydir.”$startV.htm”;   # append abstract
open(Kstartv,”>>$file”);
print Kstartv “$head <table>”;
print Kstartv “@add_link”;
print Kstartv “</table>”;
close(Kstartv);
```

Fig. 10. Running process from black screen.

CHAPTER 4

CONCLUSION

This method of data extraction, the Internet Robot, optimizes data processing and makes it become a unique tool in extracting data from the web-servers. Perl script with the regular expressions and the emulating web browsers increases the speed of data extracting as well as the accuracy. The Internet Robot is customized according to the required data and the format of data that the users desire. Up to this point, the Internet Robot which we have developed can update our webpage (Fig. 5) automatically and download all information of the journal and magazine papers as well as the conference papers from IEEE Xplore.

In order to manage all our webpage automatically we had to develop some other Internet Robots that we include in appendixes. These programs can be used to optimize our work by replacing for other soft-wares such as Microsoft front-page…

Perl Script is a powerful language particularly in data processing as we saw in this thesis which makes it become a simpler tool compared with C or C^{++} or other languages. Moreover, programmers can realize that Perl Script has a lot of functions as network programming (CGI), database management…as well as a hundreds of applicable modules developed by the open-source community which they can reuse or contribute.

BIBLIOGRAPHY

[1] S. Neeli, K. Govindasamy, B.M. Wilamowski, A. Malinowski, " Auto Data Mining from Web Servers Using Perl Script", *International Conference on Intelligent Engineering System 2008 (INES 2008)*, pp. 191-196, Feb 25-29, 2008.

[2] M. Keyed, Chia-Hui Chang, K, Shaalan, M.R. Girgis, " FiVa Tech: Page-Level Data Extraction from Template pages", *7th IEEE International Conference on Data Mining Workshops 2007 (ICDM Workshops 2007)*, pp.15-20, Oct 28-31, 2007.

[3] C.H. Chang, M. Kayed, R. Girgis, K.F. Shaalan, " A Survey of Web Information Extraction Systems", *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp.1411-1428, Oct 2006.

[4] Steven Holzner, " PERL, Black book", Edition 1999.

[5] http://www.cpan.org/modules/01modules.index.html.

[6] Aleksander Malinowski and Bogdan Wilamowski, " Internet Technology as a Tool for Solving Engineering Problems", *The 27th Annual Conference of the IEEE Industrial Electronics Society* (tutorial) pp. 1622-1630, Denver CO, Nov 29-Dec 2, 2001.

[7] http://www.gnu.org/software/wget/manual/wget.html.

[8] http://ieeexplore.ieee.org/xpl/opac.jsp.

[9] I-Chen Wu, Jui-Yuan Su, Loon-Been Chen," A Web Data Extraction Description Language and Its Implementation", *29th Annual International conference on Computer Software and Applications Conference 2005 (COMPSAC 2005)* , vol2, pp. 293-298, July 25-28, 2005.

APPENDICES

PERL CODES OF DATA EXTRACTION FROM IEEE XPLORE

APPENDIX A: Loop_issue.pl (perl program)

```
#!/usr/bin/perl -w
#use strict;

$textfile="Loop_issue.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
  chop $array[$v];
  $leg=length($array[$v]);
   for($u=0;$u<50;++$u)
     {
       $index=rindex($array[$v],$blank);
       if (($index==-1)||($index!=$leg-1))
         {
          last;
         }
       $array[$v]=substr($array[$v],0,$index);
       $leg=length($array[$v]);
     }
 }

@z1=split(/=/,$array[0]);
@z2=split(/=/,$array[1]);
@z3=split(/=/,$array[2]);
@z4=split(/=/,$array[3]);
@z5=split(/=/,$array[4]);
@z6=split(/=/,$array[5]);
@z7=split(/=/,$array[6]);
@z8=split(/=/,$array[7]);
@z9=split(/=/,$array[8]);
@z10=split(/=/,$array[9]);


%hash=($z1[0] => $z1[1],
            $z2[0] => $z2[1],
            $z3[0] => $z3[1],
```

```
            $z4[0] => $z4[1],
            $z5[0] => $z5[1],
            $z6[0] => $z6[1],
            $z7[0] => $z7[1],
            $z8[0] => $z8[1],
            $z9[0] => $z9[1],
            $z10[0] => $z10[1],);

$Tyear=$hash{TOyear}; # year
$Fyear=$hash{FROMyear};
$IS=$hash{TOissue}; #date
$beginIS=$hash{FROMissue};
$mydir=$hash{directory};#directory
$pnum=$hash{code};   #code of a journal
$startYear=$hash{startYear};   #year for vol 1 of the journal
$journal=$hash{journal};
$shortcut=$hash{shortcut};
$Maxissue=$hash{MAXissue};

          #check input errors
$long_dir=length($mydir);
$mark="/";
$check_mark=rindex($mydir,$mark);
if($check_mark!=$long_dir-1){print("you forgot the forward slash (/) of the
directory\n");}
$long_startYear=length($startYear);

for($i_pnum=0;$i_pnum<1;++$i_pnum)
  {
    if($pnum =~ m/(\D+)/ )
    {
      print("you need to put a code in numbers not in words\n");
      last;
    }
    if(($Maxissue<$beginIS)||($Maxissue<$IS))
    {
      print("Please check the maximum number of issue in textfile");
      last;
    }



    if($Tyear<$Fyear)
    {
      print("please check number of years you want to extract\n");
```

```
  last;
 }
 if($long_startYear!=4)
 {
  print("please check startyear input\n");
  last;
 }

  my $V= $Tyear - $startYear;   #volume
  $a="http://ieeexplore.ieee.org/xpl"; #patial link of other articles in an issue
  $b="http://ieeexplore.ieee.org/xpls";#patial link of abstract
  $c="http://ieeexplore.ieee.org/Xplore/login.jsp?url=";#patial link of PDF
  @split_array=("","","","","");

  $countyear=0;

  for($startV=$Fyear-$startYear;$startV<=$V;++$startV)
   {
    if ($startV==$V){$Maxissue=$IS;}
    $startyear=$Fyear+$countyear;
    $countyear=$countyear+1;

    for($startIS=$beginIS;$startIS<=$Maxissue;++$startIS)
    {
     if($startIS==1) {$month="Jan";}
     if($startIS==2) {$month="Feb";}
     if($startIS==3) {$month="March";}
     if($startIS==4) {$month="April";}
     if($startIS==5) {$month="May";}
     if($startIS==6) {$month="June";}
     if($startIS==7) {$month="July";}
     if($startIS==8) {$month="August";}
     if($startIS==9) {$month="Sept";}
     if($startIS==10){$month="Oct";}
     if($startIS==11){$month="Nov";}
     elsif($startIS==12){$month="Dec";}

       $addr=
"http://ieeexplore.ieee.org/servlet/opac?punumber=$pnum&isvol=$startV&isno=$startIS
"; #full link of an issue

       my @get_issue=&get_content($addr); #get content of page of issue
       $Nget_issue=@get_issue;
       if ($Nget_issue<150){last;}
        my $temp=&add_issue(@get_issue);  #add content of all pages in the same file
```

```
my @clean_file=&get_href(@get_issue); #clean to have file start with <a HREF
$p_issue=@clean_file;
$page_issue=$p_issue/2;

for($i=0;$i<$page_issue;++$i)
  {
    @filter=split(/ class/,$clean_file[$i]);
    @filter=(@filter,@split_array);
    @filter1=split(/xpl/,$filter[0]);
    @filter1=(@filter1,@split_array);
    $link=$a . $filter1[1];
    my @get_issue=&get_content($link);
    my $temp1=&add_issue(@get_issue);
  }

$file=$mydir."ISS_No.htm";
open(K1,"<$file");
@Lines=<K1>;
close(K1);
$N1=@Lines;
@add_link="";
@each_abst="";
$jj=0;
$str_title="strong";

for ($j=0;$j<$N1;++$j)
{
 if (($Lines[$j]=~ m/strong><br/)&&($Lines[$j]=~ m/<td class=/)) # search the
title
  {
  $start= index($Lines[$j],$str_title);
  $ad=$start+7;
  $length_string= length($Lines[$j]);
  $title= substr($Lines[$j],$ad,$length_string-14-$ad);

   if ($Lines[$j+2]!~ m/Page(\D+)s(\D+): /) #search author
     {
      $length_author=length($Lines[$j+2]);
      $aut=substr($Lines[$j+2],13,$length_author-18);
      $aut=~ s/(\; , .)//gi;
      @seperate_aut=split(/;/,$aut);
      $N_a=@seperate_aut;
      $author="";
      $#connect=-1;
```

```
                for($i_a=0;$i_a<$N_a;++$i_a)
                {
                   @f=split(/,/,$seperate_aut[$i_a]);
                   @f=(@f,@split_array);
                   if($i_a>0)
                    {
                     $long_f=length($f[0]);
                     $f[0]=substr($f[0],1,$long_f);
                    }
                   $connect[$i_a]=$f[1]." "."$f[0],";
                   $author=$author . $connect[$i_a];
                }

                }

               if ($Lines[$j+2]=~ m/Page(\D+)s(\D+): /) #search page
                 {
                  $length_page=length($Lines[$j+2]);
                  $page=substr($Lines[$j+2],23,$length_page-28);
                 }

               elsif ($Lines[$j+4]=~ m/Page(\D+)s(\D+): /)
                 {
                  $length_page=length($Lines[$j+4]);
                  $page=substr($Lines[$j+4],23,$length_page-28);
                 }
              }

            elsif (($Lines[$j]=~ m/strong><br/)&&($Lines[$j]!~ m/<td class=/))     # search
the title
             {
              $Lines[$j]=$Lines[$j-1] . $Lines[$j];
              if ($Lines[$j]!~ m/<td class=/)
               {
                $Lines[$j]=$Lines[$j-2] . $Lines[$j];
               }
              $start= index($Lines[$j],$str_title);
              $ad=$start+7;
              $length_string= length($Lines[$j]);
              $title= substr($Lines[$j],$ad,$length_string-14-$ad);

               if ($Lines[$j+2]!~ m/Page(\D+)s(\D+): /) #search author
                 {
                  $length_author=length($Lines[$j+2]);
                  $aut=substr($Lines[$j+2],13,$length_author-18);
```

```
    $aut=~ s/(\; , .)//gi;
    @seperate_aut=split(/;/,$aut);
    $N_a=@seperate_aut;
    $author="";
    $#connect=-1;
     for($i_a=0;$i_a<$N_a;++$i_a)
     {
        @f=split(/,/,$seperate_aut[$i_a]);
        @f=(@f,@split_array);
        if($i_a>0)
          {
           $long_f=length($f[0]);
           $f[0]=substr($f[0],1,$long_f);
          }
        $connect[$i_a]=$f[1]." "."$f[0],";
        $author=$author . $connect[$i_a];
     }

    }

  if ($Lines[$j+2]=~ m/Page(\D+)s(\D+): /) #search page
    {
     $length_page=length($Lines[$j+2]);
     $page=substr($Lines[$j+2],23,$length_page-28);
    }

  elsif ($Lines[$j+4]=~ m/Page(\D+)s(\D+): /)
    {
     $length_page=length($Lines[$j+4]);
     $page=substr($Lines[$j+4],23,$length_page-28);
    }
 }

if ($Lines[$j]=~ m/>Abstract(\D+)a>/)  # search abstract link of each paper
 {
  $length_abst=length($Lines[$j]);
  $abstract=substr($Lines[$j],31,$length_abst-69);
  $link_abst=$b . $abstract;
  my @get_abstract= &get_content_ab($link_abst);
  my $abst_content= &get_abstractspan(@get_abstract);
  $each_abst[$jj]= $abst_content;
 }


if ($Lines[$j]=~ m/>PDF(\D+)a>/)    # search PDF link
```

```
{
@cut=split(/PDF/,$Lines[$j]);
@cut=(@cut,@split_array);
$length_pdf= length($cut[0]);
$pdf=substr($cut[0],46,$length_pdf-71);
$link_pdf=$c . $pdf;
$link_pdf=~ s/Xplore(\W)login.jsp(\W)url(\W+)//;
$link_pdf=~ s/pdf(\?)isnumber/pdf?tp=&isnumber/;
$link_pdf=~ s/prod=JNL&//;
$link_pdf=~ s/&arSt(.+)//;
$count=$jj+1;

$add_link[$jj]="<html><tr>
        <td
valign=\"top\">$startV.$startIS.$count   </td><td>$author \"$t
itle,\"<i> $shortcut,</i> vol. $startV, no. $startIS, pp. $page, $month
$startyear.  
        <a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
        <a href=$link_pdf><font color=\"blue\"> Full
Text</font></a><br><br>
        <font                                                    size=\"3\"
color=\"blue\"><b>Abstract:</b>$each_abst[$jj]</font><br><br>
        </td></tr>
        </html>";

$no_abstract[$jj]="<html><tr>
        <td
valign=\"top\">$startV.$startIS.$count   </td><td>$author \"$t
itle,\"<i> $shortcut,</i> vol. $startV, no. $startIS, pp. $page, $month
$startyear.  
        <a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
        <a href=$link_pdf><font color=\"blue\"> Full
Text</font></a><br><br>
        </td></tr>
        </html>";
 print ("$add_link[$jj]\n");
 $jj=$jj+1;
 $author="";
 }
}


$head="
```

```
<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
<html>
<head>
<title>$journal</title>
</head>
<body>
<h1 align=\"center\">
<img border=\"0\" src=\"ieeelogo.gif\">
<font color=\"#FF6600\"> $journal 
<img border=\"0\" src=\"ieslogo.gif\"></font>
</h1>
<h2 align=\"center\">
Volume $startV,  Number $startIS, $month $startyear          
<a href=\"http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=$pnum\"><font size=\"4\">Access to the journal on IEEE XPLORE</font></a><font size=\"4\"> </font>   
<a href=\"http://ieee-ies.org/o_tie.shtml\"><font size=\"3\">IE Transactions Home Page</font></a><font size=\"4\"> </font>
</h2>
<hr/><br><br>
</body>
</html>";


$volume="$startV"."_"."$startIS";
$file=$mydir."$volume.htm";# abstract
open(Kvolume,">$file");
print Kvolume "$head <table>";
print Kvolume "@add_link";
print Kvolume "</table>";
close(Kvolume);

$file=$mydir."$startV"."s.htm";  #append no abstract
open(Kstart,">>$file");
print Kstart "$head <table>";
print Kstart "@no_abstract";
print Kstart "</table>";
close(Kstart);

$file=$mydir."$startV.htm";  # append abstract
open(Kstartv,">>$file");
print Kstartv "$head <table>";
```

```
            print Kstartv "@add_link";
            print Kstartv "</table>";
            close(Kstartv);

            $delete="ISS_No.htm";
            unlink ($delete);

            $#get_issue =-1;
            $#clean_file =-1;
            $#filter =-1;
            $#filter1 =-1;
            $#get_issue =-1;
            $#add_link =-1;
            $#each_abst =-1;
            $#Lines =-1;
            my @get_abstract =();
            $#cut =-1;
            $#no_abstract =-1;

         }
        $beginIS=1;
       }
  }


            # all subroutines

sub get_abstractspan
{
   $N=@_;
   for ($i=0;$i<$N;++$i)
   {
    if ($_[$i]=~ m/>Abstract<(\D)span>/)

       {
        $abstrvar=$i;
        last;  #break
       }
   }
   $feedback=" ";
   for ($th=1;$th<100;++$th)
    {
      if ($_[$abstrvar+$th] !~ m/<(\D)td>/)
       {
        $feedback=$feedback . $_[$abstrvar+$th];
```

```
      }
     elsif ($_[$abstrvar+$th] =~ m/<(\D)td>/)
      {
       last;
      }
   }
  return ($feedback);
}


sub get_href
{
  $N=@_;
  @new=("");
  $j=0;
  for ($i=0;$i<$N;++$i)
   {
    if ($_[$i]=~ m/<a HREF=/)
      {
       $new[$j]=$_[$i];
       $j=$j+1;
      }
   }
  return(@new);
}


sub add_issue  #add content of all pages
{
   $file=$mydir."ISS_No.htm";
   open(K,">>$file");
   print K "@_";
   close(K);
   return 0;
}


sub get_content_ab
{
 $jjj=20;
 do{
   $add=$_[0];  # all incoming input
   $fname= "med.htm";
   system("wget.exe", "-q", "-O", $fname,$add);
   $file=$mydir."med.htm";
```

```
   open(med1,"<$file");
   @lines = <med1>;
   close(med1);
   unlink($fname); # delete temporary file
   $jjj=$jjj-1;
   $N=@lines;
   for ($i=0;$i<$N;++$i)
     {
      if ($lines[$i]=~ m/>Abstract<(\D)span>/)
        {
         last;
        }
     }
  }while (($lines[$i+1]!~ m/(\w)/)&&($jjj>1));

   return(@lines);
}


sub get_content
{
   $add=$_[0];  # all incoming input
   $fname= "med.htm";
   system("wget.exe", "-q", "-O", $fname,$add);
   $file=$mydir."med.htm";
   open(r1,"<$file");
   @lines = <r1>;
   close(r1);
   unlink($fname); # delete temporary file
   return(@lines);
}
```

APPENDIX B: citation_w.pl (perl program)

```
#!/usr/bin/perl -w
#use strict;

$textfile="citation_w.txt";
open (CHECK,$textfile) || die "couldn't open the file";
@check=<CHECK>;
close(CHECK);
$space=" ";
$CH=@check;

for($g=0;$g<$CH;++$g)  #format the size of each element of array
 {
  chop $check[$g];
  $leg=length($check[$g]);
   for($h=0;$h<50;++$h)
     {
      $index=rindex($check[$g],$space);
      if (($index==-1)||($index!=$leg-1))
        {
         last;
        }
      $check[$g]=substr($check[$g],0,$index);
      $leg=length($check[$g]);
     }
 }
@zh1=split(/=/,$check[0]);
@zh2=split(/=/,$check[1]);
@zh3=split(/=/,$check[2]);
@zh4=split(/=/,$check[3]);
@zh5=split(/=/,$check[4]);
%hash=($zh1[0] => $zh1[1],
      $zh2[0] => $zh2[1],
      $zh3[0] => $zh3[1],
      $zh4[0] => $zh4[1],
      $zh5[0] => $zh5[1],);

$articleYear=$hash{articleYear}; # year
$mydir=$hash{directory};#directory
$limit=$hash{long};   #code of a journal
$startYear=$hash{startYear}; #year for vol 1 of the journal
$journal=$hash{journal};
@split_array=("","","","",);
```

```
            #check input errors
$long_dir=length($mydir);
$mark="/";
$check_mark=rindex($mydir,$mark);
$long_article=length($articleYear);
if($check_mark!=$long_dir-1){print("you forgot the forward slash (/) of the
directory\n");}
if($long_article!=4){print("please check year input\n");}
elsif(($check_mark==$long_dir-1)&&($long_article==4))
  {
   $Volume=$articleYear-$startYear;
   my($sec,$min,$hour,$mday,$month,$year)=localtime time;
   $year=$year+1900;
   $month=$month+1;

   if($month==1) {$cdate="Jan";}
   if($month==2) {$cdate="Feb";}
   if($month==3) {$cdate="March";}
   if($month==4) {$cdate="April";}
   if($month==5) {$cdate="May";}
   if($month==6) {$cdate="June";}
   if($month==7) {$cdate="July";}
   if($month==8) {$cdate="August";}
   if($month==9) {$cdate="Sept";}
   if($month==10){$cdate="Oct";}
   if($month==11){$cdate="Nov";}
   elsif($month==12){$cdate="Dec";}

   $textfile="Citations.txt";
   open (TEST,$textfile) || die "couldn't open the file";
   @test=<TEST>;
   close(TEST);
   $blank=" ";
   $L=@test;

   for($v=0;$v<$L;++$v)  #format the size of each element of array
    {
      chop $test[$v];
      $leg=length($test[$v]);
       for($u=0;$u<50;++$u)
          {
            $index=rindex($test[$v],$blank);
            if (($index==-1)||($index!=$leg-1))
              {
```

```
          last;
          }
        $test[$v]=substr($test[$v],0,$index);
        $leg=length($test[$v]);
      }
}

for($j=0;$j<$L;++$j)
 {
  $test[$j] =~ s/(â€¦)/…/gi;
  $test[$j] =~ s/(â€”)/—/gi;
  $test[$j] =~ s/(â€“)/–/gi;
  $test[$j] =~ s/(â€œ)/“/gi;
  $test[$j] =~ s/(â€)/”/gi;
  $test[$j] =~ s/(”˜)/‘/gi;
  $test[$j] =~ s/(”™)/'/gi;
  $test[$j] =~ s/("")/“/gi;
  $test[$j] =~ s/(â€˜)/‘/gi;
  $test[$j] =~ s/Ã±a/ña/gi;
  $test[$j]=~ s/(\[)(J)(\])//gi;
 }

for($i=1;$i<$L;++$i)
 {
 @z=split(/","/,$test[$i]);
 @z=(@z,@split_array);
 $abstract[$i-1]=$z[3];
 @num=split(/jsp?/,$abstract[$i-1]);
 @num=(@num,@split_array);

 if($num[1]!~ m/arnumber/)
  {
   $code[$i-1]="&arnumber=nam1";
  }
 if($num[1]=~ m/arnumber/)
  {
   $code[$i-1]=$num[1];
   $code[$i-1]=~ s/(\?)/&/gi;
   $code[$i-1]=~ s/&amp;//gi;
  }

 @zz=split(/,"/,$z[0]);
 @zz=(@zz,@split_array);
 $citation[$i-1]=$zz[0];    #citations
 $author[$i-1]=$zz[1];    #author
```

```
      @z1=split(/",/,$z[1]);
      @z1=(@z1,@split_array);
      $title[$i-1]=$z1[0];   #title
     }
     @aut=@author;
     @e=@title;
     $Length=@e;
     for($i1=0;$i1<$Length;++$i1)
      {
       $e[$i1]=~ s/…//gi;
       $e[$i1]=~ s/(\[)//gi;
       $e[$i1]=~ s/(\(')//gi;
       $e[$i1]=~ s/(\\)//gi;
      }
   $file=$mydir."$Volume"."s.htm";  #open an old file
   open(compFile,"<$file");
   @normal= <compFile>;
   close(compFile);

   $Nom=@normal;
   for($i_nom=0;$i_nom<$Nom;++$i_nom)
    {
      if(($normal[$i_nom]=~  m/<td  valign/)&&($normal[$i_nom+1]=~  m/IEEE  Trans.
on Industrial Electronics/))
         {
          $normal[$i_nom]=$normal[$i_nom] . $normal[$i_nom+1];
          $normal[$i_nom] =~s/(\n)//gi;
          $normal[$i_nom+1]=$normal[$i_nom+2];
          $normal[$i_nom+2]=$normal[$i_nom+3];
          $normal[$i_nom+3]=$normal[$i_nom+4];
         }
       elsif(($normal[$i_nom]=~ m/<td valign/)&&($normal[$i_nom+2]=~ m/IEEE Trans.
on Industrial Electronics/))
         {
          $normal[$i_nom]=$normal[$i_nom] . $normal[$i_nom+1] . $normal[$i_nom+2];
          $normal[$i_nom] =~s/(\n)//gi;
          $normal[$i_nom+1]=$normal[$i_nom+3];
          $normal[$i_nom+2]=$normal[$i_nom+4];
          $normal[$i_nom+3]=$normal[$i_nom+5];
         }
    }

   @refFile=@normal;
   $L1=@code;
   $L2=@refFile;
```

```
for($j1=0;$j1<$L1;++$j1)
{
  $e[$j1] =~ tr/[A-Z]/[a-z]/;
  $aut[$j1] =~ tr/[A-Z]/[a-z]/;
  for($j2=0;$j2<$L2;++$j2)
   {
    $normal[$j2]=~ s/(\.)//gi;
    $normal[$j2] =~ tr/[A-Z]/[a-z]/;
    if(($normal[$j2]=~ m/$e[$j1]/)||($normal[$j2]=~ m/$aut[$j1]/))
       {
        $text[$j1]=$refFile[$j2+2];
        $text[$j1]=~ s/<br><br>/  /gi;
        $abs[$j1]=$refFile[$j2+1];
        @Vol=split(/,"/,$refFile[$j2]);
        @Vol=(@Vol,@split_array);
        $Is[$j1]=$Vol[1];
        @c=split(/ "/,$Vol[0]);
        @c=(@c,@split_array);
        $paper_title[$j1]=$c[1];
        @a=split(/ "/,$refFile[$j2]);
        @a=(@a,@split_array);
        @b=split(/<td>/,$a[0]);
        @b=(@b,@split_array);
        $extract_author[$j1]=$b[1];
        last;
       }

    if($refFile[$j2] =~ m/$code[$j1]/)
       {
        $text[$j1]=$refFile[$j2+1];
        $text[$j1]=~s/<br><br>/  /gi;
        $abs[$j1]=$refFile[$j2];
        @Vol=split(/,"/,$refFile[$j2-1]);
        @Vol=(@Vol,@split_array);
        $Is[$j1]=$Vol[1];
        @c=split(/ "/,$Vol[0]);
        @c=(@c,@split_array);
        $paper_title[$j1]=$c[1];
        @a=split(/ "/,$refFile[$j2-1]);
        @a=(@a,@split_array);
        @b=split(/<td>/,$a[0]);
        @b=(@b,@split_array);
        $extract_author[$j1]=$b[1];
        last;
       }
```

```
      elsif (($refFile[$j2] !~ m/$code[$j1]/)&&(($normal[$j2]!~ m/$e[$j1]/)||($normal[$j2]!~ m/$aut[$j1]/)))
         {
          $paper_title[$j1]=$title[$j1];
          $extract_author[$j1]=$author[$j1] . ",";
          $Is[$j1]="";
          $abs[$j1]="";
          $text[$j1]="";
         }
      }
   }

    $count=0;
    $n_cite=0;
    for ($ii=0;$ii<$L-1;++$ii)
     {
       $count=$count+1;
        $link[$ii]=" <html><tr>
                    <td valign=\"top\">$count.
   </td><td>$extract_author[$ii] \"$paper_title[$ii],\"$Is[$ii]
                    $abs[$ii]
                    $text[$ii]
                    </a>   <font color=\"red\">No. of Citations =
$citation[$ii]</font><br>
                    </td></tr>
                    </html>";
       print ("$link[$ii]\n");
       $n_cite=$n_cite+$citation[$ii];
     }

   $head="
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <html>
   <head>
   <title>$journal</title></head>
   <body>
   <h1 style=\"margin-top: 0; margin-bottom: 0\" align=\"center\">
   <img border=\"0\" src=\"ieslogo.gif\">
   <font size=\"5\" color=\"800000\"> 
MOST CITED TIE $articleYear PAPERS  
   <img border=\"0\" src=\"ieeelogo.gif\"></font></h1>
   <h1 style=\"margin-top: 0; margin-bottom: 0\" align=\"center\">
```

```
   <font size=\"2\" color=\"800000\">(SOURCE : GOOGLE SCHOLAR
AS&amp;nbsp;OF&amp;nbsp;$cdate $mday, $year) </font>
   <font size=\"3\">&amp;nbsp;</font></h1>
         <p style=\"margin-top: 0; margin-bottom: 0\" align=\"center\">
   <b><font size=\"3\" color=\"FF9933\">$count papers were published in
$articleYear</font></b><font color=\"FF9933\"><b> and
   they were cited $n_cite times </b></font></p><br>";

   $file=$mydir."$articleYear"."c.htm";
   open(k2,">$file");
   print k2 "$head  <table>";
   for($countdown=0;$countdown<$limit;++$countdown)
    {
     print k2 "$link[$countdown]";
    }
   print k2 "</table>";
   close(k2);
   print $limit;

  }
```

APPENDIX C: filter_w.pl (perl program)

```
# this program extract the old articles, new articles and common articles in two forthcomings
#!/usr/bin/perl -w
# use strict;

$textfile="filter_w.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
  chop $array[$v];
  $leg=length($array[$v]);
   for($u=0;$u<50;++$u)
      {
       $index=rindex($array[$v],$blank);
       if (($index==-1)||($index!=$leg-1))
         {
          last;
         }
       $array[$v]=substr($array[$v],0,$index);
       $leg=length($array[$v]);
      }
 }
@z1=split(/=/,$array[0]);
@z2=split(/=/,$array[1]);
@z3=split(/=/,$array[2]);
%hash=($z1[0] => $z1[1],
      $z2[0] => $z2[1],
      $z3[0] => $z3[1],);

$previous=$hash{PreviousFile};
$short_previous=$previous;
if($short_previous=~ /data/){ $short_previous=~ s/_data//;}
$current=$hash{CurrentFile};
$short_current=$current;
if($short_current=~ /data/){ $short_current=~ s/_data//;}
$mydir=$hash{directory};#directory
```

```
        #check input errors
$long_dir=length($mydir);
$mark="/";
$check_mark=rindex($mydir,$mark);

for($i_pre=0;$i_pre<1;++$i_pre)
{
 if($check_mark!=$long_dir-1)
  {
   print("you forgot the back slash (/) of the directory\n");
   last;
  }

 $file=$mydir."$previous.htm";  #open an old file of forthcoming
 open(k_previous,"<$file");
 @a= <k_previous>;
 close(k_previous);

 $file=$mydir."$current.htm";  #open a new file of forthcoming
 open(k2,"<$file");
 @b= <k2>;
 close(k2);

 $Na=@a;
 $i=0;
 for ($ia=0;$ia<$Na;++$ia)
 {
  if ($a[$ia]=~ m/IEEE Trans. on Industrial Electronics/)
    {
     $a[$ia]=~ s/ (\D+)td><td>/cutit/;
     $a[$ia]=~ s/   /cutit/;
     $a[$ia] =~ s/(\()/_/gi;
     $a[$ia] =~ s/(\))/_/gi;
     $a[$ia] =~ s/(\+)/_/gi;
     $a[$ia] =~ s/(\*)/_/gi;
     $a[$ia] =~ s/(\?)/_/gi;
     $a[$ia] =~ s/_/ /gi;
     @array1=split(/cutit/,$a[$ia]);
     $c[$i]=$array1[1];
     $i=$i+1;
     $c[$i]=$a[$ia+1];
     $i=$i+1;
     $c[$i]=$a[$ia+2];
     $i=$i+1;
```

```
   }
}
@e=@c;
$Nb=@b;
$j=0;
for ($ib=0;$ib<$Nb;++$ib)
{
  if ($b[$ib]=~ m/IEEE Trans. on Industrial Electronics/)
    {
      $b[$ib]=~ s/ (\D+)td><td>/cutit/;
      $b[$ib]=~ s/   /cutit/;
      $b[$ib] =~ s/(\()/_/gi;
      $b[$ib] =~ s/(\))/_/gi;
      $b[$ib] =~ s/(\+)/_/gi;
      $b[$ib] =~ s/(\*)/_/gi;
      $b[$ib] =~ s/(\?)/_/gi;
      $b[$ib] =~ s/_ /gi;
      @array2=split(/cutit/,$b[$ib]);
      $d[$j]=$array2[1];
      $j=$j+1;
      $d[$j]=$b[$ib+1];
      $j=$j+1;
      $d[$j]=$b[$ib+2];
      $j=$j+1;
    }
}
@f=@d;

          # extract old articles from two forthcomings
$Nc=@c;
$Nd=@d;
for ($jd=0;$jd<$Nd;++$jd)  #extract authors and titles of the new files
{
 if ($d[$jd]=~ m/IEEE Trans. on Industrial Electronics/)
  {
    $journal="<i> IEEE Trans. on Industrial Electronics";
    @dd=split(/$journal/,$d[$jd]);

    for($iic=0;$iic<$Nc;++$iic)       # compare and save the old articles
       {
         if($c[$iic]=~ m/$dd[0]/)
         {
           $c[$iic]="";
           $c[$iic+1]="";
           $c[$iic+2]="";
```

```
            last;
          }
        }
    }
  }

$Nc=@c;
$countc=0;
for($iic=0;$iic<$Nc;++$iic)
{
  if($c[$iic]=~ m/IEEE Trans. on Industrial Electronics/)
    {
     $countc=$countc+1;
     $c[$iic]=$countc.". ".$c[$iic];
    }
}

$head1="
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <html>
   <head>
   <title>IEEE Transactions on Industrial Electronics</title>
   </head>
   <body>
   <h2 align=\"center\">
   $countc Old article(s) between $short_previous and $short_current..
   </h2>
   <hr/><br><br>
   </body>
   </html>";

$file=$mydir."Old.htm";
open(k1,">$file");
print k1 "$head1";
print k1 "@c";
close(k1);
             #extract new articles from two forthcomings

$Nd=@d;
$Ne=@e;
  for ($je=0;$je<$Ne;++$je)  #extract authors and titles of the old files
  {
   if ($e[$je]=~ m/IEEE Trans. on Industrial Electronics/)
    {
```

```
    $journal="<i> IEEE Trans. on Industrial Electronics";
    @ee=split(/$journal/,$e[$je]);

    for($iid=0;$iid<$Nd;++$iid)       # compare and save the new articles
      {
       if($d[$iid]=~ m/$ee[0]/)
       {
         $d[$iid]="";
         $d[$iid+1]="";
         $d[$iid+2]="";
         last;
       }
      }
   }
  }

@g=@d;
$Nd=@d;
$countd=0;
for($iid=0;$iid<$Nd;++$iid)
{
  if($d[$iid]=~ m/IEEE Trans. on Industrial Electronics/)
    {
     $countd=$countd+1;
     $d[$iid]=$countd.". ".$d[$iid];
    }
}

$head2="
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <html>
   <head>
   <title>IEEE Transactions on Industrial Electronics</title>
   </head>
   <body>
   <h2 align=\"center\">
   $countd New article(s) between $short_previous and $short_current..
   </h2>
   <hr/><br><br>
   </body>
   </html>";

$file=$mydir."New.htm";
open(k2,">$file");
```

```
print k2 "$head2";
print k2 "@d";
close(k2);

                #extract common articles

$Nf=@f;
$Ng=@g;
  for ($jg=0;$jg<$Ng;++$jg)  #extract authors and titles of the old files
  {
   if ($g[$jg]=~ m/IEEE Trans. on Industrial Electronics/)
    {
     $journal="<i> IEEE Trans. on Industrial Electronics";
     @gg=split(/$journal/,$g[$jg]);
     for($iif=0;$iif<$Nf;++$iif)       # compare and save the common articles
       {
         if($f[$iif]=~ m/$gg[0]/)
          {
           $f[$iif]="";
           $f[$iif+1]="";
           $f[$iif+2]="";
           last;
          }
       }
    }
  }

$Nf=@f;
$countf=0;
for($iif=0;$iif<$Nf;++$iif)
{
  if($f[$iif]=~ m/IEEE Trans. on Industrial Electronics/)
    {
     $countf=$countf+1;
     $f[$iif]=$countf.". ".$f[$iif];
    }
}

$head3="
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <html>
   <head>
   <title>IEEE Transactions on Industrial Electronics</title>
   </head>
```

```
   <body>
   <h2 align=\"center\">
   $countf Common article(s) between $short_previous and $short_current.
   </h2>
   <hr/><br><br>
   </body>
   </html>";

$file=$mydir."Common.htm";
open(k3,">$file");
print k3 "$head3";
print k3 "@f";
close(k3);


  print(" There are $countc old articles.\n");
  print(" There are $countd new articles.\n");
  print(" There are $countf common articles.\n");

}
```

APPENDIX D: countdown_w.pl (Perl program)

```
#!/usr/bin/perl -w
#use strict;

$textfile="countdown_w.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
   chop $array[$v];
   $leg=length($array[$v]);
    for($u=0;$u<50;++$u)
       {
         $index=rindex($array[$v],$blank);
         if (($index==-1)||($index!=$leg-1))
           {
            last;
           }
         $array[$v]=substr($array[$v],0,$index);
         $leg=length($array[$v]);
       }
 }
@z1=split(/=/,$array[0]);
@z2=split(/=/,$array[1]);
@z3=split(/=/,$array[2]);
@z4=split(/=/,$array[3]);
@z5=split(/=/,$array[4]);
@z6=split(/=/,$array[5]);
@z7=split(/=/,$array[6]);
@z8=split(/=/,$array[7]);
@z9=split(/=/,$array[8]);
@z10=split(/=/,$array[9]);
@z11=split(/=/,$array[10]);
%hash=($z1[0] => $z1[1],
       $z2[0] => $z2[1],
       $z3[0] => $z3[1],
       $z4[0] => $z4[1],
       $z5[0] => $z5[1],
```

```
      $z6[0] => $z6[1],
      $z7[0] => $z7[1],
      $z8[0] => $z8[1],
      $z9[0] => $z9[1],
      $z10[0] => $z10[1],
      $z11[0] => $z11[1],);

$Tyear=$hash{TOyear}; # year
$Fyear=$hash{FROMyear};
$mydir=$hash{directory};#directory
$startYear=$hash{startYear};
$forthcoming=$hash{forthcoming};
$countdown=$hash{COUNTDOWN};
$subtitle=$hash{SUBTITLE};
$subTyear=$hash{subTOyear}; # year
$subFyear=$hash{subFROMyear};
$IS=$hash{TOissue}; #date
$beginIS=$hash{FROMissue};

                #check errors
$long_dir=length($mydir);
$mark="/";
$check_mark=rindex($mydir,$mark);
for($dir=0;$dir<1;++$dir)
{
   if($check_mark!=$long_dir-1)
     {
      print("you forgot the forward slash (/) of the directory\n");
      last;
     }
   if($subtitle =~ m/yes/)
     {
      for($i_pnum=0;$i_pnum<1;++$i_pnum)
        {
         if($IS<$beginIS)
         {
           print ("please check the issue numbers you want to extract\n");
           last;
         }
         if($subTyear<$subFyear)
         {
           print("please check number of years you want to extract\n");
           last;
         }
```

```
my $V= $subTyear-$startYear;   #volume

for($startV=$subFyear-$startYear;$startV<=$V;++$startV)
 {
   $#sub_combine=-1;
   for($i=$beginIS;$i<=$IS;++$i)
    {
     $filename=$startV."_".$i;
     $file=$mydir."$filename.htm";  #open an old file
     open(rFile,"<$file");
     @oldFile= <rFile>;
     @sub_combine=(@sub_combine,@oldFile);
     close(rFile);
    }
   $file=$mydir."$startV.htm";  #combine all old files with abstract
   open(add_abtract,">$file");
   print add_abtract "@sub_combine";
   close(add_abtract);

   $length_com=@sub_combine;
   my @delete_abs=@sub_combine;
   for($j=0;$j<$length_com;++$j)
   {
     if($delete_abs[$j]=~ m/<b>Abstract:(\W+)b>/)
     {
      $num=$j+1;
      $delete_abs[$j]="";
      for($jj=$num;$jj<$num+200;++$jj)
       {
         if ($delete_abs[$jj]!~ m/(\W+)font><br><br>/)
           {
            $delete_abs[$jj]="";
           }
         if ($delete_abs[$jj]=~ m/(\W+)font><br><br>/)
           {
            $delete_abs[$jj]="";
            last;
           }
       }
     }
   }

   $file=$mydir."$startV"."s.htm";  #combine all old files with abstract
   open(del_abstract,">$file");
   print del_abstract "@delete_abs";
```

```
          close(del_abstract);
          print("you have done\n");
        }
      }
  }

       #countdown program

$begin=$Fyear-$startYear;
$endyear=$Tyear-$startYear;

if(($Tyear>$Fyear)&&($countdown =~ m/yes/))
  {
    if($forthcoming =~m/yes/)
      {
       $file=$mydir."forthcoming.htm";  #open forthcoming files
       open(forth_no,"<$file");
       @forth_no= <forth_no>;
       close(forth_no);

       $file=$mydir."forthcoming_a.htm";
       open(forth_a,"<$file");
       @forth_a= <forth_a>;
       close(forth_a);
      }
    $#combine=-1;
    for($startyear=$begin;$startyear<=$endyear;++$startyear)
      {
       $file=$mydir."$startyear"."s.htm";  #open all old files
       open(rFile,"<$file");
       @refFile= <rFile>;
       @combine=(@combine,@refFile);
       close(rFile);
      }

    $file=$mydir."$Fyear"."_"."$Tyear"."s.htm";  #combine all old files
    open(combination1,">$file");
    print combination1 "@combine";
    if($forthcoming =~ m/yes/){print combination1 "@forth_no";}
    close(combination1);

    $#addin=-1;
    for($startyear=$begin;$startyear<=$endyear;++$startyear)
      {
       $file=$mydir."$startyear".".htm";  #open all old files with abstract
```

```
          open(addFile,"<$file");
          @comFile= <addFile>;
          @addin=(@addin,@comFile);
          close(addFile);
         }


      $file=$mydir."$Fyear"."_"."$Tyear".".htm";  #combine all old files with abstract
      open(add_abstract,">$file");
      print add_abstract "@addin";
      if($forthcoming =~ m/yes/){print add_abstract "@forth_a";}
      close(add_abstract);
      print("you have done\n");
   }
   elsif($Tyear<=$Fyear)
     {print("please check input years of countdown program again\n");}
}
```

APPENDIX E: extract_citation.pl (perl program)

```
#!/usr/bin/perl -w
#use strict;

$textfile="extract_citation.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
  chop $array[$v];
  $leg=length($array[$v]);
   for($u=0;$u<50;++$u)
     {
      $index=rindex($array[$v],$blank);
      if (($index==-1)||($index!=$leg-1))
        {
         last;
        }
      $array[$v]=substr($array[$v],0,$index);
      $leg=length($array[$v]);
     }
 }

@z1=split(/=/,$array[0]);  #check "=" sign in the string
$Nz1=@z1;                  # and recombine it
if ($Nz1>2)
{
  for ($j1=1;$j1<$Nz1-1;++$j1)
   {
     $z1[1]=$z1[1]."=".$z1[$j1+1];
   }
}

@z2=split(/=/,$array[1]);
$Nz2=@z2;
if ($Nz2>2)
{
  for ($j2=1;$j2<$Nz2-1;++$j2)
   {
```

```
    $z2[1]=$z2[1]."=".$z2[$j2+1];
   }
}

@z3=split(/=/,$array[2]);
$Nz3=@z3;
if ($Nz3>2)
{
  for ($j3=1;$j3<$Nz3-1;++$j3)
   {
     $z3[1]=$z3[1]."=".$z3[$j3+1];
   }
}

%hash=($z1[0] => $z1[1],
       $z2[0] => $z2[1],
       $z3[0] => $z3[1],);

$year=$hash{year}; # year
$mydir=$hash{directory};#directory
$webpage=$hash{webpage};# weblink
print("$webpage\n");
@split_array=("","","","",);
$addr=$webpage;
my @get_issue=&get_content($addr); #get content of page of issue
my $temp=&add_issue(@get_issue);  #add content of all pages in the same file
$N_get_issue=@get_issue;

for($i=0;$i<$N_get_issue;++$i)
 {
  if($get_issue[$i]=~ m/pageCount.top">/)
     {
      @cutoff_1=split(/pageCount.top">/,$get_issue[$i]);
      @cutoff_2=split(/<(\D+)span>/,$cutoff_1[1]);
      $num_of_link=$cutoff_2[0];
      print("number of links:$num_of_link\n");
     }
  last;
 }
chop $addr;
for($j=2;$j<=$num_of_link;++$j)    #copy content of all links
 {
  $link="$addr"."$j";
  my @get_issue=&get_content($link);
  my $temp1=&add_issue(@get_issue);
```

```
}

 $file=$mydir."ISS_No.htm";
 open(K1,"<$file");
 @Lines=<K1>;
 close(K1);
 $N1=@Lines;
 $in=0;
 $#cite_array=-1;

 for ($i1=0;$i1<$N1;++$i1)
{
 if($Lines[$i1]=~ m/<br>Author(\D)s(\D+)/) #extract author
    {
     @aut=split(/<br>Source/,$Lines[$i1]);
     $length_author=length($aut[0]);
     $auth=substr($aut[0],16,$length_author-16);
     $auth=~ s/et al.//gi;
     @seperate_aut=split(/, /,$auth);
     $N_a=@seperate_aut;
     $author="";
     $#connect=-1;
      for($i_a=0;$i_a<$N_a;++$i_a)
      {
         @f1=split(/ /,$seperate_aut[$i_a]);
         @f1=(@f1,@split_array);
         @f2=split(//,$f1[1]);
         $lengthf2=@f2;
         if ($lengthf2=="1"){$f=$f2[0].".";}
         if ($lengthf2=="2"){$f=$f2[0].".".$f2[1].".";}
         if ($lengthf2=="3"){$f=$f2[0].".".$f2[1].".".$f2[2].".";}
         $connect[$i_a]=$f." "."$f1[0], ";
         $author=$author . $connect[$i_a];
      }
     chop $author;

     if ($Lines[$i1]=~ m/Pages: <span/)   #extract page number
        {
         @p=split(/Pages: <span/,$Lines[$i1]);
         @pp=split(/<(\D)span>/,$p[1]);
         $length_page=length($pp[0]);
         $page=substr($pp[0],19,$length_page-19);
        }
     if ($Lines[$i1]=~ m/Volume: <span class/) #extract volume number
        {
```

```
        @vol=split(/Volume: <span class/,$Lines[$i1]);
        @vol1=split(/<(\D)span>/,$vol[1]);
        $Vlength=length($vol1[0]);
        $volume=substr($vol1[0],13,$Vlength-13);
       }

     if ($Lines[$i1+1]=~ m/<br>Times Cited:/) #extract number of citation
      {
       if($Lines[$i1+1]=~ m/&amp;db_id=WOS">/)
       {
        @cite=split(/&amp;db_id=WOS">/,$Lines[$i1+1]);
        $cite[1]=~ s/<(\D)a><(\D)span>//gi;
        chop $cite[1];
        chop $cite[1];
        $citation=$cite[1];
       }
       if($Lines[$i1+1]!~ m/&amp;db_id=WOS">/)
       {
        @cite=split(/data_bold">/,$Lines[$i1+1]);
        $cite[1]=~ s/<(\D)span>//gi;
        chop $cite[1];
        chop $cite[1];
        $citation=$cite[1];
       }

      }

     if ($Lines[$i1-1]=~ m/Title: <a class/)  #extract Title
       {
        @t=split(/&amp;doc=/,$Lines[$i1-1]);
        $t[1]=~ s/<(\D)a>//;
        $t[1]=~ s/(\d+)">//;
        $title=$t[1];
        chop $title;
       }
    if ($volume==$year-1953)
    {
    $combine[$in]=$citation.",\""."$author\"".",\""."$title\"".",\""."year. $year\"".",\""."vol.
$volume\"".",\""."pp. $page\"".",\""."IEEE Transactions on Industrial Electronics\""."\n";
    $cite_array[$in]=$citation;
    $hash1{$combine[$in]}= $cite_array[$in];  #add element into hash
    $in=$in+1;
    }
     }
 }
```

```
$#information=-1;
$first=" Cites,Authors,Title,Year,Volume,Page,Publisher\n";
$ii=0;
foreach (sort { $hash1{$b} <=> $hash1{$a} } keys(%hash1) ) #arrange in oder.
{
   $information[$ii]=$_;
   $ii=$ii+1;
}

$file=$mydir."Citations_isi_"."$year.txt";
open(K,">$file");
print K "$first";
print K "@information";
close(K);

$delete="ISS_No.htm";
unlink ($delete);


              # Subroutine
sub add_issue  #add content of all pages
{
   $file=$mydir."ISS_No.htm";
   open(K,">>$file");
   print K "@_";
   close(K);
   return 0;
}

sub get_content
{
   $add=$_[0];  # all incoming input
   $fname= "med.htm";
   system("wget.exe", "-q", "-O", $fname,$add);
   $file=$mydir."med.htm";
   open(r1,"<$file");
   @lines = <r1>;
   close(r1);
   unlink($fname); # delete temporary file
   return(@lines);
}
```

APPENDIX F: forthcoming_w.pl (perl program)

```
#!/usr/bin/perl -w
#use strict;
$textfile="forthcoming_w.txt";
open (TEST,$textfile) || die "couldn't open the file";
@array=<TEST>;
close(TEST);
$blank=" ";
$L=@array;

for($v=0;$v<$L;++$v)  #format the size of each element of array
 {
  chop $array[$v];
  $leg=length($array[$v]);
   for($u=0;$u<50;++$u)
     {
      $index=rindex($array[$v],$blank);
      if (($index==-1)||($index!=$leg-1))
        {
         last;
        }
      $array[$v]=substr($array[$v],0,$index);
      $leg=length($array[$v]);
     }
 }
@z1=split(/=/,$array[0]);
@z2=split(/=/,$array[1]);
@z3=split(/=/,$array[2]);
@z4=split(/=/,$array[3]);
%hash=($z1[0] => $z1[1],
      $z2[0] => $z2[1],
      $z3[0] => $z3[1],
      $z4[0] => $z4[1],);

$mydir=$hash{directory};#directory
$pnum=$hash{code};   #code of a journal
$journal=$hash{journal};
$shortcut=$hash{shortcut};

            #check input errors
$long_dir=length($mydir);
$mark="/";
```

```
$check_mark=rindex($mydir,$mark);
if($check_mark!=$long_dir-1){print("you forgot the forward slash (/) of the
directory\n");}
for($i_pnum=0;$i_pnum<1;++$i_pnum)
  {
    if($pnum =~ m/(\D+)/ )
    {
      print("you need to put a code in numbers not in words\n");
      last;
    }

    my($sec,$min,$hour,$mday,$month,$year)=localtime time;
    $year=$year+1900;
    $month=$month+1;

    if($month==1) {$cdate="Jan";}
    if($month==2) {$cdate="Feb";}
    if($month==3) {$cdate="March";}
    if($month==4) {$cdate="April";}
    if($month==5) {$cdate="May";}
    if($month==6) {$cdate="June";}
    if($month==7) {$cdate="July";}
    if($month==8) {$cdate="August";}
    if($month==9) {$cdate="Sept";}
    if($month==10){$cdate="Oct";}
    if($month==11){$cdate="Nov";}
    elsif($month==12){$cdate="Dec";}

    my $V=$year - 1953;  #volume
    $page_issue=1;
    $a="http://ieeexplore.ieee.org/xpl";  #patial link of other articles in an issue
    $b="http://ieeexplore.ieee.org/xpls"; #patial link of abstract
    $c="http://ieeexplore.ieee.org/Xplore/login.jsp?url=";#patial link of PDF

$addr="http://ieeexplore.ieee.org/xpl/tocpreprint.jsp?isnumber=4387790&Submit3=Vie
w+Articles&punumber=$pnum";#full link of an issue
    @split_array=("","","","",);

    my @get_issue=&get_content($addr); #get content of page of issue
    my $temp=&add_issue(@get_issue);  #add content of all pages in the same file
    my @clean_file=&get_href(@get_issue); #clean to have file start with <a HREF

    for($i=0;$i<$page_issue-1;++$i)
      {
        @filter=split(/ class/,$clean_file[$i]);
```

```
    @filter1=split(/xpl/,$filter[0]);
    $link=$a . $filter1[1];
    my @get_issue=&get_content($link);
    my $temp1=&add_issue(@get_issue);
   }

$file=$mydir."ISS_No.htm";
open(K1,"<$file");
@Lines=<K1>;
close(K1);
$N1=@Lines;
@add_link="";
@each_abst="";
$jj=0;
$str_title="strong";
for ($j=0;$j<$N1;++$j)
  {
  if ($Lines[$j]=~ m/strong><br/)   # search the title
   {
   $start= index($Lines[$j],$str_title);
   $ad=$start+7;
   $length_string= length($Lines[$j]);
   $title= substr($Lines[$j],$ad,$length_string-14-$ad);
   $length_author=length($Lines[$j+1]);# author
   $aut=substr($Lines[$j+1],20,$length_author-29);
   $aut=~ s/(\; , .)//gi;
   @seperate_aut=split(/;/,$aut);
   $N_a=@seperate_aut;
   $author="";
   $#connect=-1;
    for($i_a=0;$i_a<$N_a;++$i_a)
    {
       @f=split(/,/,$seperate_aut[$i_a]);
       if($i_a>0)
         {
          $long_f=length($f[0]);
          $f[0]=substr($f[0],1,$long_f);
         }
       $connect[$i_a]=$f[1]." "."$f[0],";
       $author=$author . $connect[$i_a];
    }
   $length_page=length($Lines[$j+2]);#page
   $page=substr($Lines[$j+2],25,$length_page-31);
    }
   if ($Lines[$j]=~ m/>Abstract(\D+)a>/)  # search abstract link of each paper
```

```
        {
          $length_abst=length($Lines[$j]);
          $abstract=substr($Lines[$j],35,$length_abst-68);
          $link_abst=$b . $abstract;
          my @get_abstract= &get_content_ab($link_abst);
          my $abst_content= &get_abstractspan(@get_abstract);
          $each_abst[$jj]= $abst_content;
        }
       if (($Lines[$j]=~ m/>PDF(\D+)a>/)&&($Lines[$j]!~ m/<!--/))     # search PDF link
       {
        @cut=split(/PDF/,$Lines[$j]);
        $length_pdf= length($cut[0]);
        $pdf=substr($cut[0],46,$length_pdf-48);
        $f_link=$c . $pdf;
        $f_link=~ s/tp=//;
        @d_link=split(/&/,$f_link);
        @d_link=(@d_link,@split_array);
        @d_num= split(/=/,$d_link[2]);
        @d_num=(@d_num,@split_array);
        $link_pdf=$d_link[0].$d_link[2]."&".$d_link[1];
        $url="url=/iel5/41/".$d_num[1];
        $link_pdf=~ s/url=/$url/;
        $link_pdf=~ s/(\W)Xplore(\W)login.jsp(\W)url=//;
        $count=$jj+1;
        $add_link[$jj]="<html><tr>
                        <td
valign=\"top\">$count   </td><td>$author \"$title,\"<i>
$shortcut,</i> vol. $V, no. ?, pp. $page, . $year.  
                        <a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
                        <a href=$link_pdf><font color=\"blue\"> Full Text</font></a><br><br>
                        <font size=\"3\"
color=\"blue\"><b>Abstract:</b>$each_abst[$jj]</font><br><br>
                        </td></tr>
                        </html>";

         $no_abstract[$jj]="<html><tr>
                        <td
valign=\"top\">$count   </td><td>$author \"$title,\"<i>
$shortcut,</i> vol. $V, no. ?, pp. $page, . $year.  
                        <a href=$link_abst><font color=\"blue\"> Abstract Link</font>
</a>  
                        <a href=$link_pdf><font color=\"blue\"> Full Text</font></a><br><br>
                        </td></tr>
                        </html>";
```

```
      print ("$add_link[$jj]\n");
      $jj=$jj+1;
      $author="";
     }
    }
   $head="
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01//EN\">
   <html>
   <head>
   <title>Forthcoming Articles - Accepted publication</title>
   </head>
   <body>
   <h1 align=\"center\" >
   <img border=\"0\" src=\"ieeelogo.gif\">
   <font color=\"#FF6600\"> $journal 
   <img border=\"0\" src=\"ieslogo.gif\"></font>
   </h1>
   <h2 align=\"center\">
   $jj forthcoming articles as of $cdate $mday   $year
         
   <a href=\"http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=$pnum\"><font
size=\"4\">Access to the journal on IEEE XPLORE</font></a><font size=\"4\">
</font>   
   <a href=\"http://ieee-ies.org/o_tie.shtml\"><font size=\"3\">IE Transactions Home
Page</font></a><font size=\"4\"> </font>
   </h2>
   <hr/><br><br>
   </body>
   </html>";

   $file=$mydir."forthcoming_a.htm";
   open(K,">$file");
   print K "$head <table>";
   print K "@add_link";
   print K "</table>";
   close(K);

   $file=$mydir."forthcoming.htm";
   open(K,">$file");
   print K "$head <table>";
   print K "@no_abstract";
   print K "</table>";
   close(K);
```

```
    $delete="ISS_No.htm";
    unlink ($delete);   #delete a file
  }

              # all subroutines
sub get_abstractspan
{
   $N=@_;
   for ($i=0;$i<$N;++$i)
   {
    if ($_[$i]=~ m/>Abstract<(\D)span>/)
       {
        last;
       }
   }
   return ($_[$i+3]);
}

sub get_href
{
   $N=@_;
   @new=("");
   $j=0;
   for ($i=0;$i<$N;++$i)
    {
     if ($_[$i]=~ m/<a HREF=/)
       {
        $new[$j]=$_[$i];
        $j=$j+1;
       }
    }
   return(@new);
}

sub add_issue  #add content of all pages
{
    $file=$mydir."ISS_No.htm";
    open(K,">>$file");
    print K "@_";
    close(K);
    return 0;
}
sub get_content_ab
{
  $jjj=20;
```

```
  do{
    $add=$_[0];  # all incoming input
    $fname= "med.htm";
    system("wget.exe", "-q", "-O", $fname,$add);
    $file=$mydir."med.htm";
    open(r_content,"<$file");
    @lines = <r_content>;
    close(r_content);
    unlink($fname); # delete temporary file
    $jjj=$jjj-1;
    $N=@lines;
    for ($i=0;$i<$N;++$i)
      {
       if ($lines[$i]=~ m/>Abstract<(\D)span>/)
         {
          last;
         }
      }
  }while (($lines[$i+3]!~ m/(\w)/)&&($jjj>1));

    return(@lines);
}
sub get_content
{
    $add=$_[0];  # all incoming input
    $fname= "med.htm";
    system("wget.exe", "-q", "-O", $fname,$add);
    $file=$mydir."med.htm";
    open(r_getcontent,"<$file");
    @lines = <r_getcontent>;
    close(r_getcontent);
    unlink($fname); # delete temporary file
    return(@lines);
}
```

Users Manual

All our Robots are directed by the control text files. These control text files work as the interface with users.

The Loop_issue program:

```
code=41
MAXissue=12
FROMissue=5
FROMyear=2009
TOissue=5
TOyear=2009
startYear=1953
directory=C:/ALL BACK UP/Industrial Electronics on Trans/
journal=IEEE Transactions on Industrial Electronics
shortcut=IEEE Trans. on Industrial Electronics
```

The loop_issue program can be used to download one issue or many issues in one year or in many years. If users want to download one issue in one year, they have to modify the same issue number FROMissue = TOissue and the same year number FROMyear = TOyear in text file. If users want to download all issues in one year for examples all issues from 1 to 6 in year 2008, they have to modify FROMissue=1 and TOissue=6, FROMyear=2008 and TOyear=2008. If users want to download many issues for many years, they have to modify the different year numbers for FROMyear and TOyear. While code=41 is the code number of Transactions on Industrial Electronics. However, this code number can be changed depending on what Transactions on IEEE Xplore that users want to extract data. Journal and shortcut are just the format which the users want to define for beauty and convenience. And directory is the location of programs and html files will be generated.

This program will generate all issues files (for example 55_1.htm, 55_2.htm …in 2008 including abstracts) and whole year issue files with the abstracts and without the abstracts (for example 55.htm, 55s.htmfor 2008).

The forthcoming program:

```
code=41
directory=C:/ALL BACK UP/Industrial Electronics on Trans/
journal=IEEE Transactions on Industrial Electronics
shortcut=IEEE Trans. on Industrial Electronics
```

This program generates two forthcoming article files with the abstracts and without the abstracts: forthcoming.htm and forthcoming_a.htm. It works very much the same like the Loop_issue program but the control text file is simpler. The users only have to define the code number and this Robot will automatically define the date of the forthcoming papers.

The countdown program:

The countdown program has two sub-programs **subtitle** and **countdown**. By inputting yes or no, users can decide to use one sub-program or two sub-programs at the same time.

```
directory=C:/ALL BACK UP/Industrial Electronics on Trans/
SUBTITLE=yes
FROMissue=1
TOissue=2
subFROMyear=2009
subTOyear=2009
COUNTDOWN=no
forthcoming=no
startYear=1953
FROMyear=1988
TOyear=2008
```

1. **The Subtitle program:**

Purpose of this program is to modify whole year issue files again (for example 55.htm and 55s.htm in 2008) after users add the subtitles in each issue file. Instead of using Microsoft front-page to combine them manually, this program can substitute and

work faster. And users have to decide the number of issue files which need to be combined, kind of similar to the way users define in the loop_issue program.

This program will combine all issue files in one year (for example 55_1.htm, 55_2.htm… in 2008) to generate two new files with the abstracts and without the abstracts (55.htm and 55s.htm).

2. **The countdown program:**

Users can use this program to combine all articles from this year to another year with abstracts and without abstracts by defining the input variables in the text file: FROMyear=2000, TOyear=2001.
This program will generate two files for example 2000_2001.htm and 2000_2001s.htm. If users modify forthcoming=yes/no, the forthcoming papers may be included in this file or not.

The citation program:

```
directory=C:/ALL BACK UP/Industrial Electronics on Trans/
long=100
startYear=1953
articleYear=1998
journal=IEEE Trans. Industrial Electronics
```

This program will let you know how many times an article is cited in only one year. In order to know that, users have to do two steps.
Step1: Open the Publish&Perish software and save the text file with the Citations name into the directory of program.
Step2: By defining the same number of the article year in the citation_w text file as in Publish&Perish , now you can run this program.
The program will generate a file name 2007c.htm for example. While “long=100” means that you only extract top 100 papers that are most cited.

The filter program:

```
directory=C:/ALL BACK UP/Industrial Electronics on Trans/
PreviousFile=08_09_21_data
CurrentFile=09_02_09_data
```

This program compares the forthcoming articles that users downloaded before and now to extract and create three files: New.htm, Old.htm and common.htm.

The New.htm file contains all information of the new forthcoming articles only in the forthcoming file users have just downloaded.

The Old.htm file contains all information of the old forthcoming articles only in the forthcoming file users downloaded before.

The common.htm file contains all information of the common articles between the previous forthcoming file and the actual forthcoming file.

The extract citation program:

```
webpage=http://apps.isiknowledge.com/summary.do?product=WOS&qid=1&
SID=3Fl7g58@GKN14FHEO6B&search_mode=GeneralSearch&formValue(s
ummary_mode)=GeneralSearch&page=1
directory=C:/Documents and Settings/Dinh Nam/My Documents/back up Perl
program/
year=2007
```

This program extracts the citation information of the papers from ISI web of knowledge and arranges them in the descending order of number of citations. Basically, it substitutes for “Publish and Perish” software.