Choosing Hash Functions

- Mostly black magic; division method: \( h(k) = k \mod m \)
  - do not use \( m = 2 \) (will not use all the bits)
- Multiplication method:
  - choose \( m \) not = 2, \( p \), not too close to power of 2 or 10.
  - If \( m = 2^p \), then we do is scramble by multiplication, and choose \( p \) bits to the left of binary point.

More multiplicative method

Example: \( m = 8 \):
- each time \( k \) increased:
  - go around the circle,
  - read off sector number.
- Note what happens if \( A = 0.5 \) or \( A = 1/2 \).

Universal Hashing

- biggest problem with hash function:
  - There is always an adversarial sequence that "kills" it!
- Can not choose truly random function - \( m \) to the power of keys different functions. Too much storage!!
- We need a small family \( H \) of hash functions, such that, for any input, any small percentage of these functions are "killed".
- Existence of such family? Size?
  - First, let's look at properties: what if \( h() \) is truly random?
    - \( \Pr[ h(x) = h(y) ] \sim \frac{1}{m} \)
  - Then:
    - \( \Pr[ h(x) = h(y) ] = \frac{\sum_{i=1}^{m} \Pr[ h(x) = h(y) ]}{m} = \frac{1}{m} \)

Construction

Universal Hash Functions

- Need: for any \( x \) and \( y \), proportion of functions in \( H \) that map both \( x \) and \( y \) to the same slot is 1/m.
- Take \( m \) prime.
  - input \( n \times k \), output \( n \times k \times m \) \( \forall A \in \mathbb{R} \)
  - \( x_i \in [0, m-1] \) chosen uniformly at random.
- Define a function for each possible choice of \( a \).
  - \( h(k) = \sum_{i=1}^{m} a_i \mod m \)
- Claim: the family \( H \) is universal.

Proving Universality

- Total number of functions in \( H \): \( m^n \)
- Given particular \( x \) and \( y \), what proportion of these functions map \( h(x) \neq h(y) \)? \( \frac{m^n - 1}{m^n} \)
- Choose \( a \), \( a \neq a , \) etc first. There are \( m \) choices. Now we need to choose \( a \), to make \( h(x) = h(y) \):
  - \( a = \sum_{i=1}^{m} (x_i - y_i) \mod m \)
  - If \( a \neq 0 \) there is only 1 solution.
- Thus, total number of functions such that \( h(x) = h(y) \) is \( m^n \), exactly the right property.
**Binary Search Trees**  
(Chapter 13)

- In addition to insert/delete:
  - Heaps supported insertion.
  - Hashing supported search.
  - What if we want both min/max/search, and also pred/succ?
- Binary Search trees:

```
     x
   /   \
  y   z
```

```
left tree  key
right tree key
```

**Examples**

- Legal B-Trees:
- In-Order walk: `InOrder(left(x)) print(x) InOrder(right(x))`
- Note that given B-tree, can output sorted in O(n) time! Gives lower bound on constructing B-Tree. (Compare with Heap!)