Article

Website Internal Link Optimization Strategy and SEO Effect Evaluation Based on Dijkstra Algorithm

Yifan Yang [1,*]

[1] Viterbi school of Engineering, University of Southern California, Los Angeles, CA, 90089, USA
[*] Correspondence: Yifan Yang, Viterbi school of Engineering, University of Southern California, Los Angeles, CA, 90089, USA

Abstract: Since SEO is becoming more and more important for the improvement of page visits and ranking, adjusting the internal link of the website to achieve the best SEO effect has become an important link. Therefore, this study adopts Dijkstra algorithm to establish the optimization method of the internal link of the website and evaluate the SEO performance. Firstly, the application process of Dijkstra algorithm in network topology is briefly introduced, and the advantages and difficulties of Dijkstra algorithm in improving website internal chain are analyzed in detail, and suggestions are given. At the same time, the limitations encountered in the practical application of this method and the possible research direction in the future are pointed out.

Keywords: Dijkstra algorithm; website optimization; internal links; SEO effect; search engine ranking

Received: 06 April 2025
Revised: 12 April 2025
Accepted: 26 April 2025
Published: 29 April 2025

1. Introduction

With the development of the Internet, websites continue to expand the scale, SEO has become an effective way to improve website traffic and search engine ranking, and internal link optimization, is an indispensable part of SEO, can effectively improve page readability and search engine crawling ability. Dijkstra algorithm is a famous algorithm in graph theory, which has important practical significance in improving the structure of Internet links. This paper uses Dijkstra algorithm to optimize internal links for the purpose of case analysis of the impact of internal links on SEO.

2. Theoretical Basis of Website SEO and Internal Link Optimization

2.1. Basic Concepts and Importance of Website SEO

SEO (Search engine optimization) is the design and organization of web content so that it has a good position in search engine rankings, increasing the visibility of the website on the web and the number of views. The main goal of SEO is to increase the chances of relevant terms appearing at the top of search results so that the website gets more natural traffic.

SEO optimization includes: keyword optimization, internal chain optimization, etc., which internal chain optimization is through the design of the relationship between pages, so that search engines can more easily understand the structure of the site and improve the speed of search engine crawling and indexing [1].

2.2. Core Concepts of Dijkstra's Algorithm

In graph theory, the classic algorithm for solving the single source shortest route is Dijkstra's algorithm, which was proposed and applied by Dijkstra in 1956. The main idea is to continuously update the shortest distance from each station to the departure station to select the optimal path to each station, use greedy thinking to find the current best path, and then expand all stations and distances outward in turn. The core formula of Dijkstra's algorithm is as follows:

$$d(v) = \min(d(v), d(u) + w(u, v)) \quad (1)$$

Among them, $d(v)$ represents the current shortest path from the starting point to node v, $d(u)$ represents the shortest path from the starting point to node u, and $w(u, v)$ is the weight of the edge from node u to node v. By updating the formula, the path of node v is compared with the path passed from node u, and the shorter path is selected (Figure 1).

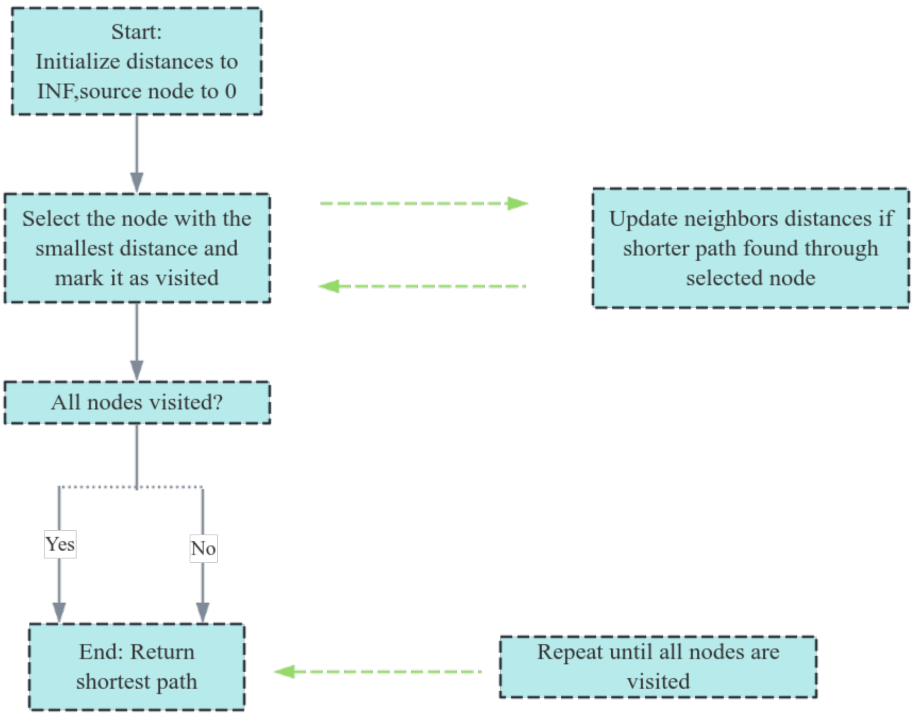
Start: Initialize distances to INF,source node to 0
Select the node with the smallest distance and mark it as visited
Update neighbors distances if shorter path found through selected node
All nodes visited?
Yes
No
End: Return shortest path
Repeat until all nodes are visited

Figure 1. Basic Flowchart of Dijkstra's Algorithm.

After completing the above operations, we can use Dijkstra algorithm to derive the source point and end point of the shortest path, and optimize the network topology accordingly. At present, Dijkstra algorithm is widely used, mainly in communication network, map navigation, website internal link optimization and other fields.

3. The Application of Dijkstra Algorithm in the Optimization of Internal Links of Websites

3.1. Dijkstra's Algorithm Cannot Deal With the Graph Containing Negative Weighted Edges

Dijkstra's algorithm is one of the most commonly used algorithms to find the optimal path, which works effectively when every link in the network has a positive value. If a negative line appears, Dijkstra's algorithm will not work properly. This is because when there is a negative connection, the calculation method of Dijkstra's algorithm will have a certain influence, resulting in the inaccurate calculation results of the optimal path [2]. This will make the judgment of Dijkstra's algorithm invalid, and it is very likely that wrong optimization schemes or unbalanced website weights will occur. Its algorithm assumes that a certain path will not be updated every time it selects the shortest path.

3.2. The Execution Speed of Dijkstra's Algorithm Is Slow When Optimizing Large-Scale Data

Dijkstra's algorithm is slow to perform on large data sets, especially if the number of pages on a website is large. In this case, the time complexity of Dijkstra's algorithm is

$O(V^2)$, where V is the number of nodes in the graph. This means that for large websites, if there are millions of nodes, the algorithm's traversal of nodes and edges and the number of updates will be very large, which will lead to an exponential increase in the execution time of the algorithm, and can not meet the response needs in real time [3].

The amount of data on large-scale websites is huge, resulting in very expensive calculations and updates. Although Dijkstra's algorithm itself can optimize its time complexity by using a priority queue such as a heap, the optimized time complexity is:

$$O((E+V)logV) \tag{2}$$

where E is the number of edges and V is the number of vertices. However, slow execution still occurs when more complex graphs are encountered. This problem is especially prominent when frequently modifying the content of the web page, because every time the content or structure of the web page changes, it needs to be re-optimized, leading to a large computational cost.

3.3. Websites with Different Structures Will Affect Dijkstra's Algorithm

The speed and optimization results of Dijkstra's algorithm are limited by the size and type of website structure. However, different structures (such as hierarchical structure, flat structure and dynamic structure) have very different influences on Dijkstra's algorithm (Figure 2).

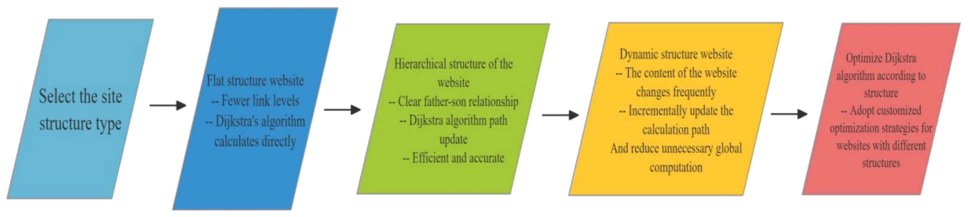
Select the site structure type
Flat structure website
-- Fewer link levels
-- Dijkstra's algorithm calculates directly
Hierarchical structure of the website
-- Clear father-son relationship
-- Dijkstra algorithm path update
-- Efficient and accurate
Dynamic structure website
-- The content of the website changes frequently
-- Incrementally update the calculation path
And reduce unnecessary global computation
Optimize Dijkstra algorithm according to structure
-- Adopt customized optimization strategies for websites with different structures

Figure 2. Application of Dijkstra Algorithm in Different Website Structures.

First of all, according to the page itself to determine the optimization strategy using Dijkstra strategy. For flat websites, the internal depth is not large and the connection is relatively direct, so Dijkstra's algorithm can find the optimal path in time. For hierarchical websites, they usually have obvious links between upper and lower pages, so Dijkstra's algorithm can quickly find routes in the hierarchical structure. Because the relationship between levels is determined, Dijkstra algorithm is not only efficient but also accurate, which ensures the stability and credibility of optimization results. Finally, for dynamic websites such as social networks and e-commerce platforms, content changes or page connections often change. At this time, Dijkstra's algorithm needs to be able to adapt to constantly changing scenarios [4].

3.4. The Results of Dijkstra's Algorithm Are Uncertain

In the process of network site optimization, the output result of Dijkstra algorithm after each operation may present a random phenomenon. Especially after changes in the content, strength, and structure of the site, the optimized path may be different each time the algorithm is run.

Dijkstra's algorithm is constantly improved with iterations, and each improvement relies on the weight and relationship between the current nodes to calculate the shortest path again. For example, in the first round, the shortest path distance from node 1 to node 2 is 5, but in the next round of iteration, the shortest path distance from node 1 to node 2 is 6, which indicates that the page topology or weight has changed [5]. Similarly, the shortest distance between node 1 and node 3 also changes in turn, which is caused by a change in the line relationship between the two nodes or a change in the weight relationship (Table 1).

Table 1. Uncertainty of Results in the Optimization Process of Dijkstra Algorithm.

Number of iterations	The shortest path from page 1 to page 2	The shortest path from page 1 to page 3	Optimization result variation
1	5	8	Initial path
2	6	7	Page weight update
3	4	9	Page structure adjustment
4	5	6	Optimized stability

The result after each retry is the optimized result. Sometimes these optimizations are due to dynamic changes in the content, link structure, and weight of the web page. For example, the shortest path from page 1 to page 2 has changed again after the third retry, indicating that the link structure has changed significantly.

4. Application and Adjustment of Dijkstra Algorithm in Website Internal Link Optimization

4.1. Build the Edge Weights of the Graph According to the Weight of the Page and the Importance of the Link

When using Dijkstra algorithm to optimize the internal links of a site, how to assign weights is the key. The page of each website can be used as the point of the graph, and the inner chain of each page can be used as the edge of the graph. The assignment of opposite sides enables Dijkstra's algorithm to find the shortest route and optimize the connection direction of each point in the station. To ensure the effectiveness of the algorithm, we need to assign appropriate weights according to the value and relevance of the page. How to determine page weight: Page weight can be given according to indicators, such as search engine ranking, traffic, content quality, and so on. In general, the page weight w_p can be expressed as:

$$w_p = f(SEO, Flow\ \ rate, Content\ \ quality, Number\ \ of\ \ external\ \ links)) \quad (3)$$

Where f is the weight calculation function that combines several factors to determine the importance of the page. SEO and content quality are core factors in page optimization, while traffic and the number of external links reflect the popularity of the page.

The value of links: The value of links is mainly reflected in the contact between pages and the type of links. Main navigation links, recommended links in SEO compared to ordinary content links more valuable. Setting link weight w_l can be expressed by the following formula:

$$w_l = f(Link\ \ type, Page\ \ weight, Access\ \ frequency) \quad (4)$$

Here, w_l is the weight of links between pages, and f is the link importance function.

Dijkstra algorithm can assign the weight of each link according to the importance of the page and the importance of the link, so as to optimize the shortest path between the key pages, thus playing a role in SEO optimization. The weight $w(u,v)$ of each edge can be calculated by the following formula:

$$w(u,v) = w_p(u) + w_l(u,v) + w_p(v) \quad (5)$$

Where, $w_p(u)$ and $w_p(v)$ are the weights of page u and v respectively, and $w_l(u,v)$ represents the link weights of page u to page v.

4.2. Identify and Resolve Isolated Pages in Your Website

Independent pages are pages that do not have sufficient internal link support, such pages are difficult to find effectively by search engines or do not have good SEO results. In the optimization process of the website, if many pages belong to independent pages, it may affect the overall weight distribution of the website, resulting in some important pages that cannot be completely found by search engines. This happens because the coherence between pages is not effectively managed and some pages do not have enough links or any link support.

In order to improve the adverse effects of independent pages, we can use Dijkstra algorithm to improve the relationship between pages and ensure that each page is connected with important pages through effective links. The specific approach is to use Dijkstra algorithm to calculate the shortest path between web pages, find independent pages, and give web pages to other web pages, so as to improve the effect of web SEO.

4.3. Dynamic Adjustment of Dijkstra Algorithm

Dijkstra's algorithm needs to adapt to changes in web page information and web page architecture. Every time the content of a web page changes, the adjacency and weight of a web page will also change to a certain extent. Therefore, Dijkstra's algorithm needs to be able to rearrange the path in time and always obtain the optimal path.

Dijkstra's algorithm was updated online by incremental update method. Only when the page or link of the website changes, the calculation needs to be performed, rather than the entire network map is repeatedly checked, in order to greatly improve the calculation speed, especially for large Internet.

Step 1: Identify change points: Constantly look at changes in the pages and structure of the site to find out where the site has changed.

Step 2: Perform incremental optimization: Update only those parts that have changed, do not need to re-establish the entire path between the network.

Step 3: Real-time update: Through the search engine, the weight of the web page and related paths are updated in a timely manner, thereby improving the accuracy of optimization (Figure 3).

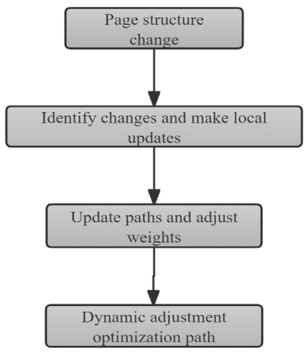
Page structure change
Identify changes and make local updates
Update paths and adjust weights
Dynamic adjustment optimization path

Figure 3. Dynamic Adjustment of the Incremental Update Flow of Dijkstra's Algorithm.

Therefore, we need to dynamically adjust Dijkstra's algorithm based on the changes of the network environment, which not only optimizes the response efficiency of the algorithm to web content, but also ensures the long-term good SEO effect of search engine optimization. At the same time, since the content and structure of web pages often change, the algorithm must be flexible and responsive. In this way, we can carry out weight calculation and link route re-calculation through local updates, achieve continuous improvement of SEO effect and avoid excessive system load due to large-scale changes.

4.4. Periodically Rerun Dijkstra's Algorithm

Since the content of web pages is constantly changing, and the link structure is also constantly changing, Dijkstra algorithm is implemented regularly to ensure that the connections between web pages are in the best state. Such regular updates can respond to changes in the network environment, ensuring that the algorithm can respond to changes in information or page structure in a timely manner and calculate the best path.

By running Dijkstra's algorithm periodically, we ensure that the network link structure is always maintained in its optimal position. We can set the interval at which the algorithm runs again, once a day, once a week or once a month. Update the shortest distance between all web pages in real time. When we run Dijkstra's algorithm, we have to recalculate all the web page shortest paths, which conforms to this formula (1).

5. SEO Effect Evaluation Based on Dijkstra Algorithm

5.1. Evaluation Criteria for SEO Effectiveness

The degree of influence of network internal chain optimization on network quality is an important index to judge whether an optimization scheme is feasible. Therefore, the key indicators to define and evaluate the degree of internal chain optimization are generally the following: the degree of page ranking optimization value increase; The extent to which clicks and traffic to the website have increased; The extent to which web crawling is accelerated; The increase in the proportion of users jumping out of the page and the time spent by users; Extent to which the site index coverage expands.

5.2. Design Experiments and Collect and Analyze Data

Through the experimental research and experimental design of the Dijkstra algorithm to optimize the internal chain of web pages, we can optimize the Internet search engine ranking (SEO) by its experimental research. The specific experimental design steps are as follows: randomly select a site with about 1000 web pages and different categories and content as the test object, and its internal chain structure is simple and not optimized. Then the whole process is divided into two steps, the first step as "as is" or "no optimization"; The second step is to optimize the internal chain structure state as Dijkstra algorithm.

1) Experiment Content:

Before optimization: Need to record the SEO related data of the site before optimization, including ranking data, traffic data, bounce rate data, per capita residence time data, index rate data, etc.

After optimization: Using Dijkstra algorithm to optimize and adjust the internal link structure of the website, adjust the page link weight and path, and use SEO tools to optimize and track the results, especially focusing on improving the weight of key pages of the website to improve its search ranking.

2) Data Collection:

We used Google to analyze traffic and bounce rates before and after optimization.

Use Google search results tool to extract web page index coverage and crawl frequency and other related data

Monitor the movement of your pages in search engines such as Google and Bing.

Track and record users' time on the web and user behavior, and judge the changes in user experience.

Data analysis: Through comparative analysis of different search engine optimization (SEO) parameters before and after the experiment, including the ranking position of web pages, visits, browsing time, bounce rate, etc., statistical methods were used to test the difference. Based on the above data results, the importance of Dijkstra algorithm in optimizing web pages and the efficiency of optimizing internal chain structure were analyzed (Table 2).

Table 2. The Change of Visits and the Improvement of Page Rank before and after.

Index	Before optimization	Post optimization	Variation
Page rank (Keyword 1)	15	5	Move up 10 places
Page rank (Keyword 2)	25	10	Up 15 places
Page views (times/day)	1500	2500	Boost 1000 times
Bounce rate (%)	50%	35%	15% reduction

Average stay time (min)	2.5	3.2	Increase by 0.7 minutes

After the adjustment, the page keyword ranking is significantly improved, indicating that the internal chain optimization has a positive impact on the search engine ranking. Improve the link structure, improve the support of key pages by internal links, and then bring more visitor traffic to the page. The optimized internal link structure strengthens the continuity of users browsing articles on the website and reduces the probability of users leaving the website. The changed page design improves the user experience and prolongs the user's stay.

6. Conclusion

This experimental paper focuses on the application of the Dijkstra algorithm to optimize the internal linking structure of websites and evaluates its impact on search engine optimization (SEO) performance. Through actual experimentation, the study verifies that implementing the algorithm can effectively enhance web page rankings, increase site traffic, and improve user satisfaction. Furthermore, it is recognized that in large-scale dynamic network environments, additional research will be necessary to further refine the algorithm, aiming to strengthen its optimization capabilities and improve its long-term stability in real-world applications.

References

1. M. Daruka and M. Damle, "Boosting D2C (Direct-to-Consumer) websites with search engine optimisation (SEO)," in *Proc. Somaiya Int. Conf. Technol. Inf. Manag. (SICTIM)*, Mumbai, India, 2023, pp. 84–89, doi: 10.1109/SICTIM56495.2023.10104723.
2. A. S. Kyzy and R. Ismailova, "Visibility of Moodle applications in Central Asia: Analysis of SEO," *Univ. Access Inf. Soc.*, vol. 23, pp. 493–503, 2024, doi: 10.1007/s10209-022-00923-6.
3. C. A. Kaya, R. Guler, M. C. Yavuz, et al., "Does fasting and high-fatty diet affect osseointegration: An experimental study," *Niger. J. Clin. Pract.*, vol. 28, no. 1, pp. 19–26, 2025, doi: 10.4103/njcp.njcp_835_23.
4. T. Rout, A. Mohapatra, M. Kar, et al., "Essential proteins in cancer networks: A graph-based perspective using Dijkstra's algorithm," *Netw. Model. Anal. Health Inform. Bioinform.*, vol. 13, p. 42, 2024, doi: 10.1007/s13721-024-00477-y.
5. S. M. Langbak, C. Schou, and K. D. Hansen, "Multi-autonomous mobile robot traffic management based on layered costmaps and a modified Dijkstra's algorithm," *Procedia Comput. Sci.*, vol. 232, pp. 53–63, 2024, doi: 10.1016/j.procs.2024.01.006.