Hash and Collision

Kuan-Yu Chen (陳冠宇)

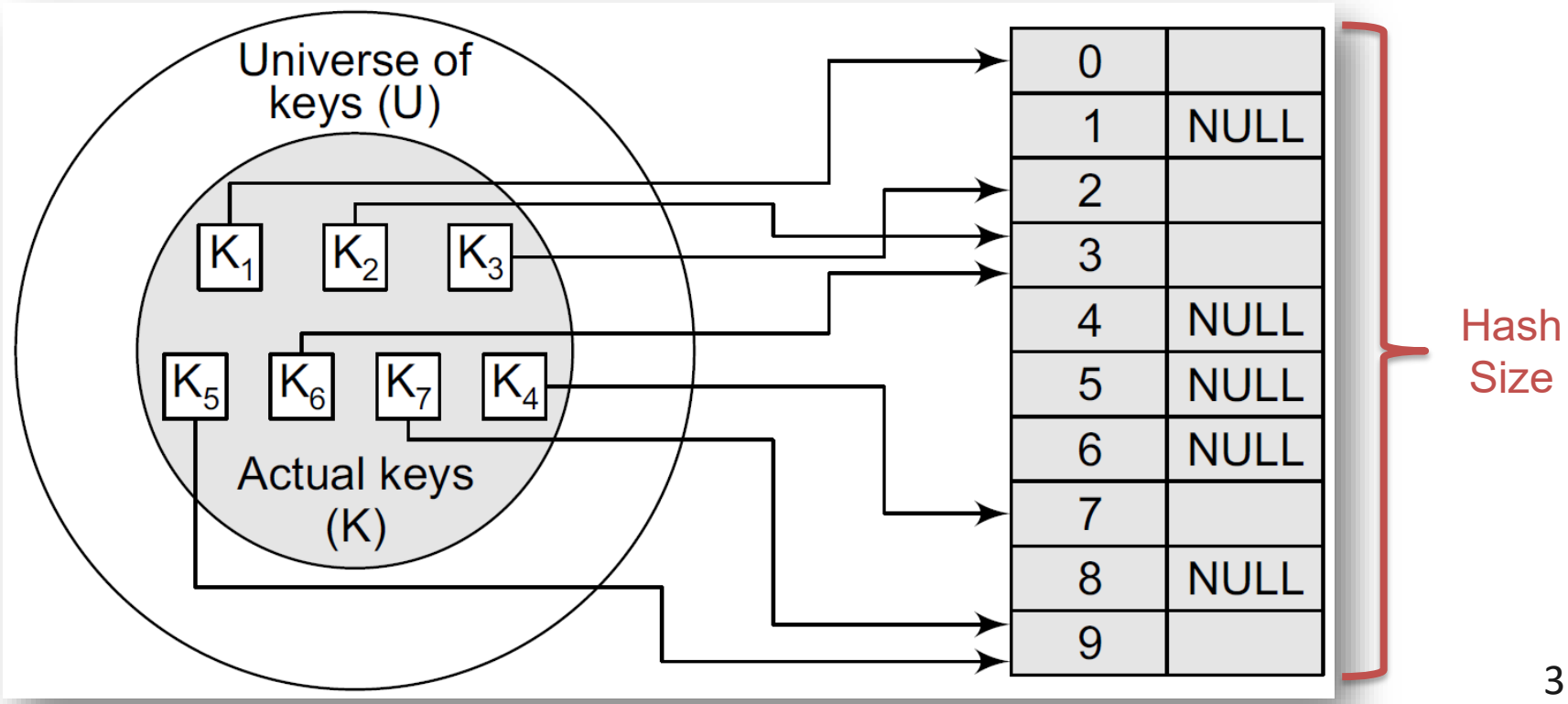
Review

- Graph is an important data structure

	Undirected Graph	Directed Graph
Definition	$G = (V, E)$	
Node Degree	$\deg(u)$	$\deg(u) = indeg(u) + outdeg(u)$
Representation	Sequential Representation Linked Representation Adjacency Multi-list	
Search	Breadth-first Search (Queue) Depth-first Search (Stack)	
Minimal Spanning Tree	Prim's Algorithm Kruskal's Algorithm	
Shortest Path		Dijkstra's Algorithm Bellman-ford Algorithm

Hashing

- Hash table is a data structure in which keys are mapped to array positions by a hash function
  - An element with key k is stored at index $h(k)$
    - It means a hash function h is used to calculate the index at which the element with key k will be stored



Hash Functions

- A hash function is a **mathematical formula** which, when applied to a key, produces an integer which can be used as an index for the key in the hash table
  - Properties of a good hash function
    - ***Low cost***

      The cost of executing a hash function must be small
    - ***Determinism***

      A hash procedure must be deterministic
    - ***Uniformity***

      A good hash function must map the keys as evenly as possible over its output range

      » This means that the probability of generating every hash value in the output range should roughly be the same

Representative Hash Functions.

- Division Method
  - It is the most simple method of hashing an integer x

$$h(x) = x \bmod M$$

    - Since it requires only a single division operation, the method works very fast
    - Extra care should be taken to select a suitable value for M

      It is usually to choose M to be a prime number

Example

- Calculate the hash values of keys 1234 and 5462 by referring to $M = 97$

$$h(1234) = 1234\%97 = 70$$

$$h(5642) = 5642\%97 = 16$$

Representative Hash Functions..

- Multiplication Method
  - The method is defined by

$$h(x) = \lfloor m(xA \bmod 1) \rfloor$$

    - Choose a constant A such that $0 < A < 1$
    - Multiply the key x by A
    - Extract the fractional part of xA
    - Multiply the fractional part of xA by the size of hash table m

  - The greatest advantage of this method is that it works practically with any value of A
    - Generally, a good choice of A is $\frac{\sqrt{5}-1}{2} = 0.618033$

Example

- Given a hash table of size 1000, map the key 12345 to an appropriate location in the hash table

$$A = 0.618033$$

$$m = 1000$$

$$\begin{aligned} h(12345) &= \lfloor m(12345 \times A \bmod 1) \rfloor \\ &= \lfloor 1000 \times (12345 \times 0.618033 \bmod 1) \rfloor \\ &= \lfloor 1000 \times (7629.617385 \bmod 1) \rfloor \\ &= \lfloor 1000 \times (0.617385) \rfloor \\ &= \lfloor 617.385 \rfloor = 617 \end{aligned}$$

Representative Hash Functions...

- Mid-Square Method
  - The algorithm works well because most or all digits of the key value contribute to the result
  - In general, the function is defined by:

$$h(x) = r - digit(x^2) = s$$

where s is obtained by selecting r digits from x^2

- Square the value of the key
- Extract the middle r digits of the result
- r depends on the hash size

Example

- Suppose a hash table has 100 memory locations, please calculate the hash value for keys 1234 and 5642 using the mid-square method
  - Note that the hash table has 100 memory locations whose indices vary from 0 to 99
    - This means that only two digits are needed to map the key to a location in the hash table, so $r = 2$

$$x = 1234, \qquad x^2 = 1522756, \qquad h(1234) = 27$$

$$x = 5642, \qquad x^2 = 31832164, \qquad h(5642) = 21$$

Representative Hash Functions….

- Folding Method

$$h(x) = r - digit\left(sum\big(divide(x)\big)\right) = s$$

  - The folding method works in the following two steps
    - Divide the key value into a number of parts
    - Add the individual parts, and extract the last r digits of the result

  - Note that the number of digits in each part of the key will vary depending upon the size of the hash table

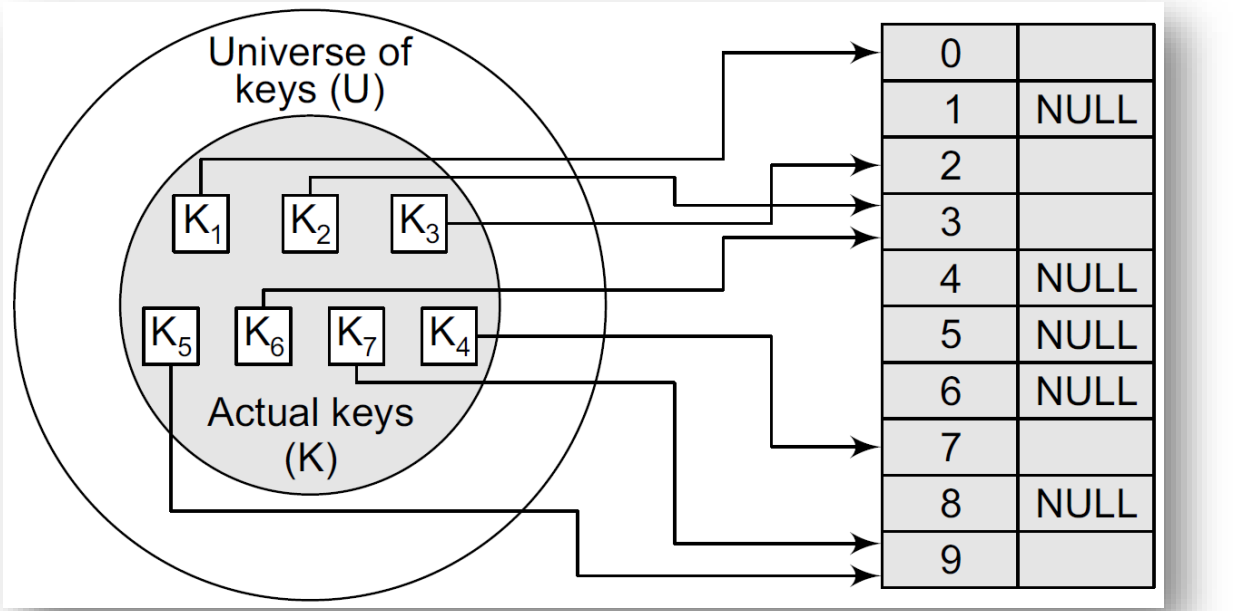
Example

- Given a hash table of 100 locations, calculate the hash value using folding method for keys 5678, 321, and 34567
  - Since there are 100 memory locations to address, we will break the key into parts where each part (except the last) will contain two digits

key	5678	321	34567
Parts	56 and 78	32 and 1	34, 56 and 7
Sum	134	33	97
Hash value	34	33	97

Collision

- When two or more keys map to the same memory location, a **collision** is said to occur
  - k_2 and k_6
  - k_5 and k_7



- A method used to solve the problem of collision, also called **collision resolution technique**, is applied
  - Open addressing
  - Chaining

Open Addressing

- By using the technique, the hash table contains two types of values: **sentinel values** (e.g., –1) and **data values**
  - The sentinel value indicates that the location contains no data value at present but can be used to hold a value

- If the location already has some data value stored in it, then other slots are examined systematically in the forward direction to find a free slot
  - If even a single free location is not found, then we have an OVERFLOW condition

- The process of examining memory locations in the hash table is called **probing**
  - linear probing, quadratic probing, double hashing, and rehashing

Linear Probing

- The simplest approach to resolve a collision is linear probing
  - An extension of the division method

- If a value is already stored at a location generated by $h'(x)$, then the following hash function is used to resolve the collision

$$h(x, i) = [h'(x) + i] \bmod M$$

  - M is the size of the hash table, $h'(x) = x \bmod M$, and i is the **probe number** that varies from 0 to $M-1$

  - When we have to store a value, we try the slots: $[h'(x)] \bmod M$, $[h'(x) + 1] \bmod M$, $[h'(x) + 2] \bmod M$, $[h'(x) + 3] \bmod M$, and so no, until a vacant location is found

$h(x,i) = [h'(x) + i] \bmod M$

Example.

- Consider a hash table of size 10, please use linear probing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table
  - Initial

0	1	2	3	4	5	6	7	8	9
–1	–1	–1	–1	–1	–1	–1	–1	–1	–1

  - Step 1
    - $x = 72$
    - $h(72,0) = [h'(72) + 0] \bmod 10$
      $= [(72 \bmod 10) + 0] \bmod 10 = 2$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	–1	–1	–1	–1

$h(x,i) = [h'(x) + i] \bmod M$

Example..

- Consider a hash table of size 10, please use linear probing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	–1	–1	–1	–1

  - Step 2
    - $x = 27$
    - $h(27,0) = [h'(27) + 0] \bmod 10$
      $= [(27 \bmod 10) + 0] \bmod 10 = 7$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	–1	27	–1	–1

  - Step 3
    - $x = 36$
    - $h(36,0) = [h'(36) + 0] \bmod 10$
      $= [(36 \bmod 10) + 0] \bmod 10 = 6$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	36	27	–1	–1

$h(x,i) = [h'(x) + i] \bmod M$

Example…

- Consider a hash table of size 10, please use linear probing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	36	27	–1	–1

  - Step 4
    - $x = 24$
    - $h(24,0) = [h'(24) + 0] \bmod 10$
      $= [(24 \bmod 10) + 0] \bmod 10 = 4$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	24	–1	36	27	–1	–1

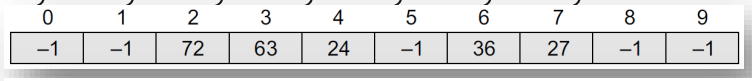
  - Step 5
    - $x = 63$
    - $h(63,0) = [h'(63) + 0] \bmod 10$
      $= [(63 \bmod 10) + 0] \bmod 10 = 3$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	63	24	–1	36	27	–1	–1

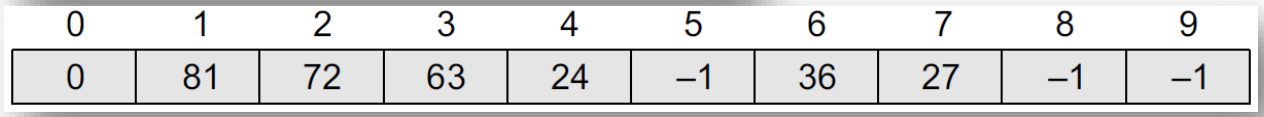
$h(x,i) = [h'(x) + i] \bmod M$

Example….

- Consider a hash table of size 10, please use linear probing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table
  - Step 6

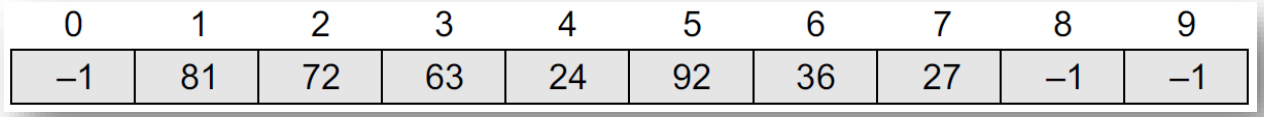


0	1	2	3	4	5	6	7	8	9
–1	–1	72	63	24	–1	36	27	–1	–1



0	1	2	3	4	5	6	7	8	9
0	81	72	63	24	–1	36	27	–1	–1

    - $x = 81$
    - $h(81,0) = [h'(81) + 0] \bmod 10$
      $= [(81 \bmod 10) + 0] \bmod 10 = 1$
  - Step 7



0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	92	36	27	–1	–1

    - $x = 92$
    - $h(92,0) = [h'(92) + 0] \bmod 10$
      $= [(92 \bmod 10) + 0] \bmod 10 = 2$
    - $h(92,1) = [h'(92) + 1] \bmod 10$
      $= [(92 \bmod 10) + 1] \bmod 10 = 3$
    - $h(92,2) = [h'(92) + 2] \bmod 10$
      $= [(92 \bmod 10) + 2] \bmod 10 = 4$
    - $h(92,3) = [h'(92) + 3] \bmod 10$
      $= [(92 \bmod 10) + 3] \bmod 10 = 5$

$h(x,i) = [h'(x) + i] \bmod M$

Example.....

- Consider a hash table of size 10, please use linear probing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	92	36	27	–1	–1

  - Step 8
    - $x = 101$
    - $h(101,0) = [h'(101) + 0] \bmod 10$
      $= [(101 \bmod 10) + 0] \bmod 10 = 1$
    - $h(101,1) = [h'(101) + 1] \bmod 10$
      $= [(101 \bmod 10) + 1] \bmod 10 = 2$
    - ...
    - $h(101,7) = [h'(101) + 7] \bmod 10$
      $= [(101 \bmod 10) + 7] \bmod 10 = 8$

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	92	36	27	101	–1

Quadratic Probing

- If a value is already stored at a location generated by $h'(x)$, then the following hash function is used to resolve the collision

$$h(x, i) = [h'(x) + c_1 \times i + c_2 \times i^2] \bmod M$$

  - M is the size of the hash table, $h'(x) = x \bmod M$, i is the probe number that varies from 0 to $M-1$, and c_1 and c_2 are constants such that $c_1 \neq 0$ and $c_2 \neq 0$

$h(x,i) = [h'(x) + c_1 \times i + c_2 \times i^2] \bmod M$

Example.

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table
  - If we set $c_1 = 1$ and $c_2 = 3$
  - Initial

0	1	2	3	4	5	6	7	8	9
–1	–1	–1	–1	–1	–1	–1	–1	–1	–1

  - Step 1
    - $x = 72$
    - $h(72,0) = [h'(72) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(72 \bmod 10) + 0 + 0] \bmod 10 = 2$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	–1	–1	–1	–1

$h(x,i) = [h'(x) + c_1 \times i + c_2 \times i^2] \bmod M$

Example..

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table
  - Step 2
    - $x = 27$
    - $h(27,0) = [h'(27) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(27 \bmod 10) + 0 + 0] \bmod 10 = 7$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	–1	27	–1	–1

  - Step 3
    - $x = 36$
    - $h(36,0) = [h'(36) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(36 \bmod 10) + 0 + 0] \bmod 10 = 6$

0	1	2	3	4	5	6	7	8	9
–1	–1	72	–1	–1	–1	36	27	–1	–1

$h(x,i) = [h'(x) + c_1 \times i + c_2 \times i^2] \bmod M$

Example…

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table
  - Step 4
    - $h(24,0) = [h'(24) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(24 \bmod 10) + 0 + 0] \bmod 10 = 4$
  - Step 5
    - $h(63,0) = [h'(63) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(63 \bmod 10) + 0 + 0] \bmod 10 = 3$
  - Step 6
    - $h(81,0) = [h'(81) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(81 \bmod 10) + 0 + 0] \bmod 10 = 1$

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	–1	36	27	–1	–1

$h(x,i) = [h'(x) + c_1 \times i + c_2 \times i^2] \bmod M$

Example….

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	–1	36	27	–1	–1

  - Step 7
    - $h(101,0) = [h'(101) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(101 \bmod 10) + 0 + 0] \bmod 10 = 1$
    - $h(101,1) = [h'(101) + 1 \times 1 + 3 \times 1^2] \bmod 10$
      $= [(101 \bmod 10) + 1 + 3] \bmod 10 = 5$

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	101	36	27	–1	–1

Example.....

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	101	36	27	–1	–1

  - Step 8
    - $h(92,0) = [h'(92) + 1 \times 0 + 3 \times 0^2] \bmod 10$
      $= [(92 \bmod 10) + 0 + 0] \bmod 10 = 2$
    - $h(92,1) = [h'(92) + 1 \times 1 + 3 \times 1^2] \bmod 10$
      $= [(92 \bmod 10) + 1 + 3] \bmod 10 = 6$
    - $h(92,2) = [h'(92) + 1 \times 2 + 3 \times 2^2] \bmod 10$
      $= [(92 \bmod 10) + 2 + 12] \bmod 10 = 6$
    - $h(92,3) = [h'(92) + 1 \times 3 + 3 \times 3^2] \bmod 10$
      $= [(92 \bmod 10) + 3 + 27] \bmod 10 = 2$
    - $h(92,4) = [h'(92) + 1 \times 4 + 3 \times 4^2] \bmod 10$
      $= [(92 \bmod 10) + 4 + 48] \bmod 10 = 4$
    - $h(92,5) = [h'(92) + 1 \times 5 + 3 \times 5^2] \bmod 10$
      $= [(92 \bmod 10) + 5 + 75] \bmod 10 = 2$

Example……

- Consider a hash table of size 10, please use quadratic probing to insert the keys 72, 27, 36, 24, 63, 81, 101, and 92 into the table
  - $h(92,6) = [h'(92) + 1 \times 6 + 3 \times 6^2] \bmod 10$
    $= [(92 \bmod 10) + 6 + 108] \bmod 10 = 6$
  - $h(92,7) = [h'(92) + 1 \times 7 + 3 \times 7^2] \bmod 10$
    $= [(92 \bmod 10) + 7 + 147] \bmod 10 = 6$
  - $h(92,8) = [h'(92) + 1 \times 8 + 3 \times 8^2] \bmod 10$
    $= [(92 \bmod 10) + 8 + 192] \bmod 10 = 2$
  - $h(92,9) = [h'(92) + 1 \times 9 + 3 \times 9^2] \bmod 10$
    $= [(92 \bmod 10) + 9 + 243] \bmod 10 = 4$

  – One of the major **drawbacks** of quadratic probing is that a sequence of successive probes **may only explore a fraction of the table**, and **this fraction may be quite small**
    - If this happens, then we will not be able to find an empty location in the table despite the fact that the table is not full

Double Hashing

- Double hashing uses one hash value and then repeatedly steps forward an interval until an empty location is reached
  - The interval is decided using a second, independent hash function, hence the name **double hashing**

$$h(x, i) = [h_1(x) + i \times h_2(x)] \bmod M$$

  - M is the size of the hash table
  - $h_1(x)$ and $h_2(x)$ are two hash functions
    - $h_1(x) = x \bmod M$
    - $h_2(x) = x \bmod M'$
    - M' is chosen to be less than M

      We can choose $M' = M - 1$ or $M' = M - 2$
  - i is the probe number that varies from 0 to $M-1$

Example.

- Consider a hash table of size 10, please use double hashing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table
  - If we set $h_1 = x \bmod 10$ and $h_2 = x \bmod 8$
  - Initial

0	1	2	3	4	5	6	7	8	9
–1	–1	–1	–1	–1	–1	–1	–1	–1	–1

  - Step 1
    - $x = 72$
    - $h(72,0) = [h_1(72) + 0 \times h_2(72)] \bmod 10$
      $= [(72 \bmod 10) + 0 \times (72 \bmod 8)] \bmod 10 = 2$
  - Step 2
    - $x = 27$
    - $h(27,0) = [h_1(27) + 0 \times h_2(27)] \bmod 10$
      $= [(27 \bmod 10) + 0 \times (27 \bmod 8)] \bmod 10 = 7$

Example..

- Consider a hash table of size 10, please use double hashing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table
  - Step 3
    - $x = 36$
    - $h(36,0) = [h_1(36) + 0 \times h_2(36)] \bmod 10$
      $= [(36 \bmod 10) + 0 \times (36 \bmod 8)] \bmod 10 = 6$
  - Step 4
    - $x = 24$
    - $h(24,0) = [h_1(24) + 0 \times h_2(24)] \bmod 10$
      $= [(24 \bmod 10) + 0 \times (24 \bmod 8)] \bmod 10 = 4$
  - Step 5
    - $x = 63$
    - $h(63,0) = [h_1(63) + 0 \times h_2(63)] \bmod 10$
      $= [(63 \bmod 10) + 0 \times (63 \bmod 8)] \bmod 10 = 3$

Example...

- Consider a hash table of size 10, please use double hashing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table
  - Step 6
    - $x = 81$
    - $h(81,0) = [h_1(81) + 0 \times h_2(81)] \bmod 10$
      $= [(81 \bmod 10) + 0 \times (81 \bmod 8)] \bmod 10 = 1$

0	1	2	3	4	5	6	7	8	9
–1	81	72	63	24	–1	36	27	–1	–1

  - Step 7
    - $x = 92$
    - $h(92,0) = [h_1(92) + 0 \times h_2(92)] \bmod 10$
      $= [(92 \bmod 10) + 0 \times (92 \bmod 8)] \bmod 10 = 2$
    - $h(92,1) = [h_1(92) + 1 \times h_2(92)] \bmod 10$
      $= [(92 \bmod 10) + 1 \times (92 \bmod 8)] \bmod 10 = 6$
    - $h(92,2) = [h_1(92) + 2 \times h_2(92)] \bmod 10$
      $= [(92 \bmod 10) + 2 \times (92 \bmod 8)] \bmod 10 = 0$

Example....

- Consider a hash table of size 10, please use double hashing to insert the keys 72, 27, 36, 24, 63, 81, 92, and 101 into the table

0	1	2	3	4	5	6	7	8	9
92	81	72	63	24	–1	36	27	–1	–1

  - Step 8
    - $x = 101$
    - $h(101,0) = [h_1(101) + 0 \times h_2(101)] \bmod 10$
      $= [(101 \bmod 10) + 0 \times (101 \bmod 8)] \bmod 10 = 1$
    - $h(101,1) = [h_1(101) + 1 \times h_2(101)] \bmod 10$
      $= [(101 \bmod 10) + 1 \times (101 \bmod 8)] \bmod 10 = 6$
    - $h(101,2) = [h_1(101) + 2 \times h_2(101)] \bmod 10$
      $= [(101 \bmod 10) + 2 \times (101 \bmod 8)] \bmod 10 = 1$
    - ....
  - Double hashing has the same drawback as the quadratic probing method

Rehashing

- When the hash table becomes nearly full, the number of collisions increases, thereby degrading the performance of insertion and search operations
  - A better option is to **create a new hash table** with size double of the original hash table

- By performing the rehashing, all the entries in the original hash table will then have to be moved to the new hash table
  - This is done by taking each entry, computing its new hash value, and then inserting it in the new hash table

- Though rehashing seems to be a simple process, it is **quite expensive** and must therefore not be done frequently

Example

- Consider the hash table of size 5 given below
  - The hash function used is $h(x) = x \bmod 5$ with linear probing

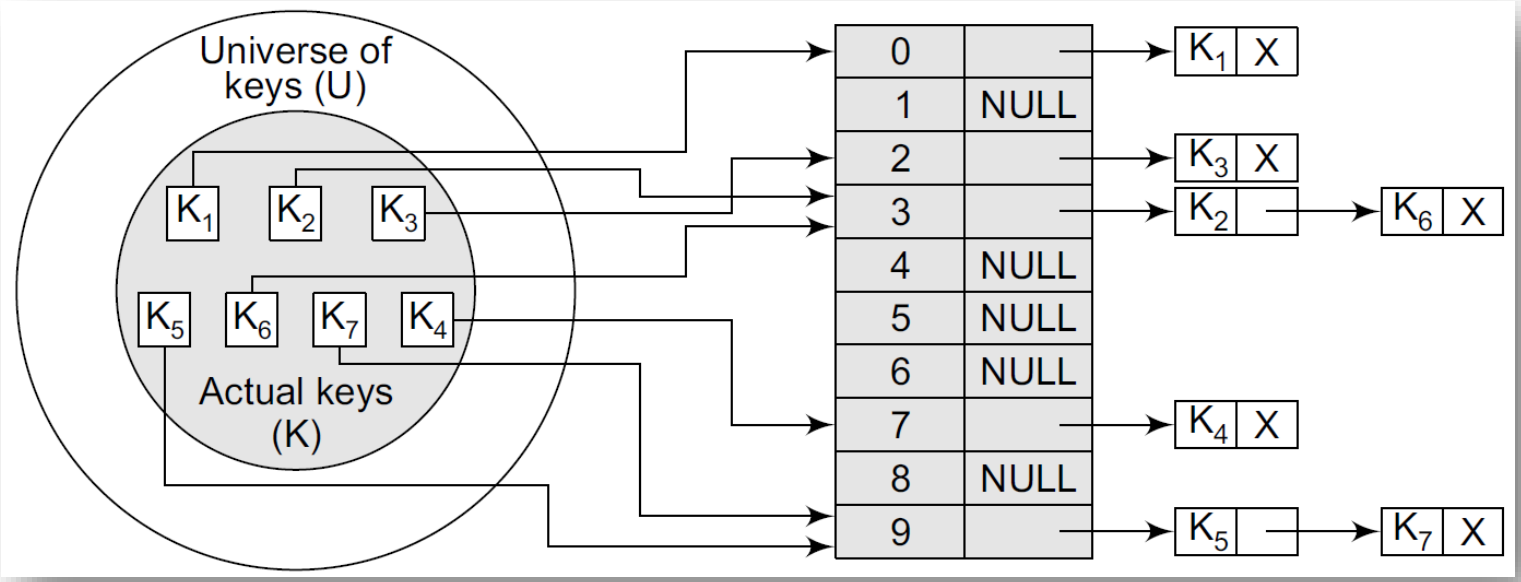
0	1	2	3	4
	26	31	43	17

  - Rehash the entries into to a new hash table
    - Note that the new hash table is of 10 locations, double the size of the original table
    - Rehash the key values from the old hash table into the new one using hash function $h(x) = x \bmod 10$ with linear probing

0	1	2	3	4	5	6	7	8	9
	31		43			26	17		

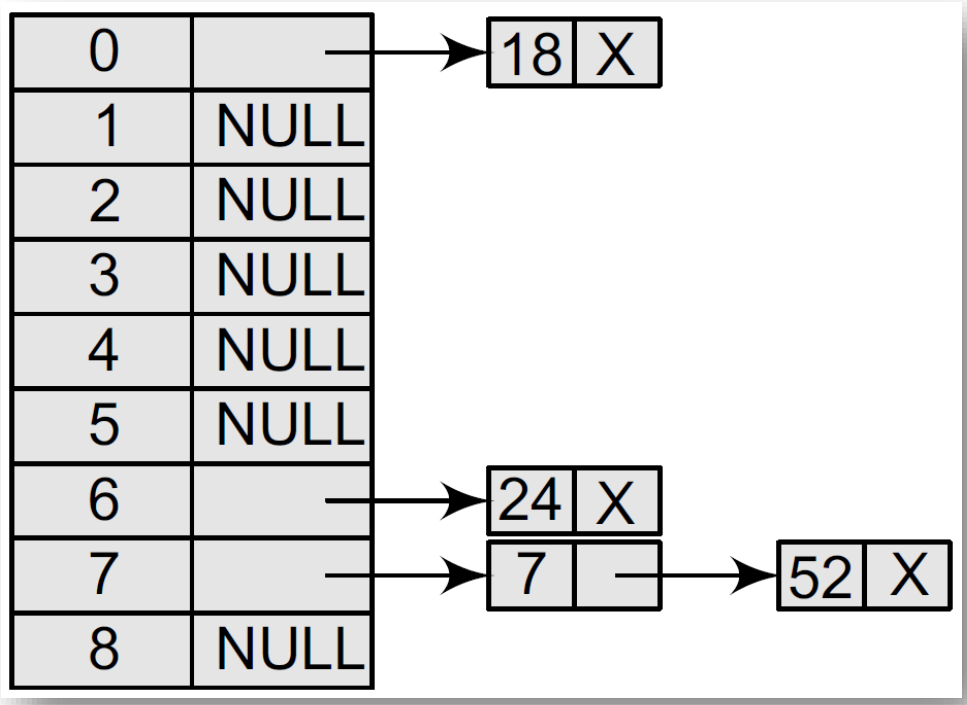
Chaining

- In chaining, each location in a hash table stores a pointer to a linked list that contains all the key values that were hashed to that location
  - Location n in the hash table points to the head of the linked list of all the key values that hashed to n
  - If no key value hashes to n, then location n in the hash table contains NULL



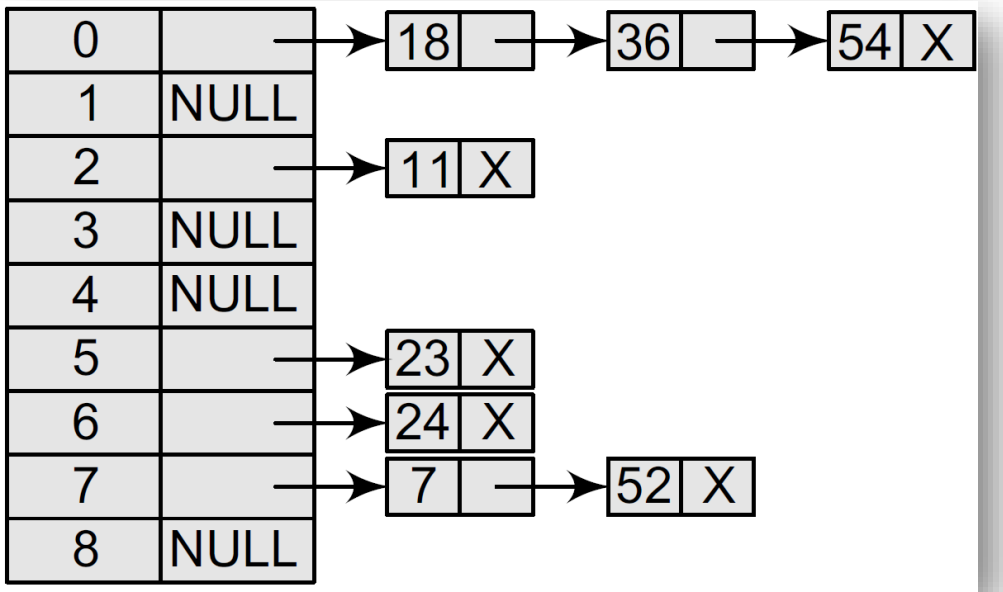
Example.

- Insert the keys 7, 24, 18, 52, 36, 54, 11, and 23 in a chained hash table of 9 memory locations
  - The hash function is $h(x) = x \bmod 9$
  - Step 1
    - $x = 7$
    - $h(7) = 7 \bmod 9 = 7$
  - Step 2
    - $x = 24$
    - $h(24) = 24 \bmod 9 = 6$
  - Step 3
    - $x = 18$
    - $h(18) = 18 \bmod 9 = 0$
  - Step 4
    - $x = 52$
    - $h(52) = 52 \bmod 9 = 7$

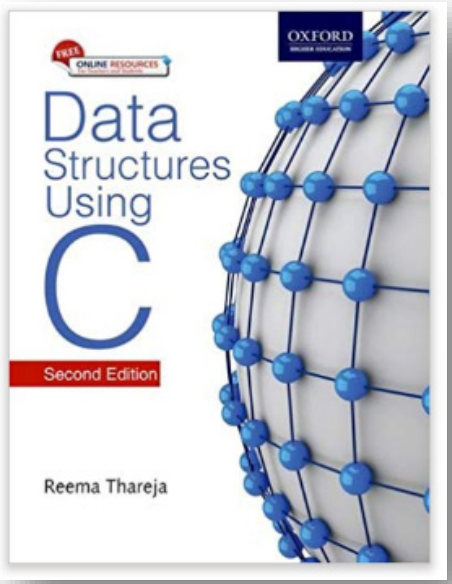


Example..

- Insert the keys 7, 24, 18, 52, 36, 54, 11, and 23 in a chained hash table of 9 memory locations
  - Step 5
    - $x = 36$
    - $h(36) = 36 \bmod 9 = 0$
  - Step 6
    - $x = 54$
    - $h(54) = 54 \bmod 9 = 0$
  - Step 7
    - $x = 11$
    - $h(11) = 11 \bmod 9 = 2$
  - Step 8
    - $x = 23$
    - $h(23) = 23 \bmod 9 = 5$



Summary



Grading

- Homework: 35%
  - HW0: 3%
  - HW1 ~ HW4: 32%
- Midterm: 35%
- Final: 40%

- TAs:
  - 黃郁洺
  - 郭勁宏
  - 湯冠維

Teaching Assistant Recruitment

- If you
  - want to teach junior students
  - have good idea to train students

- Please contact me in July!

Questions?

kychen@mail.ntust.edu.tw