Hash Tables (continued) & Disjoint Sets
Chapter 5 & 8 in Weiss

Quadratic Probing

f(i) = i²

- Probe sequence:
  0th probe = h(k) mod TableSize
  1st probe = (h(k) + 1) mod TableSize
  2nd probe = (h(k) + 4) mod TableSize
  3rd probe = (h(k) + 9) mod TableSize
  . . . 
  ith probe = (h(k) + i²) mod TableSize

Less likely to encounter Primary Clustering

Quadratic Probing Example

Insert:
89
18
49
58
79

Quadratic Probing: Properties

- For any λ < ½, quadratic probing will find an empty slot; for bigger λ, quadratic probing may find a slot
- Quadratic probing does not suffer from primary clustering: keys hashing to the same area are not bad
  - Secondary Clustering!

Quadratic Probing: Success guarantee for λ < ½

- If size is prime and λ < ½, then quadratic probing will find an empty slot in size/2 probes or fewer.
  - show for all 0 ≤ i, j ≤ size/2 and i ≠ j
    - h(k) + i² mod size ≠ h(k) + j² mod size
  - by contradiction: suppose that for some i ≠ j:
    - h(k) + i² mod size = h(k) + j² mod size
    - i² mod size = j² mod size
    - (i² - j²) mod size = 0
    - [(i + j)(i - j)] mod size = 0
    - BUT size does not divide (i+j) or (i-j)

Quadratic Probing:

Success guarantee for λ < ½

- If size is prime and λ < ½, then quadratic probing will find an empty slot in size/2 probes or fewer.
  - show for all 0 ≤ i, j ≤ size/2 and i ≠ j
    - h(k) + i² mod size ≠ h(k) + j² mod size
  - by contradiction: suppose that for some i ≠ j:
    - h(k) + i² mod size = h(k) + j² mod size
    - i² mod size = j² mod size
    - (i² - j²) mod size = 0
    - [(i + j)(i - j)] mod size = 0
    - BUT size does not divide (i+j) or (i-j)
Double Hashing

\[ f(i) = i \cdot g(k) \]

where \(g\) is a second hash function

- Probe sequence:
  
  0\textsuperscript{th} probe = \(h(k) \mod \text{TableSize}\)
  
  1\textsuperscript{st} probe = \((h(k) + g(k)) \mod \text{TableSize}\)

  2\textsuperscript{nd} probe = \((h(k) + 2g(k)) \mod \text{TableSize}\)

  3\textsuperscript{rd} probe = \((h(k) + 3g(k)) \mod \text{TableSize}\)

  \ldots

  \(i\textsuperscript{th} \) probe = \((h(k) + ig(k)) \mod \text{TableSize}\)

Double Hashing Example

\[ h(k) = k \mod 7 \text{ and } g(k) = 5 - (k \mod 5) \]

Insert these values into the hash table in this order. Resolve any collisions with double hashing:

13, 28, 33, 147, 43

Resolving Collisions with Double Hashing

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hash Functions:

\[ H(K) = K \mod M \]

\[ H_2(K) = 1 + \left((K/M) \mod (M-1)\right) \]

\[ M = 10 \]

Insert these values into the hash table in this order. Resolve any collisions with double hashing:

13, 28, 33, 147, 43

Rehashing

**Idea:** When the table gets too full, create a bigger table (usually 2x as large) and hash all the items from the original table into the new table.

- When to rehash?
  - half full (\(\lambda = 0.5\))
  - when an insertion fails
  - some other threshold
- Cost of rehashing?

Hashing Summary

- Hashing is one of the most important data structures.
- Hashing has many applications where operations are limited to find, insert, and delete.
- Dynamic hash tables have good amortized complexity. (cost of doubling table and rehashing is amortized over many inserts)

Disjoint Sets

Chapter 8
Disjoint Union - Find

• Maintain a set of pairwise disjoint sets.
  \( \{3,5,7\}, \{4,2,8\}, \{9\}, \{1,6\} \)
• Each set has a unique name, one of its members
  \( \{3,5,7\}, \{4,2,8\}, \{9\}, \{1,6\} \)

Union

• \( \text{Union}(x,y) \) – take the union of two sets named \( x \) and \( y \)
  \( \{3,5,7\}, \{4,2,8\}, \{2\}, \{1,6\} \)
  \( \text{Union}(5,1) \)
  \( \{3,5,7,1,6\}, \{4,2,8\}, \{2\} \).

Find

• \( \text{Find}(x) \) – return the name of the set containing \( x \).
  \( \{3,5,7,1,6\}, \{4,2,8\}, \{2\} \),
  \( \text{Find}(1) = 5 \)
  \( \text{Find}(4) = 8 \)

Building Mazes

• Build a random maze by erasing edges.

Building Mazes (2)

• Pick Start and End

Building Mazes (3)

• Repeatedly pick random edges to delete.
Desired Properties

- None of the boundary is deleted
- Every cell is reachable from every other cell.
- Only one path from any one cell to another
  (There are no cycles – no cell can reach itself by a path unless it retraces some part of the path.)

A Cycle

A Good Solution

A Hidden Tree

Number the Cells

We have disjoint sets \( S = \{1\}, \{2\}, \{3\}, \ldots, \{36\}\) each cell is unto itself.
We have all possible edges \( E = \{(1,2), (1,7), (2,8), (2,3), \ldots\}\) 60 edges total.

<table>
<thead>
<tr>
<th>Start</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>End</td>
</tr>
</tbody>
</table>

Basic Algorithm

- \( S = \) set of sets of connected cells
- \( E = \) set of edges
- \( Maze = \) set of maze edges (initially empty)

```
While there is more than one set in S {
        pick a random edge \((x,y)\) and remove from E
        u := Find(x);
        v := Find(y);
        if u \(!=\) v then  // removing edge \((x,y)\) connects previously non-
                         // connected cells x and y - leave this edge removed!
            Union(u,v)
        else  // cells x and y were already connected, add this
            // edge to set of edges that will make up final maze.
            add \((x,y)\) to Maze
    }
All remaining members of E together with Maze form the maze
```
Example Step

Pick (8,14)

Start

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td></td>
</tr>
</tbody>
</table>

End

S
\{1,2,7,8,9,13,19,14,20,26,27\}

\{3\}

\{4\}

\{5\}

\{6\}

\{10\}

\{11,17\}

\{12\}

\{15,16,21\}

\{22,23,24,29,39,32,33,34,35,36\}

Example

Pick (8,14)

Find(8) = 7

Find(14) = 20

Union(7,20)

S
\{1,2,7,8,9,13,19,14,20,26,27\}

\{3\}

\{4\}

\{5\}

\{6\}

\{10\}

\{11,17\}

\{12\}

\{15,16,21\}

\{22,23,24,29,39,32,33,34,35,36\}

Example at the End

Pick (19,20)

Start

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td></td>
</tr>
</tbody>
</table>

End

S
\{1,2,7,8,9,13,19,14,20,26,27\}

\{3\}

\{4\}

\{5\}

\{6\}

\{10\}

\{11,17\}

\{12\}

\{15,16,21\}

\{22,23,24,29,39,32,33,34,35,36\}

Example at the End

Pick (19,20)

Start

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td></td>
</tr>
</tbody>
</table>

End

S
\{1,2,3,4,5,6,...,36\}

\[E\]

\[\text{Maze}\]