CS280 – Skip Lists
April 7, 2016

http://azrael.digipen.edu/~mmead/www/Courses/CS280/SkipLists.html

Homework

- Building adjacency list is wrong in stress test
    - Building list wrong
- Shouldn't be 'checking for duplicates' anywhere
- Make sure not to call RNG yourself, will mess up future calls in driver

Skip Lists

- Relatively new data structure
- Like a fast fowrard ability for a linked list
    - Search quickly at first, slow down as you reach destination
- At any level 'n', we can move 2^n nodes
- Node height should increase as list gets large
- Have the potential to be better performance than tree
- Have to decide how big of node you want to create
    - Using dynamic memory since node sizes can be different, otherwise you would be wasting a ton of space
- Going to randomize list to get better performance, like hash table
- We need to 'load the dice' however so we get different probabilities depending on type of node
- Node height will grow proportionall with list size
- Using random lets us get good distribition and performance without having to re-arrange a

bunch of stuff each insert

  - Probably won't get exact data distribution, but close
  - Similar to hash tables – worst case could still happen, but overall performance is very good

- Mont Carlo algorithm
  - Algorithms that get better performance the more random the data is
- Use a different object allocator for each size of node
- Use distribution to figure out objects per page for each object allocator
- Locality of reference goes away since using different OA's, and going to different levels
- RAD Studio: Rapid Application Development
  - Premier tool to make GUI's in C++