Advanced Data Structures
Simon Gog – gog@kit.edu
Dynamic Perfect Hashing

What we want:
- $O(1)$ lookup time (as in static perfect hashing)
- Keys not known in advance
- Good expected performance for insert

Cuckoo hashing
- Uses *two* hash functions $h_1$ and $h_2$
- Key $x$ stored either at position $h_1(x)$ or at $h_2(x)$
- At most one key per position in the hash table
- Worst case lookup time: $O(1)$
- Removing a key is also constant
- Insertion of a key is $O(1)$ expected, amortized
Cuckoo hashing

00  lookup(x)
01       if T[h₁(x)] = x or T[h₂(x)] = x then
02           return true
03           return false

00  insert(x)
01       if lookup(x) then
02           return
03           p ← h₁(x)
04           for i ← 0 to n − 1 do
05               if T[p] = ⊥ then
06                   T[p] ← x; return
07                   swap(x, T[p])
08                   p ← h₁+(p=h₁(x))(x)
09               rehash(); insert(x)
Cuckoo graph

The *cuckoo (undirected) graph* consists of
- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

Example
- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
Cuckoo graph

The *cuckoo (undirected) graph* consists of
- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

Example
- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
The **cuckoo (undirected) graph** consists of:
- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

**Example**
- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
The **cuckoo (undirected) graph** consists of:

- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

**Example**

- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
The *cuckoo (undirected) graph* consists of:

- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

**Example**

- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
The cuckoo (undirected) graph consists of
- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

Example
- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
Cuckoo graph

The *cuckoo (undirected) graph* consists of

- $m$ nodes (one for each table entry)
- For each key $x$ there is an edge connecting $h_1(x)$ and $h_2(x)$

**Example**

- Including key $x_5$ causes a cycle. Are cycles dangerous? What is the probability of getting a cycle?
- Including key $x_6$ will result in a rehash
Cuckoo hashing – analysis

Assumptions

- Keys have the same size and can be compared in constant time
- Two hash functions $h_1$ and $h_2$ which map to $[m]$. The probability for any function value $h_i(x)$ to be a particular value in $[m]$ is $\frac{1}{m}$. Function values are independent of each other.
- Fixed upper bound $n$ on the number of keys in the set $S$. 
Cuckoo hashing – analysis

Lemma ([1])
For any position $i$ and $j$, and any $c > 1$, if $m \geq 2cn$ then the probability that in the undirected cuckoo graph there exists a path from $i$ to $j$ of length $\ell \geq 1$, which is a shortest path from $i$ to $j$, is at most $\frac{1}{c^\ell m}$.

Proof (by induction)

- Base case: $\ell = 1$
- For each $x \in S$ we have
  $\Pr(x \text{ mapped to node } i \text{ and } j) = \frac{2}{m^2}$, since either $h_1(x) = i \land h_2(x) = j$ or $h_1(x) = j \land h_2(x) = i$
- Using union bound, we get that the probability that there is an edge between $i$ and $j$ is at most

$$\sum_{x \in S} \frac{2}{m^2} \leq \frac{2n}{m^2} \leq \frac{n}{2c} \leq \frac{1}{cm}$$
Cuckoo hashing – analysis

Proof continued

- Inductive step: $\ell > 1$ and lemma holds for length $\leq \ell - 1$
- If there is a path between $i$ and $j$ of length $\ell$ but not shorter than $\ell$ then there must be a position $k$ such that
  - A there is a shortest path of length $\ell - 1$ from $i$ to $k$ that does not go through $j$, and
  - B there is an edge from $k$ to $j$
- $\Pr(A) \leq \frac{1}{c^{\ell-1} m}$, by induction hypothesis and the fact the requirement „does not go through $j$“ makes the probability even smaller
- $\Pr(B|A) = \sum_{x \in S} \frac{2}{m^2} \leq \frac{1}{cm}$
- $\Pr(A \text{ and } B) = \Pr(A) \cdot \Pr(B|A) \leq \frac{1}{c^{\ell-1} m} \cdot \frac{1}{cm} = \frac{1}{c^\ell m^2}$
- Sum of all possible $k$ and using union bound gives an upper bound on the probability of a shortest path of length $\ell$ between $i$ and $j$ of $\frac{1}{c^\ell m}$
Cuckoo hashing – analysis

- Two keys are in the same bucket if a path connects \{h_1(x), h_2(x)\} and \{h_1(y), h_2(y)\} in the cuckoo graph (there are 4 possible ways to do this).

- Probability of two keys \(x \neq y\) to be in the same bucket can be upper bounded by

\[
4 \sum_{\ell=1}^{\infty} \frac{1}{c^\ell m} = \frac{4}{(c - 1)m} = O\left(\frac{1}{m}\right)
\]
Assume there are no cycles in the cuckoo graph
From the previous lecture we know that the time for an operation is bounded by the number of elements in the bucket
With the same analysis we get that the expected time per operation is $O(1)$ and $O(1)$ worst case on lookups (Assuming $m \geq 2cn$).

Next, analysis of the cost of rehashing...
Cuckoo hashing – analysis

Rehashing

- Consider sequence of operations involving $\epsilon n$ insertions (e.g. $\epsilon = 0.1$)
- Let $S'$ be the set of keys that exists at some time during insertions
- How likely is a cycle (=path from node $i$ back to itself)? With the previous lemma we can upper bound that a position $i$ is involved in a cycle

$$
\sum_{\ell=1}^{\infty} \frac{1}{c^\ell m} = \frac{1}{(c - 1)m}
$$

for $m \geq 2c(1 + \epsilon)n$

- Using union bound, we get an upper bound for the probability that there is at least one cycle:

$$
\sum_{i=1}^{m} \frac{1}{(c - 1)m} = \frac{1}{(c - 1)}
$$
Cuckoo hashing – analysis

Rehashing

- For $c = 3$, the probability is at most $\frac{1}{2}$ that a cycle occurs (i.e. a rehash could be required) during the $\epsilon n$ insertions.
- The probability of two rehashes (caused by a second independent cycle) is $\frac{1}{4}$, and so on.
- I.e. the expected number of rehashes during $\epsilon n$ insertions is at most

$$\sum_{i=1}^{\infty} \frac{1}{2^i} = 1$$

- If a rehash takes $O(n)$ time the expected amortized time of rehashes over $\epsilon n$ insertions is $O(\frac{1}{\epsilon})$, i.e. constant.
Cuckoo hashing – analysis

Global rebuilding

- Adapt the size of the hash table to the number of keys.
- Whenever the set becomes too small/large compared to the size of the hash table, a new smaller/larger hash table is created.
- To guarantee constant expected amortized cost per operation the size should be decreased/increased by a constant factor.
Our assumption of true randomness is not realistic.

Original work uses concept of \((c, k)\)-universal hash functions. Here the hash values of any choice of \(k\) keys are independent.

It can be shown that cuckoo hashing still performs well using \((c, k)\)-universal hash functions: Perform a rehash if a key cannot be inserted after \(k = \log n\) steps (instead of \(n\) in true randomness case).

Siegel [FOCS 1989] showed that \((1, O(\log n))\)-universal hash functions exists (taking \(O(\log n)\) space and can be evaluated in \(O(1)\) time)