COMP90038

Algorit plexity

Lecture 21: Huffman Encodi mpression

(with thanks to Harald Sønde hael Kirley)

Andres Munoz-Acosta

munoz.m@unimelb.edu.au

Peter Hall Building G.83

- The SES is open now. Please take a time to review this subject. All feedback is greatly appreciated.

- The final exam has be [illegible] M on Thursday 8th of November at Wilson H

  - It is a closed book exam. No calculator [illegible] ted.
  - Reading time will be 15 minutes. Exam duration will be 3 hours.
  - Answers must be provided in the exam paper in the space allocated.
  - The reverse side of the pages can be used for rough work.
  - All questions should be attempted. Some are easier than others.
  - Any unreadable parts will be considered wrong. Be neat in your answers.

- A sample exam paper will be provided this week.
  - We have instructed the tutors NOT to provide hints on the sample exam

- Assignment 2 is due n .
  - We will provide sample answers on du ure and through the LMS.

- Next week we will use both lectures for a quick review of the content.
  - Only examinable topics will be discussed in the review.

Recap

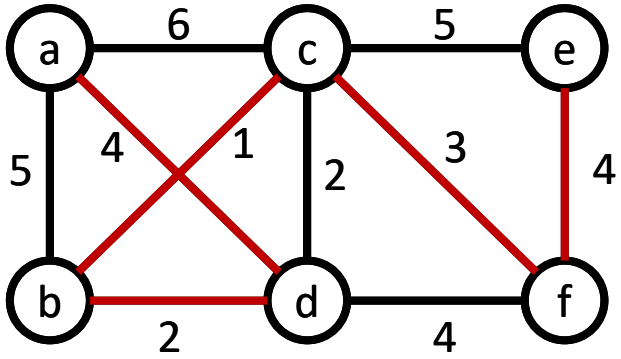
- We discussed **greedy algorithms**:
  - A problem solving strategy that takes the **locally best** choice among all feasible ones. Such choice is **irrevocable**.
  - Usually, **locally best** ch **est** results.
  - In some exceptions a g **and** .
  - Also, a greedy algorithm can provide g **mations**.

- We applied this idea to two graph problems :
  - Prim's algorithm for finding **minimum spanning trees**
  - Dijkstra's algorithm for **single-source shortest path**

What is a Minimum Spanning Tree?

- A **minimum spanning tree** of a weighted graph $\langle V,E \rangle$ is a tree $\langle V,E' \rangle$ where E' is a subset of E, such that the connections have the lowest cost

- We use Prim's algorithm panning tree.
  - It constructs a sequence **the latest est node not currently on it**.

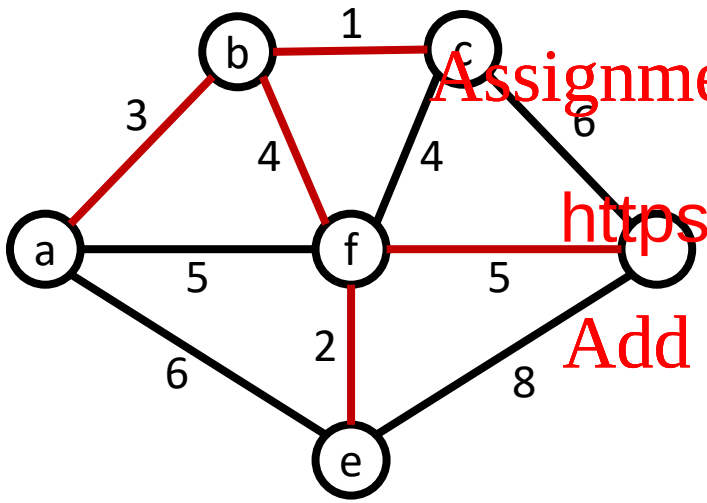


Prim's Algorithm

- We examined the complete algorithm, that uses priority queues:

Another example

- Let's work with the following graph:



- What would happen if we start on b?
  - The sequence will be different, but the edges may be the same
- How many different trees can we have?
  - If there are ties, the tie breaking has influence

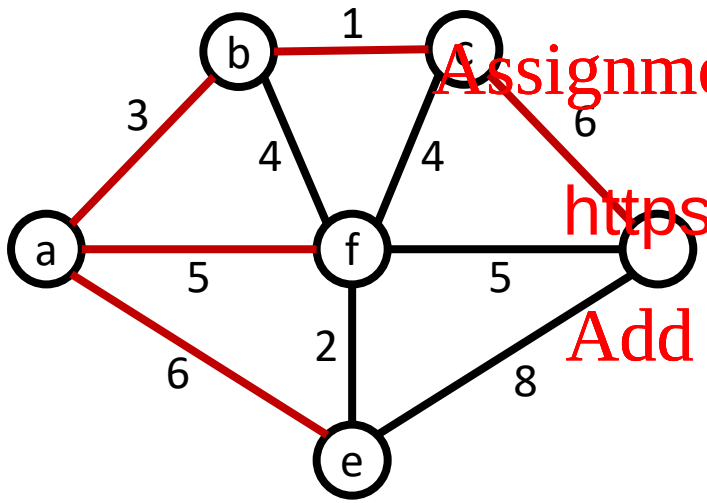
Tree T		a	b	c	d	e	f
	cost	∞	∞	∞	∞	∞	∞
	prev	nil	nil	nil	nil	nil	nil
	cost	0	∞	∞	∞	∞	∞
	prev	nil	nil	nil	nil	nil	nil
a	cost		3	∞	∞	6	5
	prev		a	nil	nil	a	a
a,b	cost			1	∞	6	4
	prev			b	nil	a	b
a,b,c	cost				6	6	4
	prev				c	a	b
a,b,c,f	cost				5	2	
	prev				f	f	
a,b,c,f,e	cost				5		
	prev				f		
a,b,c,f,e,d	cost						
	prev						

Dijkstra's Algorithm

- **Dijkstra's algorithm** finds all shortest paths **from a fixed start node**. Its complexity is the same as that of Prim's algorithm.

Another example

- Let's work with this graph again:



Tree T		a	b	c	d	e	f
	cost	∞	∞	∞	∞	∞	∞
	prev	nil	nil	nil	nil	nil	nil
	cost	0	∞	∞	∞	∞	∞
	prev	nil	nil	nil	nil	nil	nil
	cost		3	∞	∞	6	5
	prev		a	nil	nil	a	a
a,	cost			4	∞	6	5
	prev			b	nil	a	a
a,b,c	cost				10	6	5
	prev				c	a	a
a,b,c,f	cost				10	6	
	prev				c	a	
a,b,c,f,e	cost				10		
	prev				c		
a,b,c,f,e,d	cost						
	prev						

- What would happen if we start on b?
  - It is possible to end up with a different tree
- How many different trees can we have?
  - Ties can also influence the final tree.

Data compression

- From an information-theoretic point of view, most computer files contain much redundancy.

- Compression is used t ce.
  - For text files, savings up to 50 are com
  - For binary files, savings up to 90 are common.

- Savings in space mean savings in time for file transmission.

Run-Length Encoding

- For a text with long runs of **repeated characters**, we could compress by counting the runs. For example:

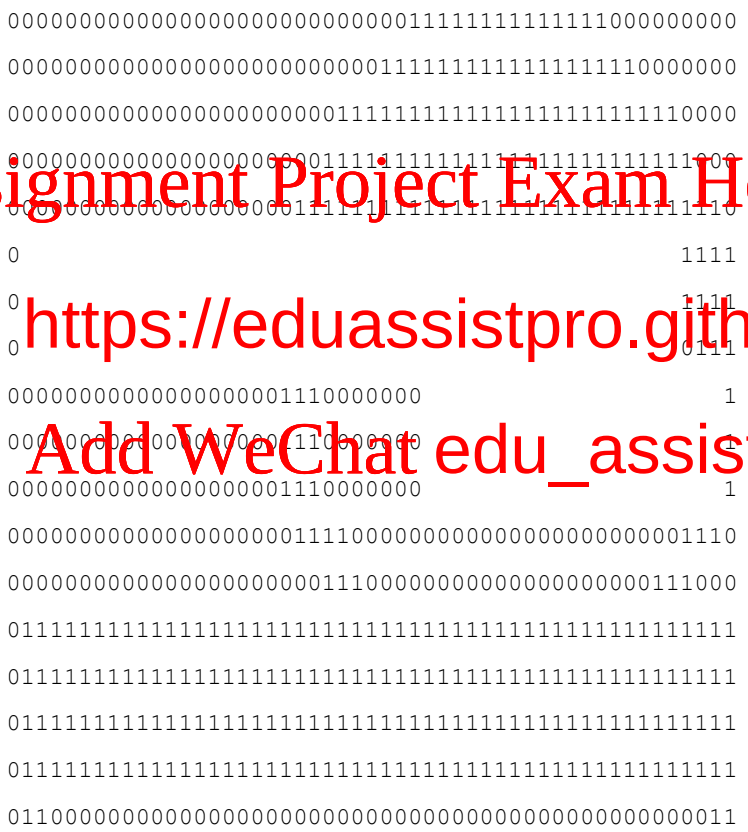
AAAABBBAA **AAABBBBCCCD**

- can then be encoded as:

4A3BAA5B8CDABCB3A4B3CD

- This is not useful for normal text. However, for **binary files** it can be very effective.

Run-Length Encoding



Variable-Length Encoding

- Fixed-length encoding uses a static number of symbols (bits) to represent a character.
  - For example, the ASCII code uses 8 bits per character.
- Variable-Length encodin to common characters.
  - In English, the most common character is could assign **0** to it.
  - However, no other character code can sta
- That is, no character's code should be a prefix of some other character's code (unless we somehow put separators between characters, which would take up space).

Variable-Length Encoding

- Suppose our alphabet is {A,B,C,D,E,F,G}
- We analyzed a text and following number of occ
- The last column shows some sensible codes that we may use for each symbol
  - Symbols with higher occurrence have shorter codes

SYMBOL	OCCURRENCE	CODE
	28	11
	4	0000
	14	011
	5	0001
E	27	10
F	12	010
G	10	001

Tries for Variable-Length Encoding

- A **trie** is a binary tree used on search applications
- To search for a key we look at individual bits of a key and descend to the **left** whenever a bit is **zero** and to the right whenever it is **one**
- Using a trie to determine codes means that n the prefix of another

Encoding messages

- To encode a message, we just need to concatenate the codes. For example:

F

010 1

B	A	G	G	E	D
0000	11	001	001	10	0001

- If we were to assign three bits per character, FACE would use 12 bits instead of 10. For BAGGED there is no space savings

SYMBOL	CODE
A	11
B	0000
C	011
D	0001
E	10
F	010
G	001

Decoding messages

- Try to decode **000110011111010** and **000011000100110** using the trie
  - Starting from the root, symbol found as a leaf
  - Repeat until the string is completed
- Remember the rules: Left branch is 0, right branch is 1

Huffman Encoding: Choosing the Codes

- Sometimes (for example for common English text) we may know the frequencies of letters fairly well.

- If we don't know about still count all characters in the given text as a first s

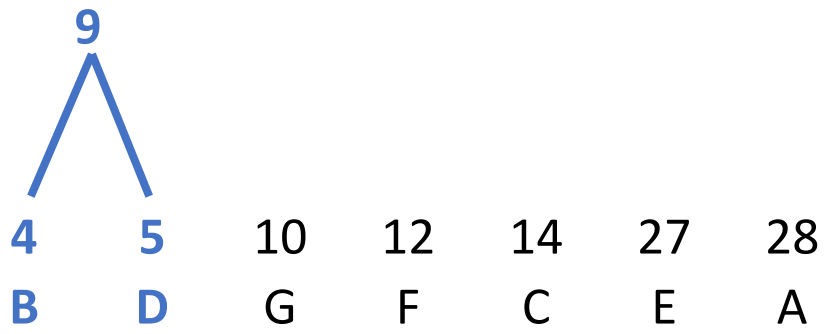
- But how do we assign codes to the cha e we know their frequencies?
  - By repeatedly selecting the two smallest weights and fusing them.

- This is **Huffman's algorithm** – another example of a **greedy method**.
  - The resulting tree is a **Huffman tree**.

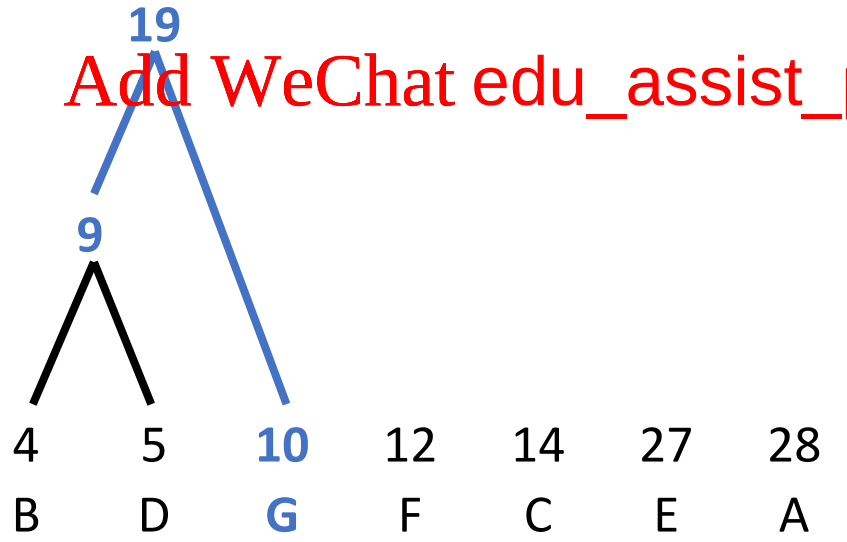
Huffman Trees (example)

4	5	10	12	14	27	28
B	D	G	F	C	E	A

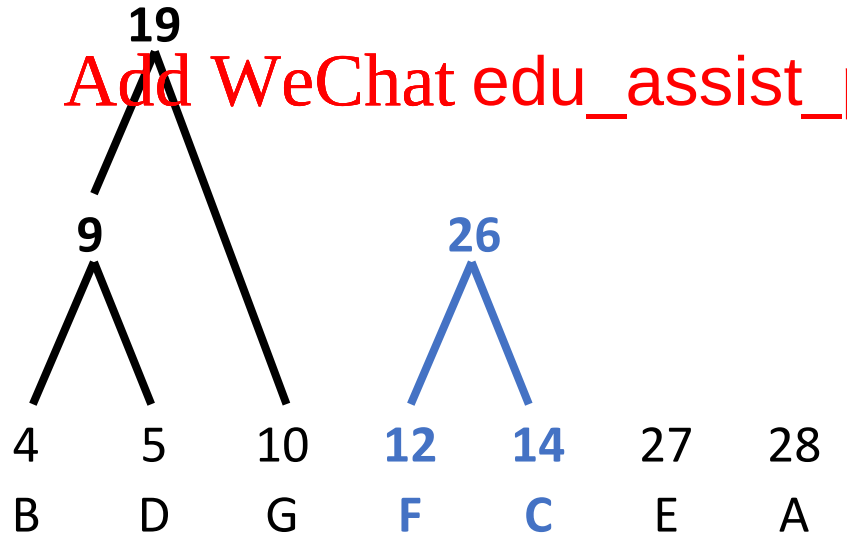
Huffman Trees (example)



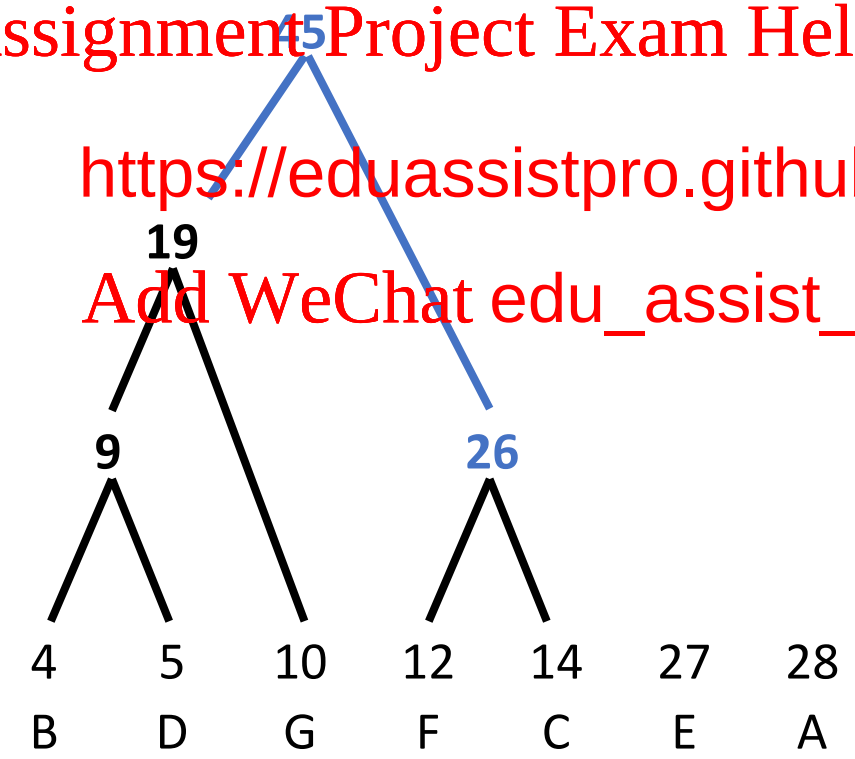
Huffman Trees (example)



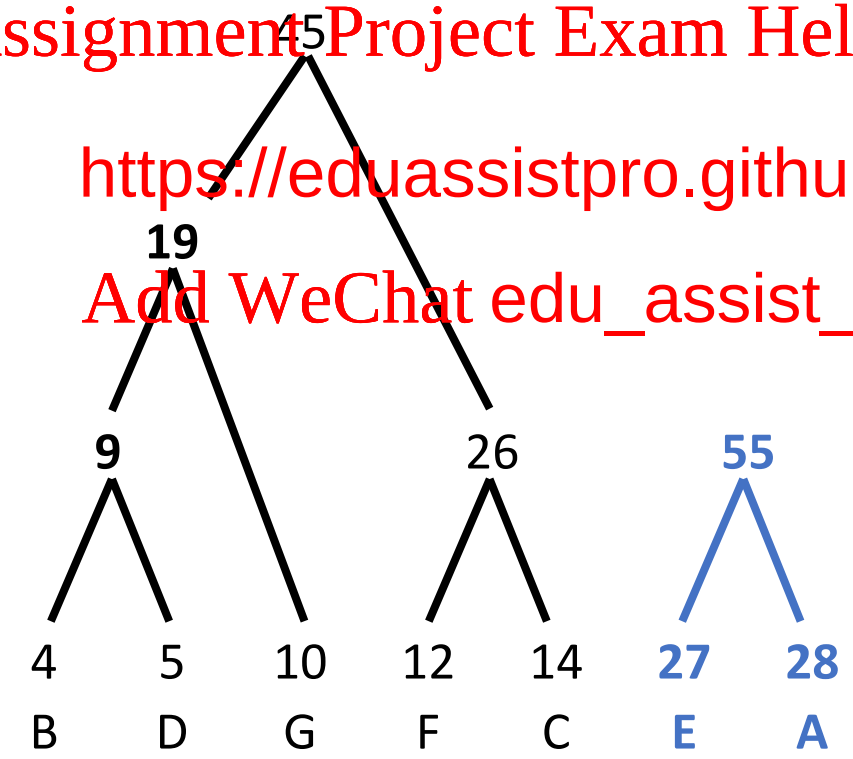
Huffman Trees (example)



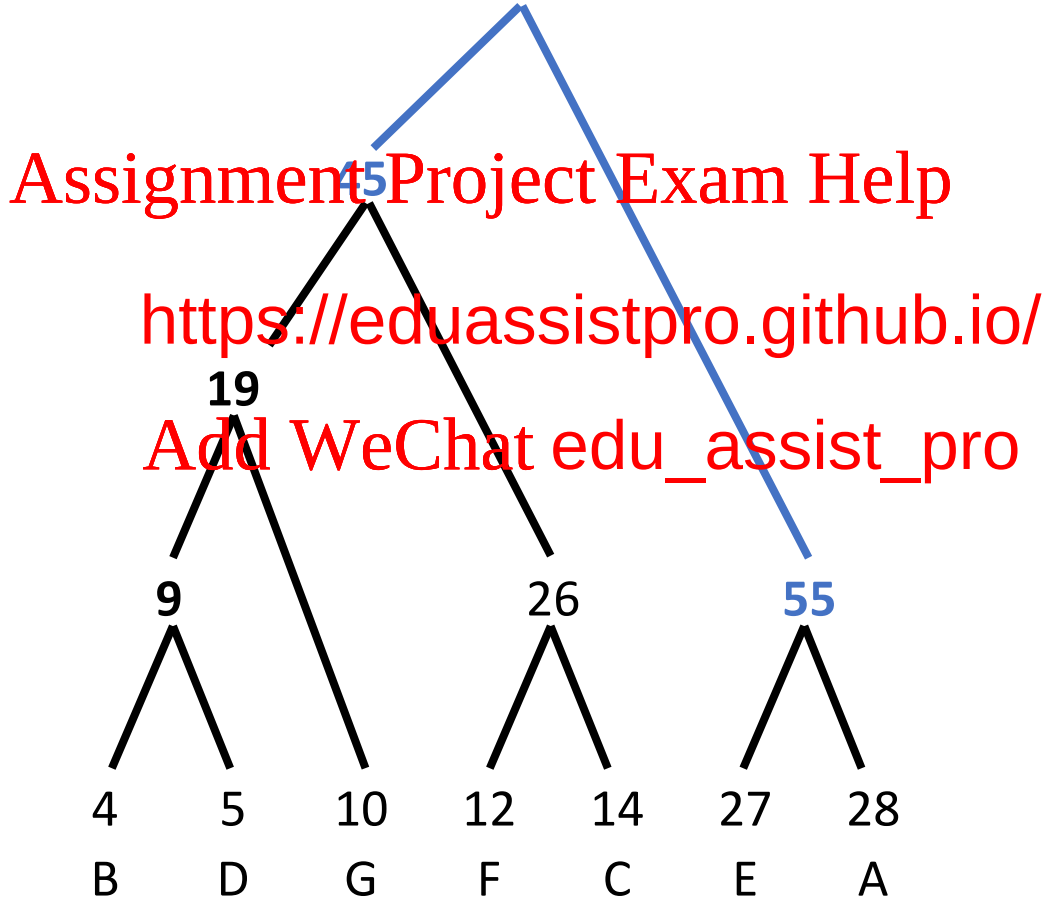
Huffman Trees (example)



Huffman Trees (example)

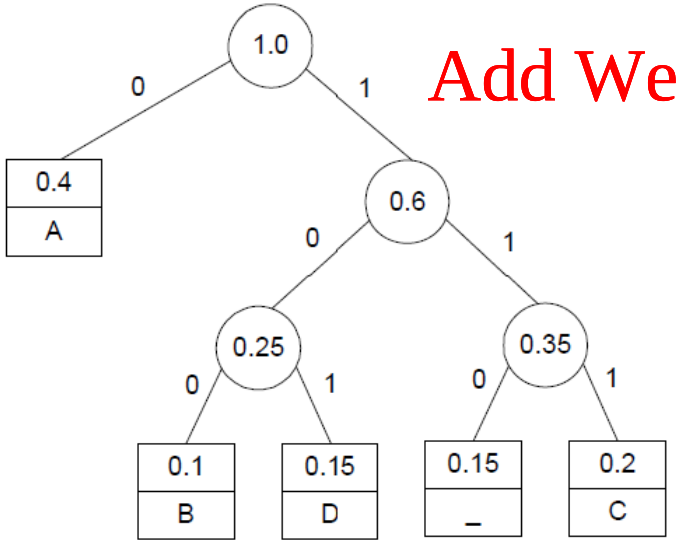


Huffman Trees (example)



An exercise

- Construct the Huffman code for data in the table, placing in the tree from left to right [A,B,C,D,_]
- Then, encode **ABACABAD** and decode **100010111001010**
- 0100011101000101 / BAD_A



SYMBOL	FEQUENCY	CODE
A	0.40	0
B	0.10	100
	0.20	111
	0.15	101
_	0.15	110

Compressed Transmission

- If the compressed file is being sent from one party to another, the parties must agree about the codes used.
  - For example, the trie can be sent along with the message.
- For long files this extra cost is negligible.
- Modern variants of Huffman encoding, like **Lempel-Ziv compression**, assign codes not to individual symbols but to sequences of symbols.

Next lecture

- We briefly discuss com pleteness and approximation algorit

- On the final week we will devote time to review all the content