




mmh3: A Python extension for MurmurHash3

Hajime Senuma ¹

¹ University of Tokyo

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Vincent Knight](#) 

Reviewers:

- [@mrshu](#)
- [@JPenuchot](#)

Submitted: 22 May 2023

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

This decade has witnessed the rapid evolution of artificial intelligence (AI), notably in the field of natural language processing (NLP), as represented by the popularity of OpenAI's ChatGPT. Another important advancement in computer science and engineering is found in the field of the Internet of Things (IoT), a crucial component of ubiquitous computing, as represented by the development of Shodan, the world's first IoT search engine.

Underlying these advancements are high-performance algorithms and data structures that use non-cryptographic hash functions. This type of hash functions is generally characterized by four properties; they are fast, their resulting bits are statistically well-distributed, they have an avalanche effect, meaning a one-bit difference in a key changes at least half of the resulting bits, and they are collision resistant. Because cryptographic strength is not required in these use cases, they can leverage the efficiency of non-cryptographic hash functions.

MurmurHash3 and its test suite, SMHasher, was developed by Appleby (2011) and is one of the earliest and most continuously popular hash functions specifically designed to implement the characteristics mentioned above.

mmh3 was launched in 2011 as a Python wrapper for MurmurHash3 and has been maintained ever since. Its API is very simple to use for Python programmers, as it offers both simple one-shot hash functions and hasher classes that allow incremental updating, whose methods are compliant to `hashlib`, a part of the Python Standard Library. The library provides Python wheels (i.e., pre-built binary packages) for immediate use on various platforms, including Linux (x86_64, aarch64, i686, ppc64le, and s390x), Windows (win32, win_amd64, and win_arm64), and macOS (Intel Mac and Apple Silicon). From version 4.0.0, mmh3 has been published under the MIT License, an OSI-approved permissive open-source license.

As of June 1, 2023, mmh3 was being downloaded more than 2 million times per month, and it ranks as the 970th most downloaded PyPI package (of around 458,000 projects), showing that only 0.22% of the remaining packages in the PyPI ecosystem are more popular (Van Kemenade et al., 2023). According to PePy, as of June 2, 2023, the total downloads of this library exceeded 79 millions.

Libraries and organizations that directly use mmh3 include Apache Iceberg (an open table format for analytic datasets), PyMilvus (a Python SDK for Milvus, an open-source vector database), Shodan (the world's first IoT search engine), and pocsuite3 (open-source remote vulnerability testing framework). Those that contain mmh3 as an indirect dependency include the ChatGPT Retrieval Plugin by OpenAI.

Statement of need

AI and High-Performance Computing

AI is one of the most resource-demanding fields in computer science and engineering. To mitigate this problem, various techniques are employed under main systems, in which non-cryptographic hash functions play key roles in a number of algorithms and data structures.

A notable technique is *feature hashing* (Shi et al., 2009; Weinberger et al., 2009). In its simplest usage, when given a string-indexed data vector, it simply converts the vector into an integer-indexed data vector in which each index is the hash result of the original string index; collision values are simply summed. Despite its simple and intuitive usage, a machine-learning process with feature hashing is statistically guaranteed to be nearly as accurate as its original process. Feature hashing has been shown to be useful for various situations, including K-means clustering (Senuma, 2011) and succinct model learning (Senuma & Aizawa, 2016).

Other popular techniques that leverage non-cryptographic hash functions include *Bloom Filter* (Bloom, 1970), a compact data structure that tests whether an element is a member of a certain set (with false positive matches), and *MinHash* (Broder, 1997), an algorithm that quickly estimates the similarity of two sets.

mmh3 has appears in various scholarly papers, including a study of Indian language NLP suites (Kakwani et al., 2020) and another about a secure system based on probabilistic structures (Adja et al., 2021). It has also appeared in technical books and computer science texts (Gorelick & Ozsvald, 2020; Kumar & Miglani, 2021; Medjedovic et al., 2022).

Internet of Things

mmh3 is applicable to the IoT field. Shodan uses mmh3 as its fingerprint for a favicon (i.e., an icon associated with a web page or website) (Shodan, 2021). Shodan (2020) explained they adopted mmh3 because it was space efficient and has scalable performance. ZoomEye, another popular IoT search engine, follows Shodan's convention.

As a result, mmh3 is considered a useful tool for cybersecurity. For example, Kopriva (2021) reported a method of discovering possible phishing websites by searching websites with Shodan, whose favicon's mmh3 hash value was the same as that of a genuine and trustable one.

Another use case of mmh3 in this area includes open-source intelligence (OSINT) activities, such as measuring the global popularity of web frameworks and servers, as some users do not change their default favicon settings specified by applications. Faraday Security (2022) described a method of using mmh3 and Shodan to approximate the popularity of Spring, a Java-based web framework.

Related software

PYMMH (Kihlander & Gusani, 2013) is a pure Python implementation of the MurmurHash3 algorithm. Among various other Python bindings for non-cryptographic hashes, xxhash by Yue Du (Du, 2014) is another popular hash library, featuring xxHash developed by Yan Collet (Collet, 2012).

Acknowledgements

The author expresses his deep gratitude to Professor Akiko Aizawa for her helpful comments on this paper. He also appreciates the contributions of individuals to the development and maintenance of mmh3. Special thanks go to Dr. Micha Gorelick, who made the first pull request to the project and later introduced the library in her technical book (Gorelick & Ozsvald, 2020).

References

- Adja, Y. C. E., Hammi, B., Serhrouchni, A., & Zeadally, S. (2021). A blockchain-based certificate revocation management and status verification system. *Computers & Security*, 104, 102209. <https://doi.org/10.1016/j.cose.2021.102209>
- Appleby, A. (2011). *MurmurHash3 and SMHasher*. <https://github.com/aappleby/smhasher>
- Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7), 422–426. <https://doi.org/10.1145/362686.362692>
- Broder, A. Z. (1997). On the resemblance and containment of documents. *Proceedings. Compression and Complexity of SEQUENCES 1997*, 21–29. <https://doi.org/10.1109/SEQUEN.1997.666900>
- Collet, Y. (2012). *xxHash*. <https://github.com/Cyan4973/xxHash>
- Du, Y. (2014). *xxhash*. <https://github.com/ifduyue/python-xxhash>
- Faraday Security. (2022). *Understanding Spring4Shell*. <https://faradaysec.com/understanding-spring4shell/>
- Gorelick, M., & Ozsvald, I. (2020). *High Performance Python: Practical Performant Programming for Humans* (2nd edition). O'Reilly Media. ISBN: 978-1-4920-5502-0
- Kakwani, D., Kunchukuttan, A., Golla, S., N. C., G., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020). IndicNLP Suite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4948–4961. <https://doi.org/10.18653/v1/2020.findings-emnlp.445>
- Kihlander, F., & Gusani, S. (2013). *PYMMH3*. <https://github.com/wc-duck/pymmh3>
- Kopriva, J. (2021). *Hunting phishing websites with favicon hashes*. SANS Internet Storm Center. <https://isc.sans.edu/diary/27326>
- Kumar, N., & Miglani, A. (2021). *Probabilistic Data Structures for Blockchain-Based Internet of Things Applications*. CRC Press. <https://doi.org/10.1201/9781003080046>
- Medjedovic, D., Tahirovic, E., & Dedovic, I. (2022). *Algorithms and Data Structures for Massive Datasets*. Manning. ISBN: 978-1-61729-803-5
- Senuma, H. (2011). K-means Clustering with Feature Hashing. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Student Session*, 122–126.
- Senuma, H., & Aizawa, A. (2016). Learning Succinct Models: Pipelined Compression with L1-Regularization, Hashing, Elias–Fano Indices, and Quantization. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2774–2784.
- Shi, Q., Petterson, J., Dror, G., Langford, J., Smola, A., & Vishwanathan, S. V. N. (2009). Hash Kernels for Structured Data. *Journal of Machine Learning Research*, 10, 2615–2637.
- Shodan. (2020). "We created this technique and decided on mmh3 due to space efficiency and performance at scale.". In *Twitter*. <https://twitter.com/shodanhq/status/1314023874128998400>
- Shodan. (2021). "It's the MMH3 hash of the http.html property. See: PyPI mmh3". In *Twitter*. <https://twitter.com/shodanhq/status/1395501365456261122>
- Van Kemenade, H., Si, R., & Dollenstein, Z. (2023). *Hugovk/top-pypi-packages: Release 2023.06* (Version 2023.06). Zenodo. <https://doi.org/10.5281/zenodo.7994944>

- ¹²³ Weinberger, K., Dasgupta, A., Langford, J., Smola, A., & Attenberg, J. (2009). Feature Hash-
¹²⁴ ing for Large Scale Multitask Learning. *Proceedings of the 26th International Conference*
¹²⁵ *on Machine Learning*. <https://doi.org/10.1145/1553374.1553516>

DRAFT