Jamie Kemman

Prof. Jason Miller

CIS-211-02

17 April 2024

Hash Tables: a Brief History and Overview

According to D. and N. Malholtra, authors of *Data Structures and Program Design Using Java*, "A hash table is a data structure which supports one of the efficient searching techniques . . . hashing" (355). Though structurally similar to the standard array, a hash table (and the associated object class HashMap within the Java utilities library) allows, by design, for consistent search and retrieval of specific values in O(1) time (D. Malholtra and N. Malholtra, 355). Compared to the O(n) of a sequential search or the O(log(n)) of a binary search within a similar structure, this is a vast improvement in search time when considered against the scope of large data sets. Though the boons presented by a hash table are readily apparent in today's world of unfathomably large databases, in the 1950s this methodology was groundbreaking. In a time when computers took up entire rooms and required teams of programmers and operators to function, limited system resources and high costs for computer time meant increases in operation efficiency presented significant savings. Although credit for the invention of hashing is primarily split between two groups at IBM who simultaneously developed its core mechanics (Knuth 3: 547), Hans Peter Luhn stands as one of the core figures.

Luhn, by account of Hallam Stevens in an article for IEEE Specturm, was a veritable polymath and inventor who held countless patents for machines and devices designed to aid information storage and retrieval in both niche scientific fields and across general business concerns (Stevens). Luhn is credited with writing the algorithm commonly known as the modulus 10 algorithm, which produces a checksum digit within the result when given a 10-digit number as input (Stevens). This algorithm can be used to check the validity of important numbers, and is also an application of hashing wherein a standardized data input is given to a corresponding algorithm in order to obtain a shortened result

which is still unique to the original input (Stevens). Although obtaining a single digit checksum has more to do with data integrity than with information storage, the modulus 10 algorithm is still an example of a hashing algorithm, Besides providing a namesake, hashing is critical to any hash table. And as for hash tables, Luhn is additionally credited with the suggestion of using hashed data to speed up information retrieval via assigning multiple unique data to a single “bucket”, or index in the case of an array (Stevens). With this foundaton in place, later computer scientists built, directly and indirectly, on Luhn’s work to form the prevalent conception and practice of hash mapping as it is today.

The process of hash mapping is perhaps aptly described by the term “key transformation” which was the prevalent descriptor in print literature before the 1970s (Knuth 3: 548). In a hash map, given data is transformed into a key, and with that key the data is assigned, or mapped, to an index. When searching for specific data within the map, the data itself can then serve as a key when combined with the correct algorithm. To extrapolate a real-world use case, email addresses, employee ID numbers, or even names could easily be used to quickly reference detailed employee files within a large company’s database. Given an applicable algorithm, the raw data of every employee email address could be transformed into a pointer for precise storage locations. Even in the intended or unintended event that the algorithm produces an identical key for multiple unique inputs, referred to as producing a “collision”, buckets can be designed to hold multiple employee records (Knuth 3: 542). Searching a bucket containing 10 employee records for a single entry will take far less time than searching a database of 100,000 employee records.

In Java, a hash map can be implemented in code via the HashMap class, and is invoked using HashMap<K,V>, where K is equal to the data type or Object of the keys and V is equal to the data type or object of the mapped values (Oracle). HashMaps in Java are created with an initial capacity denoted in available buckets. A variable load factor is also created with each HashMap, and this information is used to control automatic expansion of the HashMap as data is added and buckets are filled.

Works Cited

"Class HashMap<K,V>" *Java Platform SE 8*, Oracle, docs.oracle.com/javase/8/docs/api/java/util/HashMap.html. Accessed 17 Apr. 2024,

Knuth, Donald. *The Art of Computer Programming. 2nd ed., v*ol. 3, Addison-Wesley, 1998.

Malhotra, D., and N. Malhotra. *Data Structures and Program Design Using Java : A Self-Teaching Introduction*. Mercury Learning & Information, 2020. *EBSCOhost*, search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2377795&site=ehost-live&scope=site.

Stevens, Hallam. "Hans Peter Luhn and the Birth of the Hashing Algorithm." *IEEE Spectrum*, IEEE Spectrum, 29 July 2021, *spectrum.ieee.org/hans-peter-luhn-and-the-birth-of-the-hashing-algorithm.*