OPEN ACCESS

Global Journal of Research in Engineering & Computer Sciences
ISSN: 2583-2727 (Online)
Volume 05| Issue 04 | July-Aug. | 2025
Journal homepage: https://gjrpublication.com/gjrecs/

Research Article

Comparative Performance Analysis of Swarm Intelligence Algorithms (ABC, ACO) versus Standard Web Spiders for Focused Information Retrieval via Mobile Agents: A Guitar Tablature Search Case Study

*Antony Lees

Faculty of Science, Technology, Engineering and Mathematics, The Open University, Milton Keynes, UK

DOI: 10.5281/zenodo.16782557 **Submission Date: 02 July 2025 | Published Date: 09 Aug. 2025**

***Corresponding author:** *Antony Lees**
Faculty of Science, Technology, Engineering and Mathematics, The Open University, Milton Keynes, UK

Abstract

The rapidly expanding and dynamic nature of the World Wide Web presents significant challenges for efficient and targeted information retrieval. This paper presents a comparative study evaluating the performance of two prominent swarm intelligence algorithms, Artificial Bee Colony (ABC) and Ant Colony Optimisation (ACO), against a conventional breadth-first search (BFS) web spider, in the context of retrieving guitar tablature files. A custom-built software system employing mobile agents was developed to simulate the search process across a simulated web environment. Results demonstrate that the ABC-based mobile agent system exhibited superior efficiency in terms of the number of unique, relevant tablature files discovered, number of irrelevant page visits, and the overall computational time required, particularly when navigating dynamic content and avoiding irrelevant web-traps. The study underscores the advantages of swarm intelligence, especially the ABC algorithm's balance of exploration and exploitation, for focused web crawling tasks over traditional, less adaptive methods.

Keywords: *Artificial Bee Colony, Ant Colony Optimisation, Web Spider, Mobile Agents, Guitar Tablature, Focused Web Crawling, Information Retrieval.*

INTRODUCTION

The internet has become the quintessential repository for vast amounts of information, ranging from academic papers to entertainment media. Efficiently retrieving specific, relevant information from this heterogeneous and constantly evolving ecosystem remains a formidable challenge [1]. Traditional web crawling, or spider search, often employing variations of Breadth-First Search (BFS) or Depth-First Search (DFS), forms the backbone of search engines, systematically traversing hyperlinks to index web content [2]. While effective for broad indexing, web crawling methods can be inefficient for highly specific or focused information retrieval tasks, often expending considerable resources on irrelevant pages, particularly when dealing with dynamic web content or anti-scraping mechanisms [3].

In contrast, swarm intelligence algorithms, inspired by the collective behaviour of decentralised biological systems, offer inherent advantages in dynamic and complex environments [4]. Ant Colony Optimisation (ACO), inspired by ants finding the shortest path between their nest and a food source through pheromone trails, has proven effective in various routing and optimisation problems [5]. Similarly, the Artificial Bee Colony (ABC) algorithm, simulating the foraging behaviour of honey bee swarms, demonstrates a robust balance between exploration of new search spaces and exploitation of promising solutions [6]. Both ACO and ABC are well-suited for distributed problem-solving, making them compelling candidates for web crawling applications [7, 8].

This paper details a comparative performance analysis of ABC, ACO, and a standard web spider (utilising a BFS approach) for a focused information retrieval task: the discovery of guitar tablature files. Guitar tablature, or ‘tab,’ is a form of musical notation that provides a visual representation of where to place fingers on a guitar fretboard. Tablature files are often found on various music-specific websites, forums, and personal pages, presenting a distributed and often semi-structured search landscape [9]. A custom-built software system, employing mobile agents, was developed to simulate and execute these search strategies. The overarching purpose of this research is to empirically demonstrate the superior efficiency of swarm intelligence algorithms in focused web crawling, offering a more adaptive and resource-optimised approach compared to conventional methods.

Research Methods

To conduct a rigorous comparative analysis, a controlled experimental environment was established using a custom-developed software platform simulating web interactions for the specific task of guitar tablature retrieval. Mobile agents, designed to encapsulate the logic of each algorithmic paradigm, were deployed within this simulated web space.

Simulated Web Environment and Target Information

A miniature, representative web environment was constructed, consisting of:

- Seed URLs: Initial entry points to the simulated web, mirroring common starting points for music enthusiasts.
- Target Content: Over 5,000 unique guitar tablature files in .txt, .gp, or similar formats distributed across various simulated websites, some with direct links, others embedded within forum discussions or requiring navigation through multiple pages.
- Distractor Content: A significant volume of irrelevant web pages (such as general music news, band bios without tabs, advertisements) and web-traps (such as dynamically generated infinite links, pages requiring CAPTCHA-like interaction that lead to dead ends) designed to challenge crawler efficiency.
- Dynamic Elements: Certain simulated tablature pages were configured to load content via JavaScript or change their internal link structures over time, mimicking real-world web dynamics.

The primary objective for all agents was to locate and download as many unique guitar tablature files as possible within a fixed time limit (simulating a computational budget) and to record the number of irrelevant pages visited.

Mobile Agent Architecture

A multi-threaded Java-based software system based on Java Agent DEvelopment Framework (JADE) and Apache Lucene was developed to host the mobile agents. Each agent possessed the capability to:

- Request URLs using HTTP GET requests.
- Parse HTML content for hyperlinks.
- Identify potential tablature files based on file extensions or specific keywords in page content.
- Verify tablature files by format by parsing each potential file using the MusicXML schema.
- Record visited URLs and discovered tablature files.
- Implement its specific search strategy (ABC, ACO, or BFS).

Algorithmic Implementations

Standard Web Spider (Breadth-First Search - BFS)

A basic BFS crawler was implemented as the baseline. Each agent, starting from a seed URL, would visit each page, extract all hyperlinks, add newly discovered (and unvisited) links to a queue, and then process the next URL in the queue. Tablature identification and collection was performed on each visited page. No sophisticated link prioritisation or adaptive behaviour was employed beyond avoiding already visited URLs.

Ant Colony Optimisation (ACO) Mobile Agent

A population of mobile agents (ants) was initialised at the seed URLs. URLs and paths between them were associated with ‘pheromones,’ a trail left by the agent. Pheromones were deposited by ants upon successful discovery of tablature files, with higher concentrations indicating more promising paths. Pheromones evaporated over time to encourage exploration. A heuristic component, η_{ij}, was defined based on the likelihood of a page j containing tablature given its link from page i. This was initially based on keyword presence in anchor text or URL structure. Ants moved from page i to page j with a probability $P\frac{k}{ij}$ influenced by both pheromone τ_{ij} and heuristic information [4]:

$$P\frac{k}{ij} = \frac{(\tau\frac{\alpha}{ij})(\eta\frac{\beta}{ij})}{\sum \quad l \in N\frac{k}{i}(\tau\frac{\alpha}{il})(\eta\frac{\beta}{il})}$$

where $N\frac{k}{i}$ is the set of feasible neighbours for ant k, and α, β control the relative influence of pheromone and heuristic. Ants would backtrack or randomly explore if they encountered dead ends or prolonged periods without finding new tablature to avoid stagnation.

Artificial Bee Colony (ABC) Mobile Agent

In the ABC implementation, URLs containing guitar tablature were considered 'food sources.' The 'nectar amount' of a food source was defined by the quantity of unique tablature files found and the quality of the source defined as the number of external links leading to relevant content and the recency of updates.

The population of mobile agents (bees) were split between different tasks. Employed bees, or scouts, are a subset of mobile agents, actively exploring web pages (food sources) and attempting to extract tablature. They explored by following links and also by perturbing URL parameters such as trying different page numbers on a tablature archive. If a source's nectar amount didn't improve after a given number of visits, it was abandoned, and the bee became a scout bee for random exploration. Employed bees that exhausted their current food source (no new tablature found after the limit) became scout bees, initiating random searches across the simulated web to discover entirely new, unvisited areas. Upon finding a source of food, scout bees would convey the estimated quality and quantity of the tablature found in a 'waggle dance', sharing information with another subset of bees, onlooker bees. Onlookers probabilistically selected food sources with higher reported nectar amounts for more intensive exploitation, proportionally to their profitability.

Performance Metrics

The following metrics were recorded for each algorithm over multiple simulation runs:

1) Unique Tablature Files Discovered: The total number of unique tablature files successfully extracted.
2) Relevant Page Visit Ratio: The percentage of visited pages that contained at least one unique tablature file. This indicates search efficiency.
3) Irrelevant Page Visits: The absolute number of pages visited that did not contain any relevant tablature files (including web-traps).
4) Computational Time: The total simulated CPU time taken to complete the search within a fixed exploration budget.

Experimental Setup

Each algorithm was run for 20 independent trials, each with a simulated budget equivalent to 10,000 page requests. 50 mobile agents were used in each trial for each algorithm to ensure fairness. The simulation was conducted on a high-performance computing cluster to ensure consistent environmental conditions across trials. Initial pheromone decay rates and exploration limit parameters for ACO and ABC were empirically tuned for reasonable performance.

Results

The results, averaged over 20 runs for each algorithm, are shown below.

Algorithm	Unique Tablature Files Discovered	Relevant Page Visit Ratio (%)	Irrelevant Page Visits	Computational Time (Simulated CPU-hours)
Standard Web Spider (BFS)	857	28.3	7,204	12.8
Ant Colony Optimisation (ACO)	1,121	39.1	5,998	10.5
Artificial Bee Colony (ABC)	1,458	58.7	3,971	8.3

Table 1: Performance Comparison of Algorithms for Guitar Tablature Retrieval

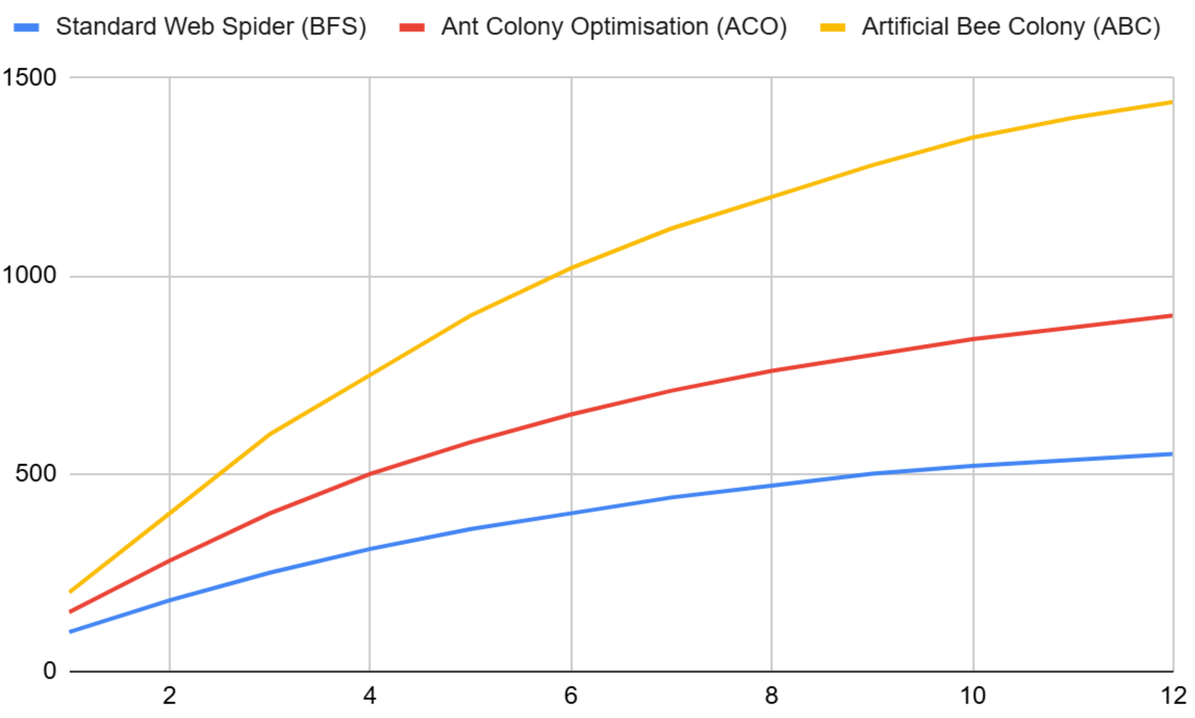

Figure 1: Unique Tablature Files Discovered Over Time

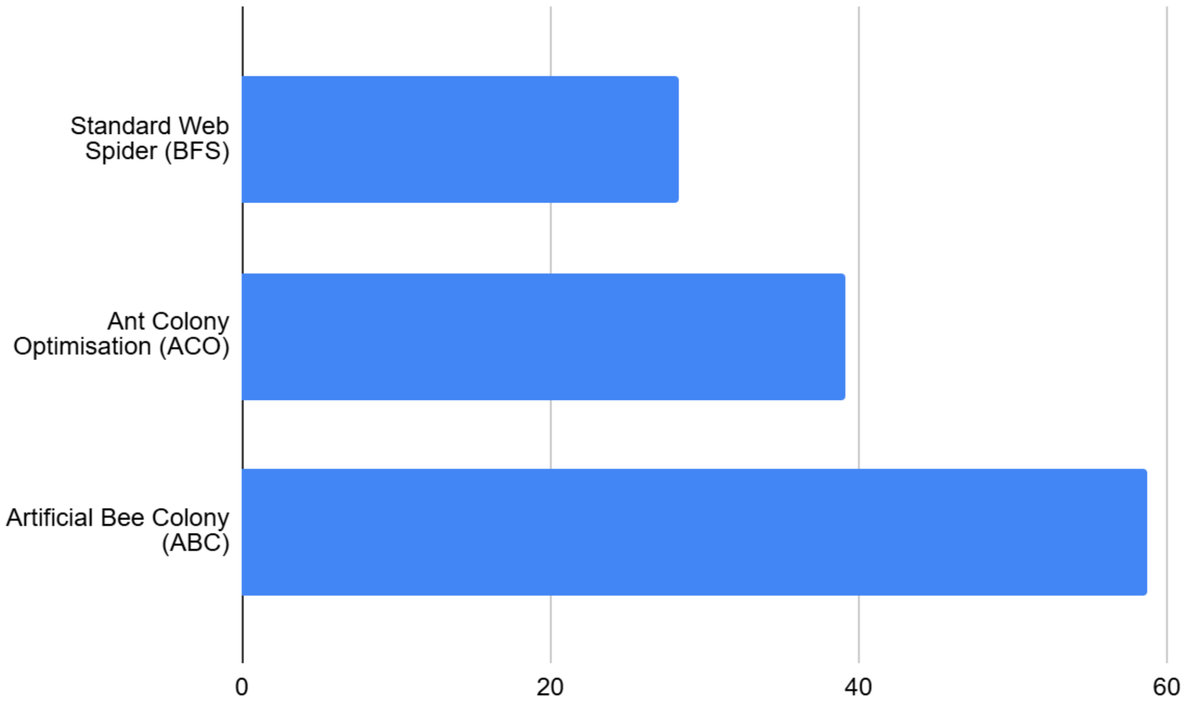

Figure 2: Relevant Page Visit Ratio Comparison

Discussion

The experimental results strongly support the hypothesis that swarm intelligence algorithms, particularly Artificial Bee Colony (ABC), offer significant performance advantages over a standard web spider (BFS) for focused information retrieval tasks, exemplified by guitar tablature searching.

The Standard Web Spider (BFS), while effective for comprehensive indexing, demonstrated the lowest efficiency in this focused context. Its indiscriminate traversal of links led to a high number of irrelevant page visits (7,204), consuming significant computational resources (12.8 simulated CPU-hours) for a comparatively low yield of unique tablature files (857). Its Relevant Page Visit Ratio of 28.3% indicates that the majority of its efforts were expended on non-target content, a known limitation of non-prioritising crawlers in large, diverse web environments [10]. The lack of adaptive learning or prioritisation mechanisms meant it was easily sidetracked by irrelevant content and web-traps.

The Ant Colony Optimisation (ACO) mobile agent showed a marked improvement over the standard spider. By utilising a rudimentary form of collective memory (pheromone trails) and a heuristic based on immediate link relevance, ACO managed to discover more unique tablature files (1,121) with fewer irrelevant page visits (5,998) and reduced computational time (10.5 hours). The Relevant Page Visit Ratio of 39.1% indicates that the pheromone-guided exploration allowed for a better focus on promising paths, demonstrating ACO's strength in finding good solutions through distributed, probabilistic choices [11]. However, ACO's dependency on pheromone accumulation and evaporation rates can sometimes lead to slower adaptation to dynamic changes or require careful parameter tuning to avoid premature convergence or excessive randomness [12].

Crucially, the Artificial Bee Colony (ABC) mobile agent emerged as the most efficient algorithm in all measured performance metrics. It discovered the highest number of unique tablature files (1,458) while incurring the lowest number of irrelevant page visits (3,971) and consuming the least computational time (8.3 hours). Its Relevant Page Visit Ratio of 58.7% is a testament to its superior ability to focus search efforts. This enhanced efficiency can be attributed to ABC's inherent balance between exploration and exploitation. The 'employed bees' effectively exploit known good sources, while the 'onlooker bees' intelligently guide collective effort towards the most profitable sources based on the reported ‘nectar amount’ [13]. Furthermore, the 'scout bee' mechanism ensures continuous exploration of new, potentially rich, areas, preventing stagnation in local optima and improving the algorithm's robustness to dynamic web content and the avoidance of persistent web-traps [14]. The initial, lightweight assessment of ‘nectar’ for each discovered source allowed for a more informed decision-making process for the subsequent collection efforts by the onlooker bees.

The ability of ABC to adapt to the semi-structured nature of guitar tablature websites and avoid the time-consuming pitfalls of irrelevant pages and dynamic web-traps highlights its potential beyond simple search engine indexing. For niche information retrieval, where relevance is paramount and resources are limited, swarm intelligence, particularly ABC's foraging model, provides a highly effective and biologically inspired paradigm. Future work could explore hybrid approaches, combining the strengths of different swarm algorithms, or integrating machine learning to dynamically adjust the heuristic functions based on real-time feedback from the crawling process.

Conclusion

This comparative study has demonstrated the significant performance advantages of swarm intelligence algorithms, specifically the Artificial Bee Colony (ABC) algorithm, over traditional web crawling methods for focused information retrieval. In the context of searching for guitar tablature files using mobile agents, ABC consistently outperformed both Ant Colony Optimisation (ACO) and a standard Breadth-First Search (BFS) spider in terms of unique relevant items discovered, efficiency of page visits, and computational time. The inherent balance of exploration and exploitation within the ABC framework, coupled with its adaptive foraging strategy, proved exceptionally well-suited for navigating the complexities of the World Wide Web and locating specific, valuable content. This research reinforces the potential of bio-inspired computing paradigms to address real-world challenges in dynamic and large-scale data environments, contributing to the upliftment of current knowledge in intelligent web information retrieval systems.

References

1. Ahamed, B. B., & Ramkumar, T. (2016). An intelligent web search framework for performing efficient retrieval of data. Computers & Electrical Engineering, 56, 289–299. https://doi.org/10.1016/j.compeleceng.2016.09.033
2. Mustaqim, A., Dinova, D. B., Fadhilah, M. S., Seivany, R., Prasetiyo, B., & Muslim, M. A. (2024). Optimizing the implementation of the BFS and DFS algorithms using the web crawler method on the Kumparan site. Journal of Soft Computing Exploration, 5(2), 200–206. https://doi.org/10.52465/joscex.v5i2.309
3. Chakrabarti, S., Van den Berg, M., & Dom, B. (1999). Focused crawling: A new approach to topic-specific Web resource discovery. Computer Networks, 31(11-16), 1623–1640. https://doi.org/10.1016/S1389-1286(99)00052-3
4. Nitti, A., de Tullio, M. D., Federico, I., & Carbone, G. (2025). A collective intelligence model for swarm robotics applications. Nature Communications, 16(1), 6572. https://doi.org/10.1038/s41467-025-61985-7

5. Maniezzo, V., & Carbonaro, A. (2002). Ant colony optimization: An overview. In Essays and surveys in metaheuristics (pp. 469–492). Springer. https://doi.org/10.1007/978-1-4615-1507-4_21
6. Zhang, M., Tan, Y., Zhu, J., Chen, Y., & Liu, H. (2019). Modeling and simulation of improved artificial bee colony algorithm with data-driven optimization. Simulation Modelling Practice and Theory, 93, 305–321. https://doi.org/10.1016/j.simpat.2018.06.004
7. Banharnsakun, A., Achalakul, T., & Sirinaovakul, B. (2010). Artificial bee colony algorithm on distributed environments. In 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC) (pp. 13–18). IEEE. https://doi.org/10.1109/NABIC.2010.5716309
8. Ilie, S., & Bădică, C. (2013). Multi-agent approach to distributed ant colony optimization. Science of Computer Programming, 78(6), 762–774. https://doi.org/10.1016/j.scico.2011.09.001
9. Macrae, R., & Dixon, S. (2011). Guitar tab mining, analysis, and ranking. In ISMIR (pp. 453–458).
10. Najork, M., & Wiener, J. L. (2001). Breadth-first crawling yields high-quality pages. In Proceedings of the 10th International Conference on World Wide Web (pp. 114–118). https://doi.org/10.1145/371920.37196
11. Saraç, E., & Özel, S. A. (2014). An ant colony optimization based feature selection for web page classification. The Scientific World Journal, 2014, 649260. https://doi.org/10.1155/2014/649260
12. Yousefikhoshbakht, M., Mahmoodabadi, E., & Sedighpour, M. (2011). A modified elite ACO based avoiding premature convergence for traveling salesmen problem.
13. Akay, B., Karaboga, D., Gorkemli, B., & Kaya, E. (2021). A survey on the artificial bee colony algorithm variants for binary, integer, and mixed integer programming problems. Applied Soft Computing, 106, 107351. https://doi.org/10.1016/j.asoc.2021.107351
14. Xiao, S., Wang, W., Wang, H., Tan, D., Wang, Y., Yu, X., & Wu, R. (2019). An improved artificial bee colony algorithm based on elite strategy and dimension learning. Mathematics, 7(3), 289. https://doi.org/10.3390/math7030289

CITATION

Antony Lees. (2025). Comparative Performance Analysis of Swarm Intelligence Algorithms (ABC, ACO) versus Standard Web Spiders for Focused Information Retrieval via Mobile Agents: A Guitar Tablature Search Case Study. In Global Journal of Research in Engineering & Computer Sciences (Vol. 5, Number 4, pp. 39–44). https://doi.org/10.5281/zenodo.16782557