International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Implementation of a Web Crawler in the Inaprocplus Application Using the Breadth-First Search Algorithm

Abdul Wahid[1], Astie Darmayantie[2]
[1,2]Faculty of Computer Science and Technology, Gunadarma University, Depok, West Java, Indonesia-16424

Abstract — *Government procurement information is available through many LPSE portals managed by ministries, institutions, and local governments. Although the information is publicly accessible, its distribution across separate portals makes tender searching less efficient because users must check many websites manually. Tender data are also dynamic because schedules, evaluation stages, and winner announcements may change during the procurement process. This study discusses the implementation of a web crawler in the Inaprocplus application using the Breadth-First Search (BFS) algorithm. BFS was selected because it explores web pages level by level, making the crawling process more structured and easier to control. The system was developed using Java, Play Framework, and PostgreSQL. The crawling process includes page retrieval, parsing, normalization, deduplication, data storage, and presentation through the Inaprocplus application. Testing was conducted on several public LPSE portals using 1,000 tender records as samples. The results show a parsing accuracy of 97.3%, a parsing error rate of 2.7%, an average execution time of 17.17 seconds per 100 records, and a user satisfaction rate of 90%. These results indicate that a BFS-based web crawler can support tender data integration in a faster, more structured, and more practical way.*

Keywords — *Breadth-First Search, Inaprocplus, LPSE, Tender, Web Crawler.*

I. Introduction

The development of information technology has increased the need for fast, accurate, and accessible data. This need is also important in government procurement, where tender information becomes a basis for suppliers to identify opportunities, prepare documents, and decide whether a procurement package is worth pursuing.

In Indonesia, government procurement information is published through LPSE, an electronic procurement service used by ministries, institutions, and local governments. However, the information is still spread across many separate portals. A user who wants to search for tenders from several institutions must open one LPSE portal, enter a keyword, check the package list, open the detail page, and then repeat the same process on another portal. This workflow is time-consuming and depends heavily on user accuracy.

Tender information is also dynamic. A package may move from announcement to evaluation, qualification verification, and winner determination. The schedule may also change. If users receive the information too late, they may lose the chance to participate in a relevant tender. Therefore, a system is needed to collect tender information from many LPSE portals and present it in a more centralized form.

One realistic approach is the use of a web crawler. A web crawler is a software component that can automatically visit web pages and collect specific information. In this study, the web crawler is used to collect tender data from public LPSE portals. The collected data are then processed and displayed through the Inaprocplus application.

To keep the crawling process organized, this study applies the Breadth-First Search (BFS) algorithm. BFS explores nodes or pages level by level. This approach is suitable for LPSE page structures, which usually move from a package list page to a package detail page and then to related supporting pages such as schedules or announcements.

Based on this background, the purpose of this study is to implement a BFS-based web crawler in the Inaprocplus application so that tender data collection and presentation can be performed in a more automatic, structured, and efficient way.

II. Literature Review

A. Web Crawler

A web crawler is a program that automatically visits web pages. The crawling process usually starts from one or more initial URLs called seed URLs. After the first page is retrieved, the crawler reads the page content, identifies new links, and adds relevant links to the list of pages to be visited next.

In this study, the crawler does not only open web pages. It also collects relevant tender attributes from LPSE pages, such as package name, institution, budget ceiling, work location, procurement method, tender status, and source link. Therefore, crawling must be followed by parsing, normalization, deduplication, and database storage.

B. Breadth-First Search

Breadth-First Search is a graph traversal algorithm that explores nodes in a level-by-level manner. The algorithm visits all nodes at the current level before moving to the next level. In practice, BFS commonly uses a queue data structure.

In web crawling, a node can be interpreted as a URL or web page. The seed URL is inserted into the queue. The crawler then takes the first URL from the queue and processes it. If the page is successfully retrieved, the crawler extracts data and finds new links. Relevant and unvisited links are added back to the queue. This process continues until the queue is empty or the crawling limit is reached.

The main advantage of BFS is its structured traversal flow. For LPSE portals, this is useful because the crawler can process

Abdul Wahid and Astie Darmayantie, "Implementation of a Web Crawler in the Inaprocplus Application Using the Breadth-First Search Algorithm," *International Research Journal of Advanced Engineering and Science*, Volume 11, Issue 2, pp. 72-74, 2026.

pages closest to the initial source first. As a result, the system is less likely to go too deep into irrelevant pages.

C. Inaprocplus

Inaprocplus is an application designed to integrate government procurement information. In this study, Inaprocplus acts as a platform for storing and displaying tender data collected from several LPSE portals. With the web crawler module, users can search for tender information without manually visiting many LPSE websites.

D. Java and PostgreSQL

The system was developed using Java with Play Framework as the web application framework. Java was selected because it supports structured and maintainable application development. PostgreSQL was used as the database to store crawling results. Storing data in a database makes tender information easier to search, filter, update, and display through the application.

III. Research Method

This study uses an implementation-based approach. It focuses on building a web crawler module in the Inaprocplus application and evaluating its performance using several testing indicators. The research object is the tender data collection process from public LPSE portals that can be accessed without authentication.

The research stages are as follows. First, seed URLs from selected LPSE portals are defined. Second, the crawling flow is designed using BFS. Third, a parser is developed to read HTML structures and extract required tender attributes. Fourth, the extracted data are normalized so that the format becomes consistent. Fifth, the data are stored in PostgreSQL. Sixth, the data are displayed through the search page in Inaprocplus. Finally, system testing is conducted to evaluate performance and user acceptance.

The BFS-based crawling flow can be described as follows:

1. The seed URL is inserted into the queue.
2. The system takes the first URL from the queue.
3. The system checks whether the URL has been visited.
4. If it has not been visited, the page is retrieved through an HTTP request.
5. The parser reads the page and extracts tender data.
6. The extracted data are normalized and stored in PostgreSQL.
7. New links are filtered based on relevance.
8. Relevant and unvisited links are inserted back into the queue.
9. The process continues until the queue is empty or the crawling limit is reached.

Testing was conducted on several active LPSE portals, including LPSE of the Ministry of Transportation, Ministry of Finance, Ministry of Defense, DKI Jakarta Provincial Government, West Java Provincial Government, and other regional LPSE portals. A total of 1,000 tender records were used as testing samples. The evaluation covered execution time, parsing accuracy, parsing error rate, efficiency compared with manual search, and user satisfaction.

IV. Results and Discussion

A. System Implementation

The implementation results show that the web crawler module works according to the designed flow. The crawler starts from the LPSE seed URL and explores pages gradually using BFS. Package list pages are processed first, followed by relevant package detail pages.

After a page is retrieved, the parser reads the HTML structure and extracts important data. The extracted data are then normalized, such as by standardizing number formats, date formats, institution names, and tender statuses. Deduplication is also applied to prevent duplicate records, either because the same URL is visited again or because a tender package appears on more than one page.

The cleaned data are stored in PostgreSQL. Users can then search for tender information through the Inaprocplus application. The search results display key information such as package name, institution, budget ceiling, and the original LPSE source link.

B. Application Performance Testing

Performance testing was conducted to evaluate whether the system can display search results quickly and accurately. The results are shown in Table I.

TABLE I. Application Display Performance Testing Results

Test Parameter	Average Value	Unit	Description
Search page response time	2.85	seconds	Time from clicking Search until results are displayed
Displayed data per search	100	records	Average data per keyword
Parsing error ratio	2.7	%	Data that failed to be read from LPSE pages
Accuracy compared with original LPSE data	97.3	%	Match rate of crawled data

Table I shows that the search page has an average response time of 2.85 seconds. This indicates that data stored in the database can be displayed quickly enough for routine tender searching. The 97.3% accuracy also shows that most crawled data match the original LPSE data.

C. Quantitative Testing Results

Further testing was conducted using several scenarios, including keyword-based tender searches, budget ceiling filters, and empty searches. The results are shown in Table II.

TABLE II. Quantitative Testing Results

No	Test Activity	Number of Records	Execution Time (seconds)	Accuracy (%)	Parsing Error (%)	Status
1	Tender search "computer"	200	27.3	98.1	1.9	As expected
2	Tender search "medical equipment"	150	22.5	96.4	3.6	Accepted
3	Minimum–maximum budget filter	100	10.7	100	0	Valid
4	Empty search "food"	100	8.2	-	-	No data found

Abdul Wahid and Astie Darmayantie, "Implementation of a Web Crawler in the Inaprocplus Application Using the Breadth-First Search Algorithm," *International Research Journal of Advanced Engineering and Science*, Volume 11, Issue 2, pp. 72-74, 2026.

Based on these results, the average execution time was 17.17 seconds per 100 records. Compared with manual searching, which required an average of 22.6 seconds for similar data, the system improved efficiency by about 24%. This improvement occurs because users no longer need to open several LPSE portals one by one. The data have already been collected, stored, and made searchable through a single application.

D. Accuracy and Parsing Error Analysis

The parsing accuracy of 97.3% indicates that the parser can read most tender data correctly. However, there is still a 2.7% parsing error rate. These errors generally occur because LPSE portals may have different HTML structures. Some portals use different table structures, class names, or information placement. When the page structure does not match the parsing rules, some data may fail to be extracted.

This finding shows that system success does not depend only on the BFS algorithm. BFS helps organize page traversal, while the parser determines whether the data in each page can be extracted correctly. For this reason, the parser needs to be maintained regularly, especially when LPSE page structures change.

E. User Satisfaction

In addition to technical testing, this study also measured user satisfaction through a questionnaire involving 20 internal respondents. The results are shown in Table III.

TABLE III. User Satisfaction Survey Results

Assessment Aspect	Satisfaction Percentage (%)	Description
Search result access speed	90	Fast and responsive
Interface usability	85	Easy navigation
System stability	95	No crash/error
Overall average	90	Very good

The survey results show an average user satisfaction score of 90%. System stability received the highest score, at 95%. This indicates that the application was stable during testing. The interface usability score of 85% is still good, but it also suggests that the user interface can be improved, for example by simplifying filters or making package details more informative.

V. CONCLUSION

Based on the implementation and testing results, a BFS-based web crawler can be applied to the Inaprocplus application to support tender data integration from multiple LPSE portals. BFS makes the page traversal process more gradual, structured, and easier to control. This pattern is suitable for LPSE page structures, which generally move from package list pages to package detail pages.

The system developed using Java, Play Framework, and PostgreSQL can perform crawling, parsing, normalization, deduplication, data storage, and tender search. The testing results show a parsing accuracy of 97.3%, a parsing error rate of 2.7%, an average execution time of 17.17 seconds per 100 records, and an efficiency improvement of about 24% compared with manual searching. The user survey also shows an average satisfaction rate of 90%.

These results indicate that Inaprocplus can help users obtain tender information faster and in a more centralized way. However, further development is still needed, especially in parser maintenance, expansion of LPSE sources, interface improvement, and automatic monitoring of tender status changes.

REFERENCES

[1] U. Alkindi, N. Akhmad, Y. Kartiko, T. Putro, and Rumini, "Implementasi algoritma Breadth First Search pada Pacman untuk mengatur pergerakan karakter,", vol. 5, no. 6, pp. 599–604,2018.
[2] Anwari and Hozairi, "Perbandingan algoritma Breadth First Search dan Dijkstra untuk penentuan rute terpendek pengiriman barang Unilever,", vol. 2, no. 1, pp. 67–72, 2019.
[3] R. Ardiansyah, "Implementation of hybrid algorithm Depth-First Search and Breadth-First Search on document repository system of student work,", vol. 1, no. 1, pp. 65–76, 2018.
[4] R. Mitchell, Web Scraping with Python: Collecting More Data from the Modern Web. Sebastopol, CA: O'Reilly Media, 2018.
[5] Nurdin, M. Hutomi, M. Qamal, and Bustami, "Sistem pengecekan toko online asli atau dropship pada Shopee menggunakan algoritma Breadth First Search,", vol. 4, no. 6, pp. 1117–1123,2019.
[6] J. Pardede, A. N. Herman, and G. Swarghani, "Perbandingan metode Breadth First Search dan backlink pada web crawler,", vol. 2, no. 2, pp. 61–69, 2017.
[7] Presidential Regulation of the Republic of Indonesia Number 16 of 2018 concerning Government Procurement of Goods/Services, 2018.
[8] I. Ramadhan and H. Sastramihardja, "Pemanfaatan web crawler dalam mengumpulkan informasi melalui internet,", pp. 548–553, 2018.
[9] R. T. Shita and Subandi, "Implementasi algoritma BFS (Breadth-First Search) pada aplikasi web crawler,", vol. 8, no. 2, pp. 127-132, 2016.
[10] A. Stanley, "Analisis penerapan algoritma DFS dan BFS pada web crawling/spider," 2019.
[11] A. Surahman, A. F. Octaviansyah, and D. Darwis, "Teknologi web crawler sebagai alat pengembangan market segmentasi untuk mencapai keunggulan bersaing pada e-marketplace,", vol. 15, no. 1, pp. 20–28, 2020.
[12] Tarmiandi, E. Z. Astuti, and S. Astuti, "Implementasi algoritma Breadth First Search pada pencarian rute terpendek tempat kos di Semarang Tengah,", vol. 1, no. 1, pp. 524–528, 2018.
[13] A. E. Wibowo, K. Muslim, and Lhaksmana,"Perbandingan performansi terhadap algoritma Breadth First Search (BFS) dan Depth First Search (DFS) pada web crawler,", vol. 6, no. 2, pp. 9905–9914, 2019.