CSE 3010 – Data Structures & Algorithms

Lecture #29-#30-#31

What will be covered today

- Introduction to hashing
- Example to understand hashing
- Collision resolution methods

Hashing

- Search technique that does two things
  - Stores the keys in a hash table based on a function called hash function
  - Retrieves an item from the list based on the same hash function to store the items of the list
- Hash function
  - Converts the key into an integer, where $0 \leq \text{integer} < N$
  - Returns a value between 0 and N-1
  - Returned value is the index at which the item will be stored in the list
- Hash table
  - Is the array in which the items are stored
- Collision
  - Situation when a newly inserted key maps to an already occupied slot in the hash table

Hash-based searching

- Linear search works well on a small number of items
- Binary and Jump search work well on an ordered collection of items
- Hash-based searching works well for large unordered collection of items
- Hash-based searching uses a hash function and hash table
- Main objectives of a good hash function
  - Distribute keys uniformly over the N slots
  - Minimize collisions
  - Not leave many unused slots

How does hashing work

1. List `L` contains `N` keys
2. `N` keys are stored in hash table `H`
3. Hash table `H` has `N` slots
4. Hash function `hash(key)` returns `h`, where $0 \leq h < N$
5. Key inserted at `H[h]`
6. Key retrieved from `H[h]`

Some hash functions

- Transforms the hash field value into an integer between 0 and N-1
- Hashing techniques
  - Mod function

    `h(Key) = Key mod N`
  - Folding

    `h(Key) = (P1 of Key + P2 of Key) mod N`

    where `P1` and `P2` are parts of the `Key`
  - Picking random digits

    `h(Key) = (Sum of every even digit of Key) mod N`

Understanding hashing using an example – List of Word and its synonyms

Word and Synonyms			
Word	Synonym 1	Synonym 2	Synonym 3
Color	Glow	Hue	
Blanket	Absolute	Overall	
Origin	Source	Root	Ancestor
Family	Clan	Group	Tribe
Extract	Juice	Abstract	
Lovely	Handsome	Pretty	
Zero	Nil	Naught	Void
Asteroid	Earth	World	
Desk	Counter	Table	Lectern
Break	Crack	Gap	
…			

65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z

Understanding hashing using an example

Hash Key Option 1 [Length of Word]
5
7
6
6
7
6
4
8
4
5

Hash Key Option 2 [ASCII code of 1st character of Word]
67
66
79
70
69
76
90
65
68
66

Hash Key Option 2 [Sum of digits of the ASCII digits]
67 = 13
66 = 12
79 = 16
70 = 7
69 = 15
76 = 13
90 = 9
65 = 11
68 = 14
66 = 12

Understanding collision

- Hashing function do not guarantee distinct address for every key
  - Hash field space is almost always larger than the address space
- ***Collision*** occurs when `h(Key)` hashes to an already-occupied address
- Finding another position to place the `h(Key)` is called ***collision resolution***

Collision resolution methods

Method	Description
Open addressing	When a collision occurs: • Check subsequent positions in order until an unused slot is found
Chaining [Also referred as Separate Chaining]	• Maintain a linked list of nodes in each slot of the hash table • When a collision occurs: • Add new nodes to linked list pointed to by the hash(key)
Multiple hashing	When a collision occurs: • Apply a second hash function • If second hash function results in collision, apply either open addressing or another hash function

Collision resolution methods - Comparison

Open Addressing	Chaining
No additional data structure is needed as all records are stored in the same hash table	Additional data structure required to resolve collision
Unique hash key is required when collision occurs	Key need not be unique as unused overflow slots are used
Difficult to determine size of hash table enough to store all records	Pre-determined number of overflow slots are added to the array
Slower performance as the hash function may be called multiple times when a collision occurs	Faster performance as there is no need to call the hash function multiple times when a collision occurs