# Article information:

Beautiful Branchless Binary Search | Probably Dance
<https://probablydance.com/2023/04/27/beautiful-branchless-binary-search/>

# Article summary:

1. The article discusses a branchless binary search algorithm called "Shar's algorithm" that was originally proposed by Leonard E Shar in 1971 and later described in John L Bentley's book from 1982.

2. The algorithm uses powers of two to jump through the array being searched, and is faster than std::lower\_bound in GCC, but may be slower for expensive comparison functions or when searching for strings.

3. The code for the algorithm is available on GitHub under the boost license, and the author has included a donation button for those who find open source work like this valuable.

# Article rating:

Appears moderately imbalanced: The article provides some useful information, but is missing several important points or pieces of evidence that would be required to present the discussed topics in a balanced and reliable way. You are encouraged to seek a more balanced perspective on the presented issues by exploring the provided research topics and looking at different information sources.

# Article analysis:

The article discusses a binary search algorithm called "Shar's algorithm" that is branchless and faster than std::lower\_bound in GCC. The author provides code for the algorithm and explains how it works, including its use of powers of two and its conditional move instruction. The article also compares the performance of branchless\_lower\_bound to std::lower\_bound in both GCC and Clang, as well as for searching strings.

The author notes that the comparison function provided by the user can impact the performance of branchless\_lower\_bound, but overall it is worth using if the comparison function is not too expensive. The article also tracks down the source of Shar's algorithm to John L Bentley's book from 1982, which was inspired by Knuth's "Sorting and Searching".

Overall, the article provides a detailed analysis of Shar's algorithm and its performance compared to std::lower\_bound. However, there are some potential biases in the article. For example, the author only tests the algorithm in GCC and Clang, so it is unclear how it performs in other compilers. Additionally, while the author acknowledges that std::lower\_bound may be faster if the comparison function is expensive, they do not provide any evidence or examples to support this claim.

Furthermore, while the article does mention some limitations of Shar's algorithm (such as its use of powers of two leading to more comparisons), it does not explore any potential risks or downsides beyond performance. It also does not present any counterarguments or alternative perspectives on whether or not branchless\_lower\_bound is truly superior to std::lower\_bound in all cases.

Overall, while informative and well-written, readers should approach this article with some caution and consider seeking out additional sources before making any decisions about which binary search algorithm to use.

# Topics for further research:

* Comparison of binary search algorithms in other compilers
* Impact of expensive comparison functions on binary search performance
* Risks and downsides of using branchless binary search algorithms
* Alternative perspectives on the superiority of branchless binary search algorithms
* Comparison of binary search algorithms for different data types (e.g. arrays vs. linked lists)
* Analysis of the impact of data distribution on binary search performance

# Report location:

<https://www.fullpicture.app/item/f29ad6c0fc30a32826655ec751ee33b4>