COMP2860 Data structures

- Huffman trees

### Definitions

- Graph is $G=(V,E)$
  - set of vertices $V$
  - set of edges $E \subseteq V \times V$
- Tree is a connected
  - acyclic graph
  - graph on $n$ vertices and $n-1$ edges

### Rooted trees

- Distinguished root node
- Every node has a unique parent
- Ordered trees: ordering of children nodes
  - ordered sequence of children instead of set
- Extended tree
  - add (square) nodes for all empty sub-trees

### External and internal nodes

- Empty tree is represented by a single external node (box)
- Claim: Any binary tree with $n$ internal nodes has $n+1$ external nodes

### Trees as binary trees

- Any tree can be converted to a binary tree
- Usually we wish to preserve some ordering for the nodes
- Many different binary trees

### Operations on binary trees

- Insert, delete
- Restructuring the tree
- Complexity of operations
  - depends heavily on the structure of the tree
Properties of trees

- height of a node $h(x)$: max path length from the node to a leaf (or external node)
  - external nodes have height 0
- height of a tree is the height of its root
- depth of a node $d(x)$: path to the root
- internal path length $i(T)$
- external path length $e(T)$

$$e(T) = i(T) + 2n$$

Internal and external paths

- Internal path length: sum of distances from the root
- Theorem: For any binary tree $T$ on $n$ internal nodes, $e(T) = i(T) + 2n$
  - Proof: by induction

Skew trees

- For different binary trees on $n$ nodes, internal path length may vary
- Skew trees: binary trees of maximal internal path length
  - and therefore max external path length
  - lead to bad performance
- Theorem: in skew trees, every node has at most one child (internal node)

Skew trees

- Theorem: in skew trees, every node has at most one child (internal node)
- Proof: prove two things:
  - A. skew tree implies one child per node
  - B. one child per node implies skew tree
Skew trees

- B. one child per node implies skew tree
- Consider the set S of all possible single-child trees on n nodes
  - chains
- Previous part proves maximal (skew) trees must lie in S
- All have the same path length
  \[ i(T) = \frac{n(n-1)}{2} \]

Balanced binary trees

- Definition 1: A binary tree is balanced iff there exists a number q such that every external node has depth either q or q+1
- Definition 2: A binary tree is balanced iff it has minimal internal path length

Tree measurements

- Binary tree T on n nodes
  - height is between \( O(\log n) \) and \( O(n) \)
  - internal and external path lengths are between \( O(n \log n) \) and \( O(n^2) \)

Huffman trees

- Data representation
  - example: ASCII characters
- Compression
  - Some characters appear more often
- Encode more frequent items with shorter representations

Prefix encoding

- No code word is a prefix of any other code word
- Tree representation

Prefix encoding

- Characters used often
  - encode by short words
- Example: cdabcccc
  - 17 bits on huffman
  - 80 bits on ASCII
Encoding sequences

- Encode sequence $S$
- $x$ appears $w(x)$ times in $S$
- Assume prefix encoding tree
- $d(x)$ depth of external node containing $x$
- Length of encoded sequence is $L$
- Given $S$ find optimal prefix encoding
  - find prefix tree that minimizes $L$

Optimal prefix tree encodings

- Define the weighted external path length of $T$ as follows:
  $$wepl(T) = \sum_{x \in E(T)} w(x)d(x)$$
- Trees of minimum external path length?
  - Might be many trees that min $wepl$
- Minimum weight external path tree is called a Huffman tree

Example

- Example: if the weights are $\{1, 1, 3, 5\}$
- Both trees are optimal
- $wepl = 17$
- If all weights are 1, Huffman is a balanced tree
  - since $wepl(T) = e(T)$

Huffman trees

- Huffman, 1952
- Start with all external nodes
  - with their weight
  - consider them as trivial trees
- Find two minimum weight trees and connect them into a new one
  - New weight for the tree is the sum of weights of combined trees
- Repeat until there is only one tree

Huffman example

- Implement with a priority queue

Correctness of Huffman algorithm

- Induction
- Definition: fringing forest for $T$ is a set of trees which may be "Huffman-combined" into $T$
- Main lemma: (loop invariant)
  - At any Intermediate step, set of trees is a fringing forest for some Huffman tree
- Proof:
  - True for base case
  - Inductive step. True for $F = \{T_1, \ldots\}$
Huffman correctness

- \( F = \{ T_1, \ldots \} \) is a fringing forest
- Next step: \( F' \) combines \( T_a \) and \( T_b \)
- Pick a node \( z \)
  - in \( T \)
  - not in \( F \)
  - of max depth
- Children of \( z, T_x \) and \( T_y \) must be in \( F \)
- Exchange sub trees keeping wepl minimal

Correctness of huffman

- New tree is also Huffman
- \( F' \) is a fringing forest for \( T' \)

\[ \text{wepl}(T_a) < \text{wepl}(T_x) \] (\( T_a \) was chosen)

\[ \text{wepl}(T_b) < \text{wepl}(T_y) \] (\( T_b \) was chosen)

\[ \text{wepl}(T_a) < \text{wepl}(T_x) \] swap \( T_a \) and \( T_x \)

\[ \text{wepl}(T_b) < \text{wepl}(T_y) \] swap \( T_b \) and \( T_y \)