CSCI 4118/6105 Lab 2 (Part 2):

Succinct Data Structures

Winter 2022

Objective

1. Understanding the use of succinct data structure for massive data.
2. Design and implementation of Burrows-Wheeler Transform and FM-index.

Pre-read/Terminologies

1. Succinct Data Structures.
2. Burrows-Wheeler Transform, LF-Mapping, and FM-Index.

Resources

- For submission: https://git.cs.dal.ca/courses/2022-winter/csci-4118-6105/lab2/???? where ???? is your CSID
- sdsl-lite Software [1]: https://github.com/simongog/sdsl-lite

Procedure

Note:

- Undergraduate students are encouraged to attempt additional questions meant for graduate students and get bonus points.
- The sdsl-lite software should be cloned in a different folder, not the same folder as the lab2 repository.

1. Download and install the sdsl-lite library:
   If you haven't already, you will have to install *cmake*; reference: cmake installation [2] [3]
   Windows users: The installation script: is *install.bat*

```
git clone https://github.com/simongog/sdsl-lite.git
cd sdsl-lite
./install.sh
```

2. Compile the FM-Index [4] example:

```
g++ -std=c++11 -O3 -DNDEBUG -I ~/include -L ~/lib examples/fm-index.cpp -o
program -lsdsl -ldivsufsort -ldivsufsort64
```

3. Run the program (where sdsl_input.txt is the input text file; present in *"lab2"* repository):

```
./program sdsl_input.txt
```

4. Compare the file size of the input with the compressed index size and Identify patterns in the input text and their co-relation to the index size. Example: Repeated words/characters (cardinality), input size, etc.

Questions

1. **All Students:** For step #4 in the procedure, report your findings (include the input, index size, file size, cardinality, etc)

2. **Under-graduate students:**
   Write a program to get the original string from: BWT (Burrows-Wheeler Transform) and the index of the original string in BWM (Burrows-Wheeler Matrix).
   How do we do it?
   - Assuming BWT and index are already given.
   - Sort BWT string in lexicographic order
   - Concat BWT with the sorted string vertically as shown in *Figure 2*
   - Repeat until the length of the BWT string
   - The string at the given index would be the original string.

   Note: The original string has to be generated from BWT, and the index of the original string in BWM (BWM is not given, only the index)

3. **Graduate students:**
   Write a program to find the number of occurrences of a substring in a text using FM-Index (Easy Implementation).
   How do we do it?
   a. BWT (Burrows-Wheeler Transform)
      - All rotations.
      - Sort in Lexical-order
      - Last Column is the BWT
   b. LF-Mapping
      - Tally from L for all characters.
   c. Count number of occurrences:
      - Search the last character of the sub-string in F.
      - Among those Fs, find the (last-1) character in corresponding Ls of appropriate ranks.
      - Among those Fs, find the character with the same rank in Fs
      - Repeat for the length of the sub-string and return the count of occurrences.

   Refer to *Figure 2* on how to get the original string (A series of sort and concatenation operations) and Figure 3 on how to get the number of occurrences of a sub-string using LF-Mapping and ranks - In the explanation section below.

Explanation

Visual representation of FM-Index working to find the number of occurrences.

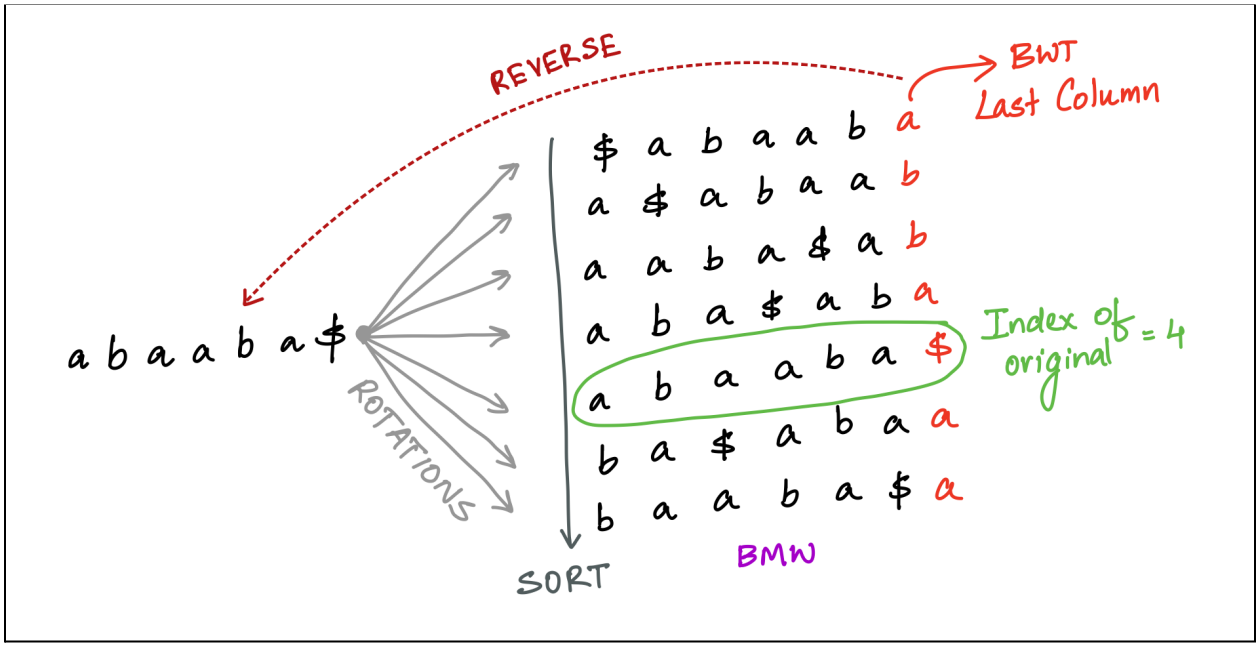

Figure 1: Generating BWM and BWT for the given input.

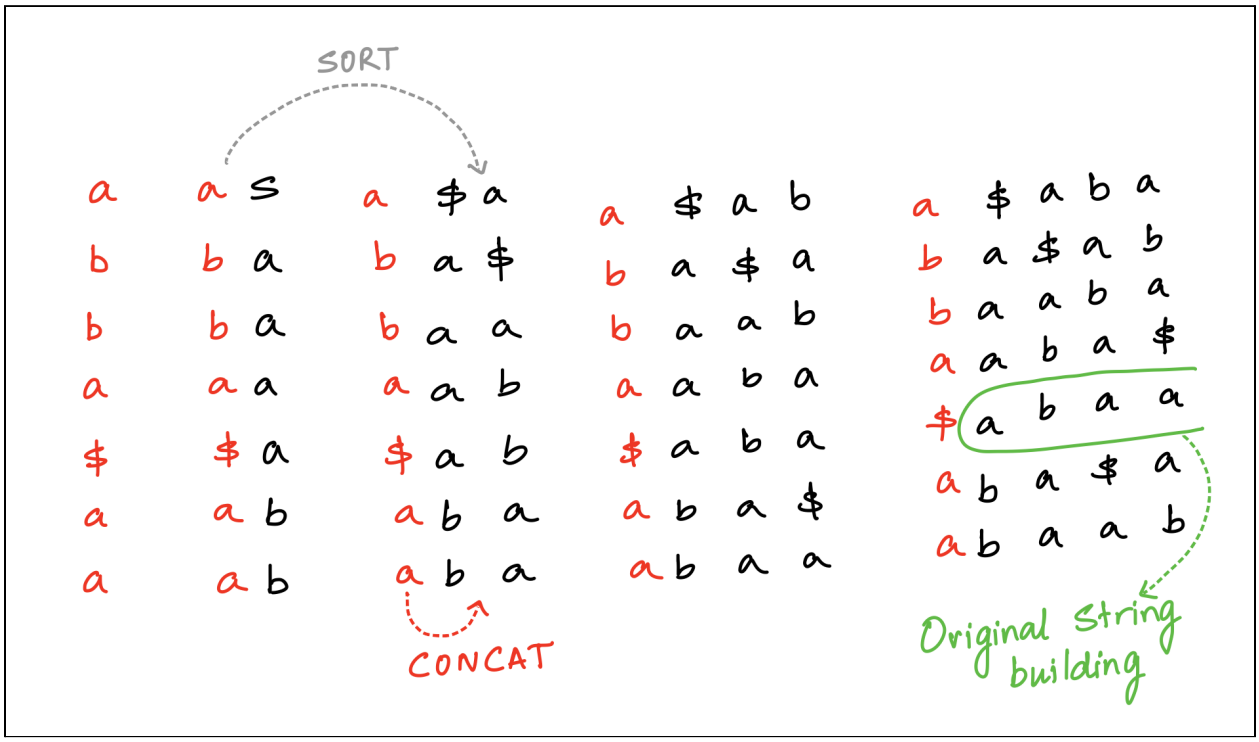

Figure 2: Getting original string from BWT and index in BWM.

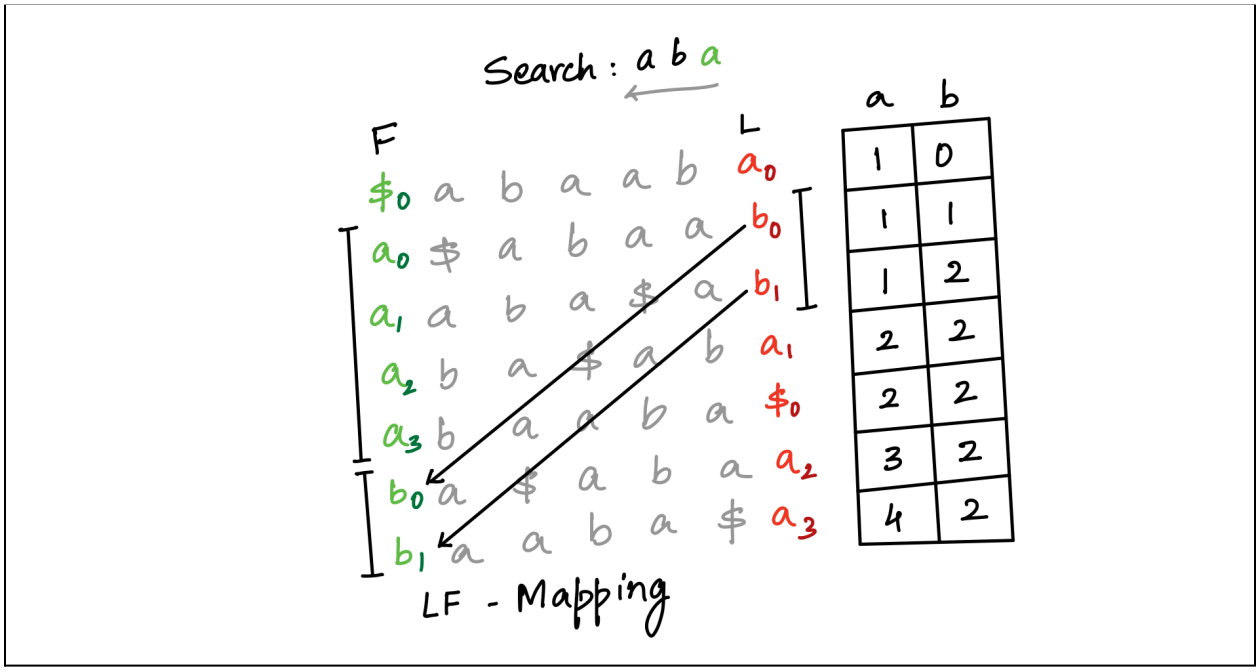

Figure 3: Finding the number of occurrences using LF-Mapping (FM-Index).

Submission

Note: For submission - git add, commit and push the answers to the lab2/???? repository and verify the submission in the GitLab web interface.

1. **All Students:** Answer Q1 (Refer to "questions" section above) in *"questions_part_2.txt"*

2. **Under-graduate Students:** Complete the implementation of the method "reverse" in "BWT.java" to get the original string (Q2 in "questions" section).

3. **Graduate Students:** Create a new method in the "BWT.java" class to find the number of occurrences in the input text (Q3 in the "questions" section).

Note:

- The program to generate BWM and BWT is already present in the code-base (For Q2 and Q3).
- You are free to use any programming language. However, you will have to re-write some of the existing code available in the repository for reference if necessary.
- To be explicit, Q1 is all for students, Q2 is only for undergraduate students, and Q3 is only for graduate students. However, all students can try all questions to get bonus points.

Grading

Task	5 Points (x3)	3 Point	0 Points
1	Q1 Thoroughly tested and reported findings in "*questions_part_2.txt*".	Q1 tested partially and reported findings in "*questions_part_2.txt*".	No evidence of testing.
2	Q2 program implemented and tested (works for different inputs).	Algorithm implemented, logically correct (pseudo-code), and/or evidence for testing and explanation.	Incorrect answer or not answered.
3	Q3 program implemented and tested (works for different inputs).	Algorithm implemented, logically correct (pseudo-code), and/or evidence for testing and explanation.	Incorrect answer or not answered.

Grading for Under-graduate Students:

Part	**Task 1**	**Task 3**	**Task 3**	**Grade**
Part 1	3	2	-	5
Part 2	5	5	-	10
Total				15

Grading for Graduate Students:

Part	**Task 1**	**Task 3**	**Task 3**	**Grade**
Part 1	3	2	5	10
Part 2	5	-	5	10
Total				20

However, under-graduate students can get an additional +1% for attempting graduate questions (3 points for both parts).

References

[1] S. Gog, “simongog/sdsl-lite,” GitHub, Apr. 02, 2021. https://github.com/simongog/sdsl-lite.

[2] “Download CMake.” https://cmake.org/download/ (accessed Feb. 04, 2022).

[3] “2.1. CMake Installation — CGold 0.1 documentation,” cgold.readthedocs.io. https://cgold.readthedocs.io/en/latest/first-step/installation.html (accessed Feb. 04, 2022).

[4] P. Ferragina and G. Manzini, “Opportunistic data structures with applications,” IEEE Xplore, Nov. 01, 2000. https://ieeexplore.ieee.org/abstract/document/892127 (accessed Feb 04, 2021).