

CPSC 474 -01

Dr. D. Bein

Project 1 Submission

PSEUDOCODE FOR OUR PROGRAM

1) Pseudocode for main.cpp

- a) Create a string event matrix of events as a 2D array
- b) Create three matrices with LC-values as 2D arrays
- c) Call algorithm calculate using the matrix of events
- d) Call algorithm verify using each of the three LC-value matrices
- e) Return

2) Pseudocode for header.h

- a) Specify the number of columns and rows for all 2D arrays

3) Pseudocode for algorithmVerify.h (example 2, 3, 4).

- a) Pass in LC-values as a 2D array
- b) Create a string event matrix of events as a 2D array
- c) While we're checking to see if our LC-values are possible
 - a. For every column
 - i. For every row
 1. If we're the next number in the sequence
 - a. Increment sequence
 - b. We're still adding values into event matrix
 2. If our current sequence number is larger than max
 - a. Set max equal to this sequence number

- d) If sequence is not equal to max
 - a. Output that the sequence is incorrect
 - b. Return with error code -1
- e) Initialize the event matrix equal to NULL
- f) For every row
 - a. Reset previous element number
 - b. For every column
 - i. If previous element number was just reset
 - 1. If our current index is not sequential with previous element number
 - a. Mark index as a receiver
 - 2. Else
 - a. if we're in column one but our event sequence is not 1
 - i. Mark index as a receiver
 - 3. Set previous as the current index's LC-value
- g) While we're still updating senders/receivers
 - a. Keep track of the receiver number we're on
 - b. Reset sender's logic clock
 - c. For every row and column
 - i. If we're currently the smallest receiver that we haven't yet found the sender for
 - 1. Update the sender's logic clock
 - 2. Set receiver's index
 - d. If we have the index for the receiver

- i. For every row and column where we haven't found the sender yet
 - 1. If the current index is the sender
 - a. Set the sender number, receiver number
 - b. Mark that we've found the sender
 - e. Unflag that we've found the sender
- h) For every row and column
 - a. If we shouldn't be NULL and aren't marked as a sender or receiver
 - i. Give the index a letter value and increment letter
- i) Output the entire event matrix
- j) Return

Output:

Algorithm Verify (1)

a	s1	r3	b
c	r2	s3	NULL
r1	d	s2	e

Algorithm Verify (2)

s1	a	r3	b
c	r2	s3	NULL
r1	d	e	s2

Algorithm Verify (3)

The output is "INCORRECT".

4) Pseudocode for *algorithmcalculate.h* (example 1)

- a) Pass in matrix of events
- b) Create LC-value matrix
- c) Initialize logicClocks to 0
- d) Initialize an array that keeps track of each row's response point
- e) While we're still making changes
 - a. For each row
 - i. For each column
 - 1. If index is not a response point or past a response point that was already found
 - a. update logic clock based on index before
 - 2. Else if this is a response point
 - a. Update the location of the response point in this row
 - 3. If index is a send point
 - a. If there's a response point sender can reach
 - i. Set LC-value equal of response point to the maximum of it's sequential value or the sender's value plus one
 - ii. Reset the response point
 - 4. Set k equal to index's clock number
 - b. Reset k when we change the row
 - c. Check if we are done making changes
- f) Output the matrix with LC-values

Mason Godfrey (mgodfrey@csu.fullerton.edu)

Bijaya Shrestha (sthavjay@csu.fullerton.edu)

g) Return

Output:

```
Algorithm Calculate (1)
```

1	2	8	9
1	6	7	0
3	4	5	6