3/16/2025

Jason A. Rochau

CSC506-1

Personalized Content Recommendation System: A Hash Table Approach

This report outlines a personalized content recommendation system that utilizes hash tables to create an association between user IDs and their interests. The program has been specifically designed to recommend niche content topics including Downhill Mountain Biking, Fly Fishing, Crane Operation, and Skateboarding. When thinking about digital platforms, this approach becomes particularly relevant as it enables rapid, real-time content updates and user-specific suggestions which are vital to maintaining user engagement (Adomavicius & Tuzhilin, 2001).

The primary objective of this program is to efficiently manage and retrieve user preferences to deliver personalized content recommendations. I would implement hash tables for their average-case constant time (O(1)) complexity for both insertion and retrieval operations, making them highly suitable for handling large datasets with minimal performance degradation. The design structure encapsulates user preferences within a hash table, allowing for quick updates and recommendations when needed. This code implementation logically mirrors the real-life process of content personalization, where each user's unique interests are stored and later retrieved to tailor the content stream to their specific preferences (Fountoulakis et al., 2010).

The algorithm leverages hash tables to achieve average-case O(1) performance for data retrieval and insertion operations. In the worst-case scenario—when multiple keys hash to the same bucket—the complexity can degrade to O(n), though with careful design considerations and an efficiently hash function implementation, we can help mitigate this risk significantly (Brehm, 2019). When examining real-life factors, such as hash collisions and memory management, we can see how they influence performance, especially as the user base grows exponentially. External factors like network latency and concurrent access also impose practical limits on system responsiveness, making scalability and efficiently caching strategies critical components for successful implementation (Kraska et al., 2018).

Working on this project wasn't without its challenges. I struggled to design a reliable hash function and find an effective way to manage collisions, which really pushed me to explore data structure optimization and delve into algorithm analysis in a hands-on manner. These hurdles helped me improve my Python skills and taught me how to analyze system performance under real-world conditions. In short, I came away from the project with a richer understanding of how to build scalable, real-time systems—a skill set I’m excited to apply in future work.

This project clearly illustrates why hash tables are so powerful for personalized content recommendations. The ability to access user preferences almost instantly means the system can deliver timely, relevant recommendations that enhance the user experience. By tackling performance issues and external challenges head-on, I built a solid foundation for a dynamic, user-focused content delivery system that’s ready to adapt as needs evolve.

References:

Adomavicius, G., & Tuzhilin, A. (2001). Expert-Driven Validation of Rule-Based User Models in Personalization Applications. Data Mining and Knowledge Discovery, 5(1–2), 33.

Fountoulakis, N., Panagiotou, K., & Steger, A. (2010). On the Insertion Time of Cuckoo Hashing.

Brehm, W. (2019). Hash tables with pseudorandom global order. ResearchGate. https://www.researchgate.net/publication/334646498\_Hash\_Tables\_with\_Pseudorandom\_Global\_Order

Kraska, T., Beutel, A., Chi, E. H., Dean, J., & Polyzotis, N. (2018). A tutorial on learned multi-dimensional indexes. Proceedings of the ACM SIGMOD International Conference on Management of Data, 2018, 1325–1339. https://doi.org/10.1145/3183713.3196935