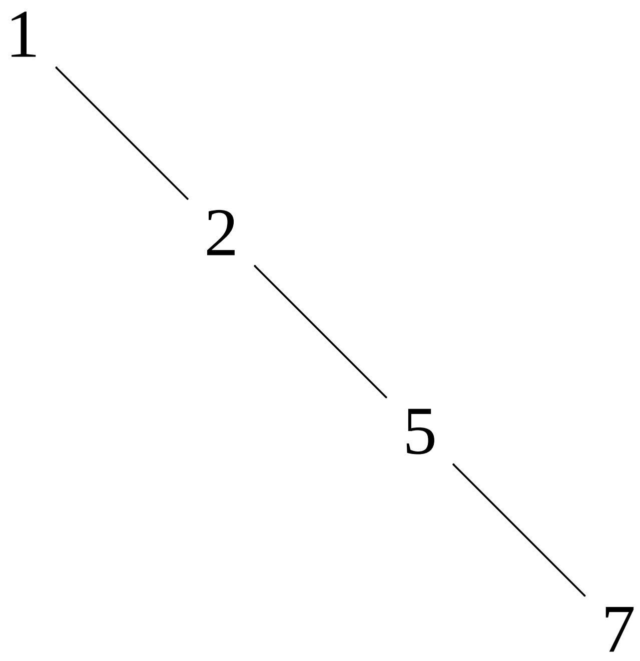
Hierarchical data structures

Bruce Merry

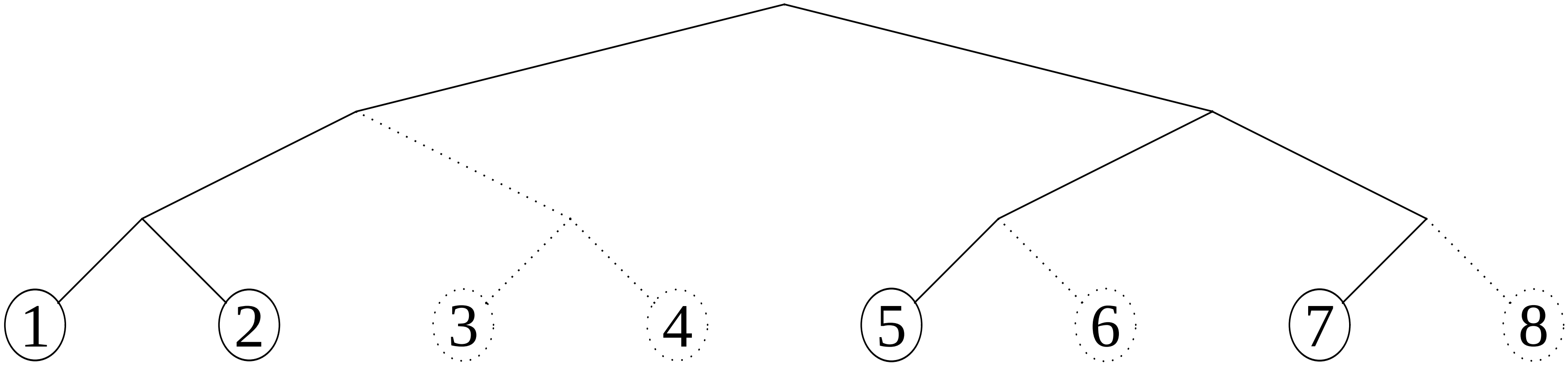
Binary search trees

Balancing is an issue.



Radix trees

Every possible number has a well-defined place in the tree.



Tag those present, and recurse only as deep as necessary.

- Numeric range R must be known in advance, but is not limited by memory.
- Operations are typically worst case $O(\log R)$ or $O(N \log R)$ (compare to **average case** $O(\log N)$ for binary search trees).

Radix trees and hash tables

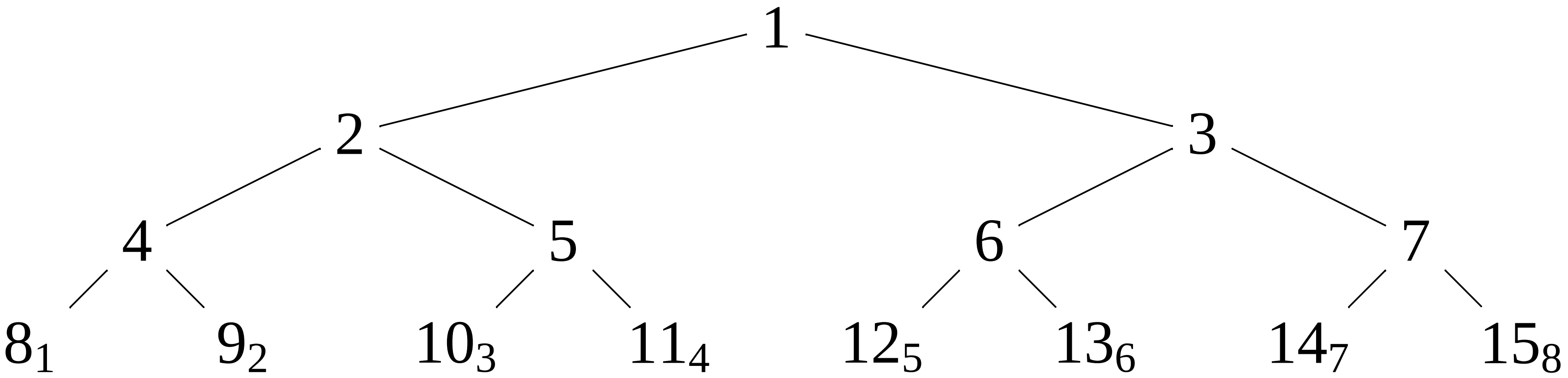
- Hash tables have less memory overhead.
- Access times for hash tables are usually faster (but not guaranteed).
- Hash tables are more general (can put in strings etc).

But

- Radix trees allow for range queries and range counts.
- Radix trees can store summary information in the higher nodes.

Radix trees for small ranges

If a complete radix tree fits in memory, then an array can be used as for heaps:



Memory allocation in linked structures

The system memory allocator has a lot of overhead. For grow only structures, maximum performance is achieved with your own.

- Allocate an array big enough to hold all the nodes.
- When you need a new element, return a pointer to the next one in the array.