Applying Concurrency Technique using Golang in Multiple Approximate Pattern Matching Problem with Burrows-Wheeler Transform

Ta Van Nhan   Nguyen Thi Hong Minh

tavannhan@gmail.com   minhnth@gmail.com

Department of Mathematics, Mechanics, and Informatics
VNU University of Science, Hanoi, Vietnam

November 5, 2020

Outline

Introduction

- Matching Problem

Preliminary

- Suffix Arrays
- Burrows-Wheeler Transform

Method

- Exact and Approximate Pattern Matching
- Applying Golang to Approximate Pattern Matching
- Advantages of Golang

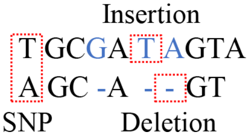
Implementation

- Data
- Results

Introduction

- Problem: Matching **sequences** to a **reference genome**.
- The purpose is to find small genetic alterations such as:

$$\begin{cases}\text{SNPs}\\ \text{Indels}\end{cases}$$



Insertion

T GCGA T AGTA

A GC -A - - GT

SNP Deletion

Figure 2: Alignment of TGCGATAGTA and AGCAGT.

Preliminary

Burrows-Wheeler Transform, Partial Suffix Arrays, Checkpoint Arrays.

BWTM	Index	SA
$ATCATGAT**C**	0	9
ATC$ATCAT**G**	1	6
ATCATGATC**$**	2	0
ATGATC$AT**C**	3	3
C$ATCATGA**T**	4	8
CATGATC$A**T**	5	2
GATC$ATCA**T**	6	5
TC$ATCATG**A**	7	7
TCATGATC$**A**	8	1
TGATC$ATC**A**	9	4

Figure 3: SA, BWT of ATCATGATC$.

Index	BWT	A	T	C	G
0	C	0	0	1	0
1	G	0	0	1	1
2	$	0	0	1	1
3	C	0	0	2	1
4	T	0	1	2	1
5	T	0	2	2	1
6	T	0	3	2	1
7	A	1	3	2	1
8	A	2	3	2	1
9	A	3	3	2	1

Figure 4: Checkpoint Arrays.

Burrows-Wheeler Transform

- Constructing the Burrows-Wheeler Transform *BWT* of the string T with linear time [1].
- Denote BWT_i, T_i by the i^{th} symbols of *BWT* and T respectively, SA_i is the value at i^{th} position of *Suffix Arrays*, one have:

$$BWT_i = \begin{cases} T_{SA_i - 1} & \text{nu } SA_i > 0 \\ \$ & \text{nu } SA_i = 0 \end{cases}$$

[1] D. Okanohara and K. Sadakane, *A Linear-Time Burrows-WheelerTransform Using Induced Sorting*. Aug. 2009, vol. 5721, p. 101.

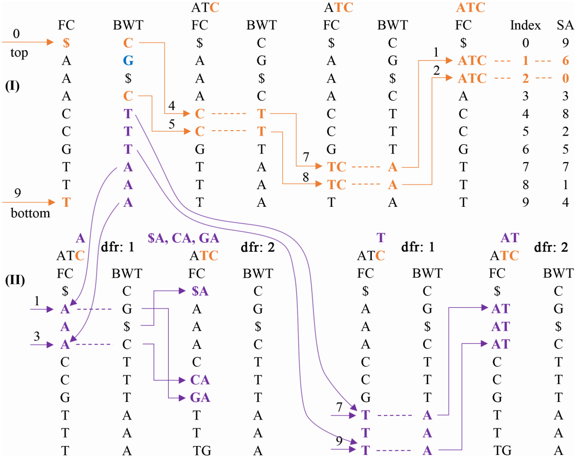
Method

- P. Ferragina and G. Manzini (2005) introduced a backward search algorithm that counts the occurrences of a pattern P on the string T with $\mathcal{O}(|P| + |T|)$ runtime [2].
- From an interval in the column BWT, one could find a new interval $[top, bottom]$ in the column FC, the intervals are updated as follows:

$$top \leftarrow FO(symbol) + CO(symbol, top - 1)$$
$$bottom \leftarrow FO(symbol) + CO(symbol, bottom) - 1$$

[2] P. Ferragina and G. Manzini, *"Indexing compressed text,"* Journal ofthe ACM, vol. 52, no. 4, pp. 552–581, Jul. 2005,issn: 0004-5411.

Approximate Pattern Matching



(I)

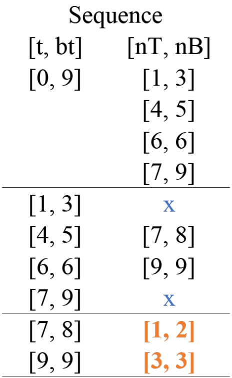
		ATC		ATC		ATC		
FC	BWT	FC	BWT	FC	BWT	FC	Index	SA
$	C	$	C	$	C	$	0	9
A	G	A	G	A	G	ATC	1	6
A	$	A	$	A	$	ATC	2	0
A	C	A	C	A	C	A	3	3
C	T	C	T	C	T	C	4	8
C	T	C	T	C	T	C	5	2
G	T	G	T	G	T	G	6	5
T	A	T	A	TC	A	T	7	7
T	A	T	A	TC	A	T	8	1
T	A	T	A	T	A	T	9	4

(II)

A ATC	dfr: 1	$A, CA, GA ATC	dfr: 2	T ATC	dfr: 1	AT ATC	dfr: 2
FC	BWT	FC	BWT	FC	BWT	FC	BWT
$	C	$A	C	$	C	$	C
A	G	A	G	A	G	AT	G
A	$	A	$	A	$	AT	$
A	C	A	C	A	C	AT	C
C	T	C	T	C	T	C	T
C	T	CA	T	C	T	C	T
G	T	GA	T	G	T	G	T
T	A	T	A	T	A	T	A
T	A	T	A	T	A	T	A
T	A	TG	A	T	A	TG	A

Figure 5: The backward search algorithm for ATC and ATCATGATC.

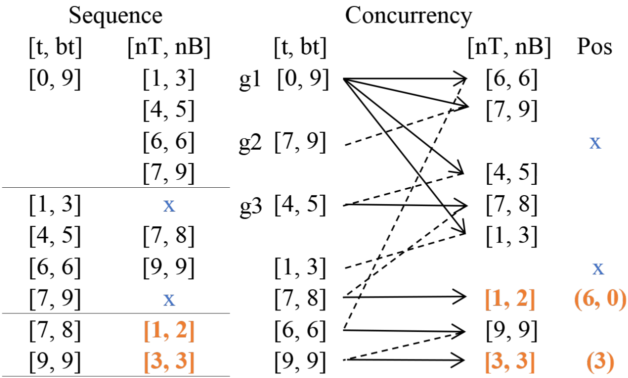
Applying Golang to Approximate Pattern Matching



Sequence	
[t, bt]	[nT, nB]
[0, 9]	[1, 3]
	[4, 5]
	[6, 6]
	[7, 9]
[1, 3]	x
[4, 5]	[7, 8]
[6, 6]	[9, 9]
[7, 9]	x
[7, 8]	**[1, 2]**
[9, 9]	**[3, 3]**

Figure 6: Considering the process of approximate matching of the pattern ATC to the string ATCATGATC with difference threshold 1.

Concurrency Algorithm



Sequence

[t, bt]	[nT, nB]
[0, 9]	[1, 3]
	[4, 5]
	[6, 6]
	[7, 9]
[1, 3]	x
[4, 5]	[7, 8]
[6, 6]	[9, 9]
[7, 9]	x
[7, 8]	**[1, 2]**
[9, 9]	**[3, 3]**

Figure 7: To find new intervals, goroutines work together and could start at the same time or at different times.

Advantages of Golang

- Golang with concurrency technique automatically exploits the operation of the computer's core without depending on the management and allocation of the operating system.
  - The threads on the cores are implemented by goroutines.
  - Each goroutine only uses very little memory from the heap (only about 2kB), which makes it possible to generate lots of goroutines at the same time.
  - Goroutines can work on logical processors at exactly the same time or can wait for each other in a queue.
  - Goroutines operate on the principle of none-sharing variables. Communication of goroutines synchronized by using buffered channels or unbuffered channel.

Implementation

Data

- The genome assembly of SARS-CoV-2 published by Fan Wu et al. (2020) [1], which is 24748 bp long was used as the reference genome for alignment.
- The raw sequences of SARS-CoV-2 published on July 28, 2020 by KwaZulu-Natal Research Innovation and Sequencing Platform from the Sequence Read Archive (SRA). The FASTQ file includes 436,610 paired-end reads [2].

Changing steps *c* in *Suffix Arrays* and *k* in *Checkpoint Arrays* to evaluate the variation level between runtime and memory used.

[1] https://www.ncbi.nlm.nih.gov/nuccore/1798174254

[2] https://sra-pub-sars-cov2.s3.amazonaws.com/sra-src/SRR12338312/KPCOVID-345_S81_L001_R1_001.fastq.gz.1

Results

Table 1: The Sequence Alignment Results with Three Differences

Name	Direction	Location	Mismatch String
100062/2	16	13030	0C0C0A248
100104/1	0	14275	10G68T52T50
100160/1	0	9679	1C0A0C88
100160/2	16	9679	1C0A0C88
100223/2	16	14314	0C67T24T157
100269/1	0	14358	49T51G55G67
10027/2	16	13128	0C0A24C224
100473/1	0	14219	31A26A64C53
1005/1	0	17162	57A78C35T12
100727/2	16	14477	66A8C27A2

Results (cont.)

Table 2: Total of Matches with Difference Thresholds

D	Time (s)	Memory (MiB)	Total
0	178.56	96	242943
1	234.31	270	354930
2	436.03	268	365724
3	1229.37	357	367946

Table 3: Approximate Matching Algorithm with Different Parameters

	c, k			
	1	*30*	*60*	*100*
Time (s)	234.31	1337.09	1361.23	1393.58
Memory (MiB)	270	274	254	131

Thanks for watching!