COMP 4321 Search Engine Project Report

Overall Design of the System

The developed system is a comprehensive Vector-Space Model-based search engine leveraging sophisticated indexing and retrieval techniques. The main components include:

Web Crawling

- Implemented through the Breadth-First Search (BFS) algorithm (HtmlParser.java), systematically retrieving and indexing web pages starting from a given URL.
- Manages metadata extraction including page title, last modification date, and page size.
- Handles errors and broken links gracefully, ensuring continuous crawling without interruptions.

Web crawling efficiently traverses web pages and manages visited URLs to avoid redundancy, thus optimizing resource usage and indexing performance.

Indexing and Storage

- Managed by the InvertedIndexManager.java utilizing the JDBM database system.
- Employs an inverted index mechanism for efficient document retrieval based on keywords.
- Supports indexing of both individual words and phrases (n-grams up to 3).

The inverted index structure significantly speeds up the query response time by allowing quick look-up of documents containing specific keywords or phrases.

Search Functionality

- Provided by SearchEngine.java, processing user queries, ranking documents based on relevance, and handling exact phrase searches.
- Implements relevance score calculation, enhanced by normalization and title-field boosting to prioritize matches appearing in document titles.

Advanced query processing algorithms ensure accurate and contextually relevant search results, greatly enhancing the user experience.

User Interfaces

- Command-line interface (SearchProgram.java) facilitating straightforward interaction through terminal queries.
- Web-based interface (SearchServlet.java and index.jsp) offering a user-friendly search experience, displaying results along with metadata, keywords, and related child and parent links.

The web application, specifically designed using Java servlets and JSP (index.jsp), ensures a responsive and intuitive user interface, enabling seamless navigation and clear presentation of comprehensive search results and associated metadata.

File Structures in the Index Database

The database employs a structured JDBM system with the following primary storage structures:

- pageIdToUrlMap and urlToPageIdMap: Bidirectional mapping between page IDs and URLs.
- wordIdToWordMap and wordToWordIdMap: Efficient management of vocabulary within the database.
- bodyInvertedIndex and titleInvertedIndex: Separate inverted indexes for rapid retrieval of body and title occurrences of indexed terms.
- pageInfoMap: Stores essential metadata such as URL, page title, modification date, page size, and links.
- pageIdToBodyWordsMap and pageIdToTitleWordsMap: Maintain sequences of indexed words to facilitate precise phrase searching.
- maxTFForPageId: Tracks maximum term frequencies within pages for TF-IDF normalization.

These database structures allow streamlined and rapid access to information, essential for high-performance search operations.

Algorithms Used

1) Relevance Ranking

- Measures document relevance based on term frequency (TF) and inverse document frequency (IDF).
- Normalizes term frequencies by the maximum term frequency within each document.

2) Title-Field Boosting

- Increases the significance of keywords appearing in titles by applying a boost factor (set at 5.0).

3) Phrase Matching

- Supports precise phrase searches, parsing user queries for quoted phrases and ensuring results contain exact phrase occurrences.

4) Breadth-First Search (BFS) for Web Crawling

- Implements systematic exploration, avoiding redundant indexing by tracking already visited URLs.

5) Stop Word Removal and Stemming

- Utilizes StopStem.java, employing the Porter stemmer to reduce vocabulary size and complexity.
- Removes common stop words from indexing and queries to enhance performance and relevance.

Each algorithm contributes uniquely to the robustness and accuracy of the search engine, optimizing search result quality.

Installation Procedure

To install and run the system:

1. Ensure Java and JDBM libraries are correctly installed and configured.
2. Place stopwords.txt and all required files in the project directory.
3. Run: mvn clean compile
4. Run: mvn exec:java -Dexec.mainClass="HtmlParser"
5. Run: mvn jetty:run
6. Open your browser and go to http://localhost:8080/ to access the search engine web interface

A straightforward installation procedure facilitates rapid deployment and testing of the system.

Highlights of Additional Features

- Enhanced indexing: Implementation of multi-word indexing (up to 3-grams), improving phrase search capabilities.
- Comprehensive metadata management: Extensive metadata (URLs, modification dates, page sizes, child and parent links) enhances the search result context.
- Robust Error Handling: Graceful management of URL and network-related errors during crawling, ensuring uninterrupted system operations.
- User-Friendly Interfaces: Both command-line and web-based interfaces deliver accessible search experiences and clear presentation of results.

These additional features significantly differentiate the system by enhancing reliability and user-friendliness.

Testing of Implemented Functions

Testing involved extensive use of TestProgram.java for database integrity and functionality validation, as well as real-time command-line (SearchProgram.java) and web interface (SearchServlet.java) interaction.

- Command-line interactions confirmed accuracy and response time.
- Web interface validated comprehensive result metadata, including child and parent links.
- Crawling performance was monitored for indexing accuracy, speed, and resilience to network errors.

Extensive testing ensures system stability and performance consistency across different usage scenarios.

Bonus Features:

Bonus 1:
The system includes an advanced relevance feedback feature similar to Google's "Get Similar Pages". Users can enhance their search experience by clicking the "Get Similar Pages" button, which automatically identifies the top five most frequent keywords from the selected page, excluding common stop words. These extracted keywords are used to rewrite and refine the original query dynamically, submitting it for a new search to yield closely related documents. This powerful feature not only improves the accuracy of search results but also significantly enhances user interaction and satisfaction, making the search process more intuitive and efficient.

Bonus 2:

The system features an intelligent search history management capability that enhances user experience by automatically tracking and displaying the three most recent search queries. This convenient feature provides users with quick access to their previous searches through clickable history items displayed directly below the search box. When a user clicks on any recent search item, the system immediately reloads the corresponding search results without requiring manual reentry of the query. The implementation uses session-based storage to maintain search history across page visits while prioritizing recent queries, automatically removing older entries when new searches are performed. This streamlined approach to search history not only reduces repetitive typing but also encourages exploration by making it effortless to revisit and compare previous search results..

Conclusion

Strengths:

- Robust and accurate indexing facilitated by efficient database structures.
- Effective and nuanced relevance ranking provided by TF-IDF weighting and title boosting.

- Excellent support for exact phrase searches enhances the overall precision of query results.

Weaknesses:

- Performance scalability could degrade with extremely large-scale datasets due to database limitations.
- Dependency on JDBM restricts flexibility and may limit performance optimization potential.

Suggested Improvements:

- Introduce parallel web crawling to increase indexing throughput.
- Optimize database transactions and improve caching strategies to enhance performance.
- Refine interface responsiveness, particularly for the web-based search service.

Interesting Features to Add:

- Real-time incremental indexing for dynamic content updates.
- Integration of user feedback mechanisms to further refine search accuracy.
- Advanced personalization of search results based on user interaction history and preferences.

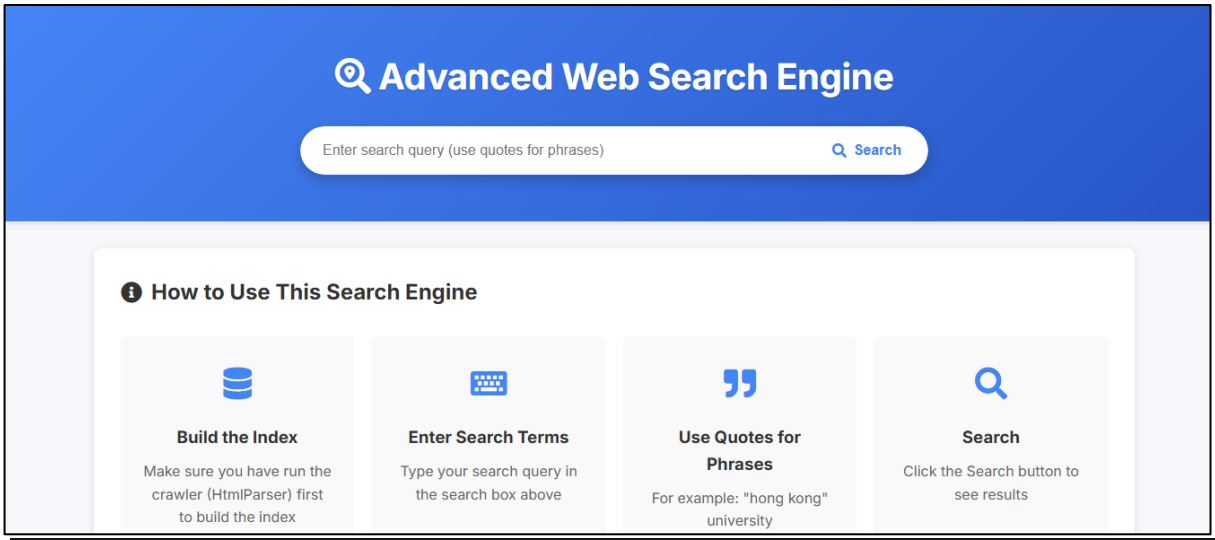
Future enhancements could significantly broaden the usability and efficiency of the search engine.

Contribution

- Alex: 33.3% – Developed web crawler and indexing components.
- Alvin: 33.3% – Designed and optimized the core search engine algorithms and backend processes.
- Shahman: 33.3% – Developed the retrieval function and the web interface.

Clear division of roles facilitated efficient teamwork and ensured comprehensive coverage of all system aspects.

Appendix




```
Search time: 11 ms; Display time: 75 ms
Phrases: []
Terms: [hong, kong]
Candidates: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48,
 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73
, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 9
8, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 11
8, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 1
38, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157,
158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177,
 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197
, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 21
7, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 2
37, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256,
257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276,
 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296
]
Search time: 8 ms; Display time: 36 ms
```