COMP9319 Web Data Compression and Search

Week 7 Live Lecture
(Occ Implementation)

Agenda for today

- Some notices
- a1 feedback
- Implementations of C[] & Occ[]
- a2 questions

Notices

- a1 results **have been emailed** to you, see us in consultations if there are questions
- a2 due after next week, so next week's live lecture (and it's just next week, i.e., week 8) will be **ONLINE via Blackboard Collaborate** as a help session / Q&A for a2 – I'll post a reminder on WebCMS3 on Wed again.
- you **don't need to attend** the next week's live lecture if you have no questions on a2, I'll post more materials as pre-recorded videos for you.

a1 feedback

- Overall
  - full marks: ½ class
  - medium: 14, mean: 12
  - 0: <9, not submitted: <9
- Implementations
  - Trie, Hash, C++ dictionaries
- Marking
  - 13 sanity tests + 2 new tests
  - 6 encoding + 7 decoding (*encoding is overall slower*)
  - Decode given ~cs9319 files instead your encoded files
- Unhappiness
  - Sample/read some submitted code from each quartile of marks
  - Feedback

a2 feedback in wk10 (No late submissions)

The penalty for late submission of assignments will be 5% (of the worth of the assignment) subtracted from the raw mark per day of being late. In other words, earned marks will be lost. For example, assume an assignment worth 20 marks is marked as 18, but had been submitted two days late. The late penalty will be 2 marks, resulting in a mark of 16 being awarded. **No assignments will be accepted later than 5 days after the original deadline.** For example, if you have your special consideration granted by UNSW for a one-week extension, there will be no late penalty if the assignment is submitted within 7 days after the original deadline. However, no further late submissions will be accepted after these 7 days.

Refer to the Course Outline on WebCMS

FM Index (C[] and Occ[])

	F	L	c
0	#	i	0
1	i	p	0
2	i	s	0
3	i	s	1
4	i	m	0
5	m	#	0
6	p	p	1
7	p	i	1
8	s	s	2
9	s	s	3
10	s	i	2
11	s	i	3

C	
#	0
i	1
m	5
p	6
s	8

FM Index (need Occ for alphabets)

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

FM Index (backward search: pssi)

pssi

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

Fst=1
Lst=4

FM Index (con't)

pssi

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

Fst=8+0
Lst=(8+2)-1

FM Index (con't)

pssi

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

Fst=8+0
Lst=(8+2)-1

FM Index (con't)

pssi

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

Fst=8+2
Lst=(8+4)-1

FM Index (con't)

pssi

	F	L	i	m	p	s	C
0	#	i	1	0	0	0	# 0
1	i	p	1	0	1	0	i 1
2	i	s	1	0	1	1	m 5
3	i	s	1	0	1	2	p 6
4	i	m	1	1	1	2	s 8
5	m	#	1	1	1	2	
6	p	p	1	1	2	2	
7	p	i	2	1	2	2	
8	s	s	2	1	2	3	
9	s	s	2	1	2	4	
10	s	i	3	1	2	4	
11	s	i	4	1	2	4	

Fst=8+2
Lst=(8+4)-1

FM Index (con't)

		F	L	i	m	p	s	C
pssi	0	#	i	1	0	0	0	# 0
	1	i	p	1	0	1	0	i 1
	2	i	s	1	0	1	1	m 5
	3	i	s	1	0	1	2	p 6
	4	i	m	1	1	1	2	s 8
	5	m	#	1	1	1	2	
	6	p	p	1	1	2	2	
	7	p	i	2	1	2	2	
	8	s	s	2	1	2	3	
	9	s	s	2	1	2	4	Fst=6+2
⇨	10	s	i	3	1	2	4	Lst=(6+2)-1
⇨	11	s	i	4	1	2	4	

FM Index (con't)

		F	L	i	m	p	s	C
pssi	0	#	i	1	0	0	0	# 0
	1	i	p	1	0	1	0	i 1
	2	i	s	1	0	1	1	m 5
	3	i	s	1	0	1	2	p 6
	4	i	m	1	1	1	2	s 8
	5	m	#	1	1	1	2	
	6	p	p	1	1	2	2	
	7	p	i	2	1	2	2	
	8	s	s	2	1	2	3	
	9	s	s	2	1	2	4	Fst=6+2
⇨	10	s	i	3	1	2	4	Lst=(6+2)-1
⇨	11	s	i	4	1	2	4	

Fst > Lst => No match

FM Index (con't)

		F	L	i	m	p	s	C
pssi	0	#	i					# 0
	1	i	p					i 1
	2	i	s					m 5
	3	i	s	1	0	1	2	p 6
	4	i	m					s 8
	5	m	#					
	6	p	p					
	7	p	i	2	1	2	2	
	8	s	s					
	9	s	s					
	10	s	i					To reduce space
	11	s	i	4	1	2	4	

FM Index (con't)

		F	L	i	m	p	s	C
pssi	0	#	i					# 0
	1	i	p					i 1
	2	i	s					m 5
	3	i	s	1	0	1	2	p 6
	4	i	m					s 8
	5	m	#					
	6	p	p					
	7	p	i	2	1	2	2	
	8	s	s					
	9	s	s					
	10	s	i					To reduce space
	11	s	i	4	1	2	4	

Memory

- Don't call unnecessary functions / use unnecessary libraries
- Don't allocate too much
- Release them when they're not needed

Speed

- For big files, file I/Os dominate the time
- Ideally, max 1-2 reads per decoding/search term char (though a file block may be re-read many times)
- Minimize #reads vs the size per read

COMP9319 2023T2 Assignment 2: BWT Backward Search (Run-length Encoded)

Your task in this assignment is to create a search program that implements BWT backward search, which can efficiently search a run-length encoded and BWT transformed (RLB) record file without decoding the file back to a larger form. The original record file as plain text file (before BWT) format is:

`[<offset1>]<text1>[<offset2>]<text2>[<offset3>]<text3>... ...`

where `<offset1>`, `<offset2>`, `<offset3>`, etc. are integer values that are used as unique record identifiers (increasing and consecutive, positive integers, not necessarily starting from 0 or 1);
and `<text1>`, `<text2>`, `<text3>`, etc. are text values, which include any visible ASCII alphabets (i.e., any character with ASCII value from 32 to 126 inclusively), tab (ASCII 9) and newline (ASCII 10 and 13). For simplicity, there will be no open or close square bracket in the text values.