CSCI 335
Software Design and Analysis III
Lecture 11 Part 2: Hashing

Professor Anita Raja

10-06-22

1

1

Agenda

- Hash tables
- Hash tables without Linked Lists
  - Linear Probing
  - Quadratic Probing
- Hash tables with Linked Lists
- Separate Chaining

2

2

Hash Applications

- Symbol tables (compilers)
- Graphs where nodes are strings (e.g. names of cities)
- Spell-checkers
- Password checking
- • …

3

3

Hashing

- Several methods of implementing hash table.
- Compare these methods analytically.
- Show numerous applications of hashing.
- Compare hash tables with binary trees.

4

4

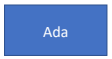
Hashing

- Hash Table ADT
  - An array of some fixed size containing the items.
  - Implementation is called hashing.
  - Used for insertions, deletions, and finds in constant average time.
  - Ordering operations are not supported efficiently.
    - findMin, findMax and printing entire table in linear time not supported.

5

5

- Find Ada: Linear search
- Find Ada : When index known
  - myData = Array(8)



Jan	Tim	Mia	Sam	Leo	Ted	Bea	Lou	Ada	Max	Zoe
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

6

6

Mia	M	77	i	105	a	97	279	4

0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

7

7

Mia	M	77	i	105	a	97	279	4

0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

8

8

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1

Mia										
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 9

9

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1

Tim		Mia								
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 10

10

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1

Bea
Zoe
Jan
Ada
Leo
Sam
Lou
Max
Ted

https://www.asciitable.xyz/

Tim				Mia						
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 11

11

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1
Bea	B	66	e	101	a	97	264	0
Zoe	Z	90	o	111	e	101	302	5
Jan	J	74	a	97	n	110	281	6
Ada	A	65	d	100	a	97	262	9
Leo	L	76	e	101	o	111	288	2
Sam	S	83	a	97	m	109	289	3
Lou	L	76	o	111	u	117	304	7
Max	M	77	a	97	x	120	294	8
Ted	T	84	e	101	d	100	285	10

Tim				Mia						
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 12 12

12

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1
Bea	B	66	e	101	a	97	264	0
Zoe	Z	90	o	111	e	101	302	5
Jan	J	74	a	97	n	110	281	6
Ada	A	65	d	100	a	97	262	9
Leo	L	76	e	101	o	111	288	2
Sam	S	83	a	97	m	109	289	3
Lou	L	76	o	111	u	117	304	7
Max	M	77	a	97	x	120	294	8
Ted	T	84	e	101	d	100	285	10

Bea	Tim	Leo	Sam	Mia	Zoe	Jan	Lou	Max	Ada	Ted
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

13

13

Index number = sum_ASCII_codes Mod array_size

Bea	Tim	Leo	Sam	Mia	Zoe	Jan	Lou	Max	Ada	Ted
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

14

14

Find Ada 262 Mod 11 = 9

myData = Array(9)

Ada

Bea	Tim	Leo	Sam	Mia	Zoe	Jan	Lou	Max	Ada	Ted
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series

15

15

Hashing

- Data is stored in a table (array/vector).
  - Let us call size of table T and number of elements stored in table N.
  - Load factor λ = N / T (helps to analyze performance).
- Ideally O(1) find/insert/delete.
  - But no ability for sorting-based features.
- Hash function gives index in constant time:
  - Ideally simple to compute and ensure 2 different keys get different cells.
  - In practice, should distribute keys evenly among cells
  - Hash Function: Key -> {0, 1, 2, …, T-1}.
  - Key: Integer/String/etc. – depending on application.
  - Mapping from keys to indices is not unique =>
    - Collision resolution is required.

16

16

Hashing Algorithm

- Calculation applied to a key to transform it into an address.
- For numeric keys, divide the key by the number of available addresses, n, and take the remainder
  address = key Mod n
- For alphanumeric keys, divide the sum of ASCII codes in a key by the number of available addresses, n and take their remainder.
- Folding method divides key into equal parts then add the parts together
  - Phone number 1-212-650-5031 becomes 01+21+26+ 50+50+31=179
  - Can divide by some constant and take the remainder

17

17

Collision Resolution

- If when an element is inserted, it hashes to the same value as an already inserted element
- Collision needs to be resolved.
- Simple methods:
  - Open addressing
  - Closed addressing
- Alternatives:
  - Cuckoo Hashing
  - Hopskotch Hashing

18

18

Collisions

- Open addressing
  - Linear probing
  - Plus 3 rehash
  - Quadratic probing (failed attempts)^2
  - Double hashing
- Closed addressing
  - Separate chaining

19

19

Open Addressing with Linear probing

20

20

Open Addressing with Linear probing

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1
Bea	B	66	e	101	a	97	264	0
Zoe	Z	90	o	111	e	101	302	5
Sue	S	83	u	117	e	101	301	4

Bea	Tim			Mia	Zoe					
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 21

21

Open Addressing with Linear probing

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1
Bea	B	66	e	101	a	97	264	0
Zoe	Z	90	o	111	e	101	302	5
Sue	S	83	u	117	e	101	301	4
Len	L	76	e	101	n	110	287	1
Moe	M	83	o	111	e	101	289	3
Lou	L	76	o	111	u	117	304	7
Rae	R	82	a	97	e	101	280	5
Max	M	77	a	97	x	120	294	8
Tod	T	84	o	111	d	100	295	9

Bea	Tim			Mia	Zoe	Sue				
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 22

22

Open Addressing with Linear probing

Mia	M	77	i	105	a	97	279	4
Tim	T	84	i	105	m	109	298	1
Bea	B	66	e	101	a	97	264	0
Zoe	Z	90	o	111	e	101	302	5
Sue	S	83	u	117	e	101	301	4
Len	L	76	e	101	n	110	287	1
Moe	M	83	o	111	e	101	289	3
Lou	L	76	o	111	u	117	304	7
Rae	R	82	a	97	e	101	280	5
Max	M	77	a	97	x	120	294	8
Tod	T	84	o	111	d	100	295	9

Bea	Tim	Len	Moe	Mia	Zoe	Sue	Lou	Rae	Max	Tod
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 23

23

Open Addressing with Linear probing

Find Rae 280 Mod 11 = 5

myData = Array(5)

Rae

Load Factor = $\frac{total\ \#\ of\ items}{size\ of\ array}$

Bea	Tim	Len	Moe	Mia	Zoe	Sue	Lou	Rae	Max	Tod
0	1	2	3	4	5	6	7	8	9	10

Acknowledgement for example: Kevin Drumm, Computer Science series 24

24

Summary

- More about other collision resolution methods next lecture
- Used to index large amounts of data
- Address of each key calculated using the key itself.
- Collisions resolved when open or closed addressing
- Hashing is widely used in database indexing, compilers, caching, password authentication and more
- Insertion, deletion and retrieval occur in constant time.

25