Distributed Computing

Programming Assignment – 1 Implementing Vector Clock

Harshit Patel
CS16BTECH11017

Comparison of the performance of vector-clocks and Singhal-Kshemkalyani optimization.

Vector Clock - Each process p_i maintains a vector $vt_i[1..n]$ where $vt_i[i]$ is the local logical clock of p_i and describes the logical time progress at process p_i . $vt_i[j]$ represents process p_i's latest knowledge of process p_j local time.

Rules -
Before executing an event, process p_i updates its local logical time as follows:
$vt_i[i] = vt_i[i] + d$ $\qquad$ $d = 1$, for our program

Each message m is piggybacked with the vector clock vt of the sender process at sending time. On the receipt of such a message (m,vt) :-
- p_i updates its global logical time
  $1 \leq k \leq n \quad vt_i[k] = \max(vt_i[k], vt_i[k])$
- executes above rule.

Therefore for each transfer of message from process p_i to p_j, p_i sends it entire vector clock to process p_j. Thus, the space required for transferring is O(n), for any single message.

The memory required at each process is O(n) to store its vector clock data.

Singhal–Kshemkalyani’s differential technique :-

This is an efficient implementation of vector clocks.

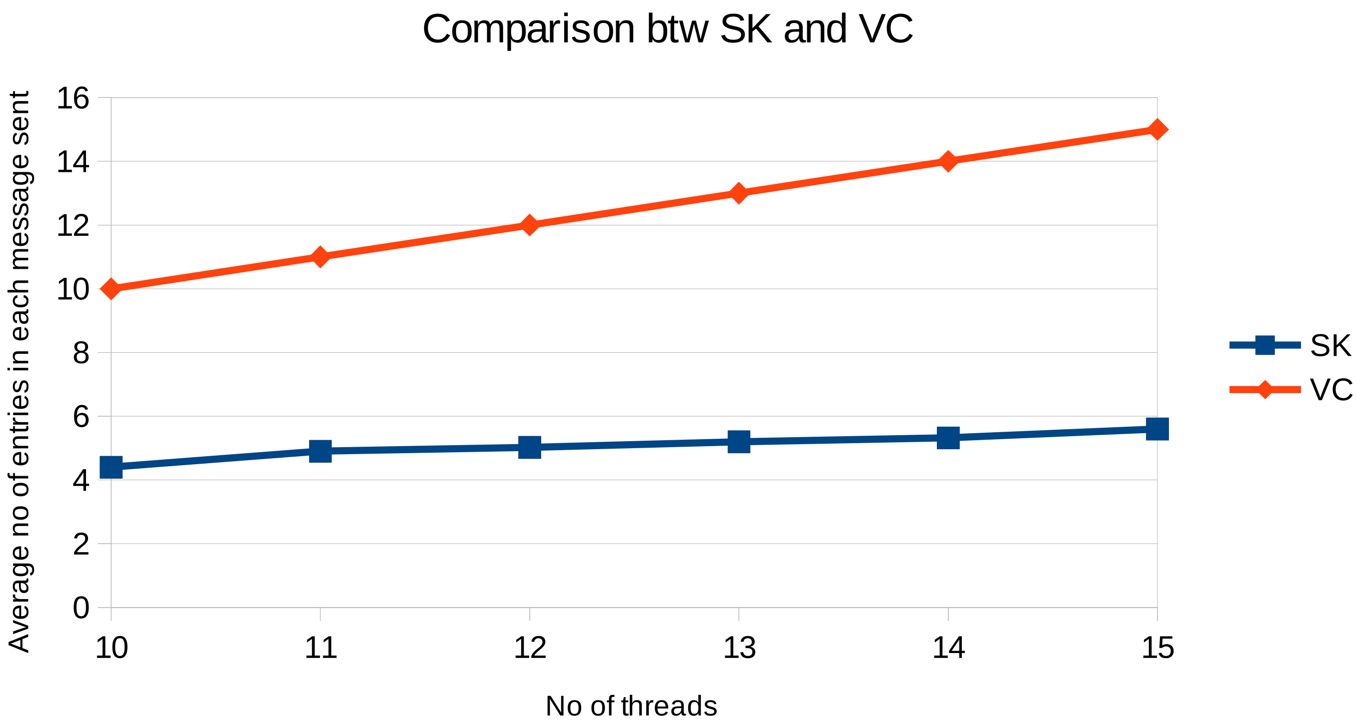
The technique works as follows: if entries i_1, i_2 .. i_{n1} of the vector clock at p_i have changed to v_1, v_2 .. v_{n1} , respectively, since the last message sent to p_j , then process p_i piggybacks a compressed timestamp of the form $\{ (i_1 , v_1), (i_2 , v_2) ... (i_{n1} , v_{n1})\}$ to the next message to p_j .

When p_j receives this message, it updates its vector clock as follows:
$vt_i[i_k] = \max(vt_i[i_k], v_k)$ for $k = 1,2 .. n_1$.

Thus this technique cuts down the message size, communication bandwidth and buffer requirements.
In the worst of case, every element of the vector clock has been updated at p_i since the last message to process p_j , and the next message from p_i to p_j will need to carry the entire vector timestamp of size n. However, on the average the size of the timestamp on a message will be less than n.

However in this technique we require 2 more array of LastSent and LastUpdate at each process inaddition to vector clock. Therefore, total storage space for each process is O(3n).

Graph comparison btw VC and SK optimization.



Analysis of the result

As we can see from the graph the average no of entries send in each message is lower for SK optimization as compared to VC.
This difference becomes more prominent for higher threads(i.e. more number of messages sent).
This difference is due to the observation that between successive message sends to the same process, only a few entries of the clock at the vector clock at the sender process are likely to change. Hence we only need to send the updated tuple.
Advantages – If the process interactions exhibit temporal and spatial localities , the cost of maintaining vector clocks in large systems can be significantly reduced by this technique. Though in this method we need to maintain 2 additonal vector clocks as compared to the earlier method.

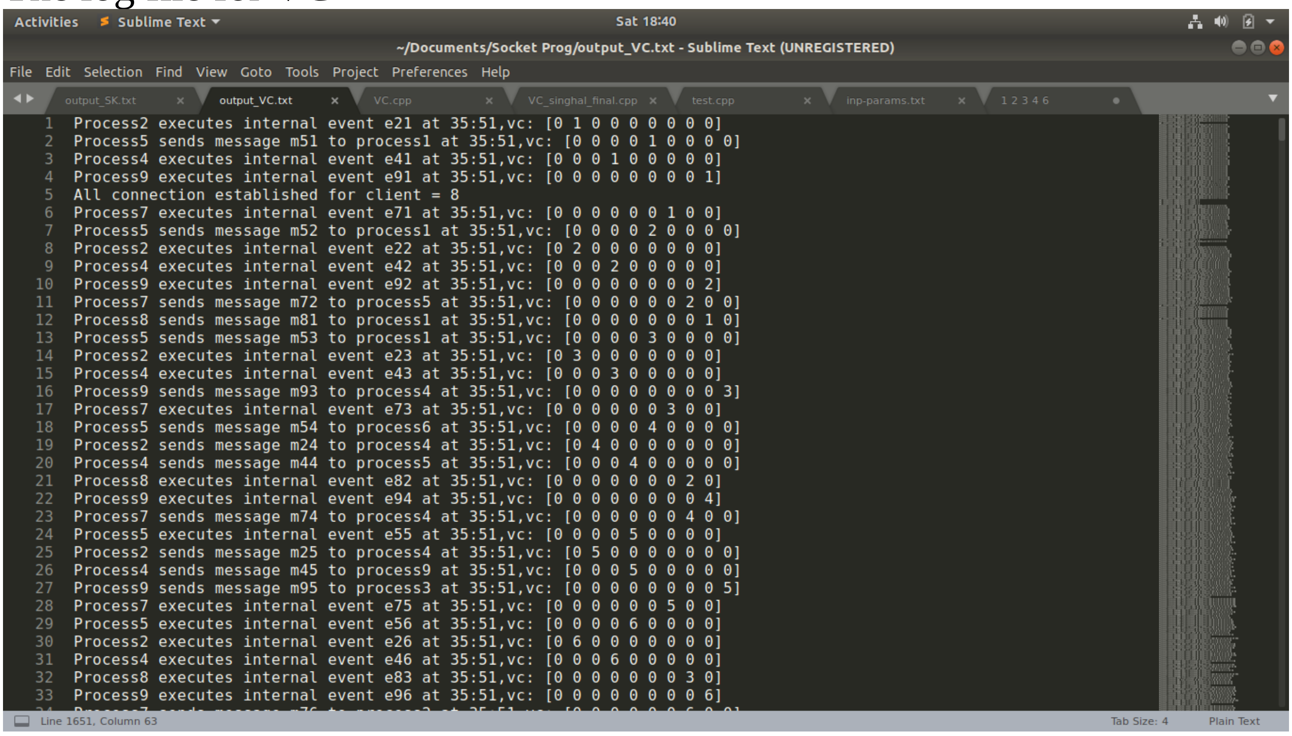
The space utilized by each process for storing the vector clocks and sending messages by both the algorithm is displayed at the end of log file. e.g.

For inp-params :-
9 20 1.5 50
1 2 3 4 6
2 1 9 4
3 1 2
4 2 5 7 8 9
5 4 6 8 1
6 2 4 1 9 7 5
7 1 2 4 5 3 9 8
8 2 9 1 3
9 1 4 3

Result by VC -
Total space for vector clocks storage in this method = 9 blocks
Total messages(vector clocks)send in this method = 4050 blocks

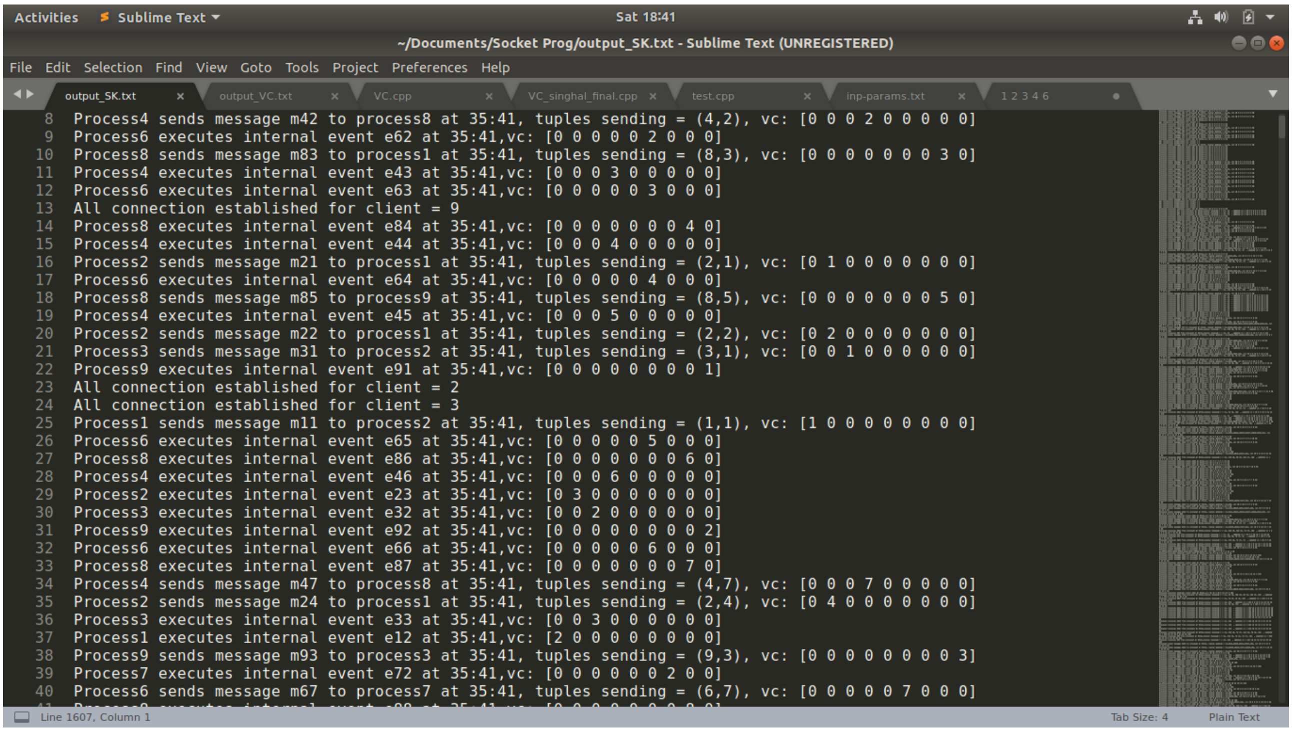
Result by SK-
Total space for vector clocks storage in this method = 27 blocks
Total tuples send in this method = 1569

The log file for VC -



```
Process2 executes internal event e21 at 35:51,vc: [0 1 0 0 0 0 0 0 0]
Process5 sends message m51 to process1 at 35:51,vc: [0 0 0 0 1 0 0 0 0]
Process4 executes internal event e41 at 35:51,vc: [0 0 0 1 0 0 0 0 0]
Process9 executes internal event e91 at 35:51,vc: [0 0 0 0 0 0 0 0 1]
All connection established for client = 8
Process7 executes internal event e71 at 35:51,vc: [0 0 0 0 0 0 1 0 0]
Process5 sends message m52 to process1 at 35:51,vc: [0 0 0 0 2 0 0 0 0]
Process2 executes internal event e22 at 35:51,vc: [0 2 0 0 0 0 0 0 0]
Process4 executes internal event e42 at 35:51,vc: [0 0 0 2 0 0 0 0 0]
Process9 executes internal event e92 at 35:51,vc: [0 0 0 0 0 0 0 0 2]
Process7 sends message m72 to process5 at 35:51,vc: [0 0 0 0 0 0 2 0 0]
Process8 sends message m81 to process1 at 35:51,vc: [0 0 0 0 0 0 0 1 0]
Process5 sends message m53 to process1 at 35:51,vc: [0 0 0 0 3 0 0 0 0]
Process2 executes internal event e23 at 35:51,vc: [0 3 0 0 0 0 0 0 0]
Process4 executes internal event e43 at 35:51,vc: [0 0 0 3 0 0 0 0 0]
Process9 sends message m93 to process4 at 35:51,vc: [0 0 0 0 0 0 0 0 3]
Process7 executes internal event e73 at 35:51,vc: [0 0 0 0 0 0 3 0 0]
Process5 sends message m54 to process6 at 35:51,vc: [0 0 0 0 4 0 0 0 0]
Process2 sends message m24 to process4 at 35:51,vc: [0 4 0 0 0 0 0 0 0]
Process4 sends message m44 to process5 at 35:51,vc: [0 0 0 4 0 0 0 0 0]
Process8 executes internal event e82 at 35:51,vc: [0 0 0 0 0 0 0 2 0]
Process9 executes internal event e94 at 35:51,vc: [0 0 0 0 0 0 0 0 4]
Process7 sends message m74 to process4 at 35:51,vc: [0 0 0 0 0 0 4 0 0]
Process5 executes internal event e55 at 35:51,vc: [0 0 0 0 5 0 0 0 0]
Process2 sends message m25 to process4 at 35:51,vc: [0 5 0 0 0 0 0 0 0]
Process4 sends message m45 to process9 at 35:51,vc: [0 0 0 5 0 0 0 0 0]
Process9 sends message m95 to process3 at 35:51,vc: [0 0 0 0 0 0 0 0 5]
Process7 executes internal event e75 at 35:51,vc: [0 0 0 0 0 0 5 0 0]
Process5 executes internal event e56 at 35:51,vc: [0 0 0 0 6 0 0 0 0]
Process2 executes internal event e26 at 35:51,vc: [0 6 0 0 0 0 0 0 0]
Process4 executes internal event e46 at 35:51,vc: [0 0 0 6 0 0 0 0 0]
Process8 executes internal event e83 at 35:51,vc: [0 0 0 0 0 0 0 3 0]
Process9 executes internal event e96 at 35:51,vc: [0 0 0 0 0 0 0 0 6]
```

The log file for SK -



Activities
Sublime Text
Sat 18:41
~/Documents/Socket Prog/output_SK.txt - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
output_SK.txt
output_VC.txt
VC.cpp
VC_singhal_final.cpp
test.cpp
inp-params.txt
1 2 3 4 6
Process4 sends message m42 to process8 at 35:41, tuples sending = (4,2), vc: [0 0 0 2 0 0 0 0 0]
Process6 executes internal event e62 at 35:41,vc: [0 0 0 0 0 2 0 0 0]
Process8 sends message m83 to process1 at 35:41, tuples sending = (8,3), vc: [0 0 0 0 0 0 0 3 0]
Process4 executes internal event e43 at 35:41,vc: [0 0 0 3 0 0 0 0 0]
Process6 executes internal event e63 at 35:41,vc: [0 0 0 0 0 3 0 0 0]
All connection established for client = 9
Process8 executes internal event e84 at 35:41,vc: [0 0 0 0 0 0 0 4 0]
Process4 executes internal event e44 at 35:41,vc: [0 0 0 4 0 0 0 0 0]
Process2 sends message m21 to process1 at 35:41, tuples sending = (2,1), vc: [0 1 0 0 0 0 0 0 0]
Process6 executes internal event e64 at 35:41,vc: [0 0 0 0 0 4 0 0 0]
Process8 sends message m85 to process9 at 35:41, tuples sending = (8,5), vc: [0 0 0 0 0 0 0 5 0]
Process4 executes internal event e45 at 35:41,vc: [0 0 0 5 0 0 0 0 0]
Process2 sends message m22 to process1 at 35:41, tuples sending = (2,2), vc: [0 2 0 0 0 0 0 0 0]
Process3 sends message m31 to process2 at 35:41, tuples sending = (3,1), vc: [0 0 1 0 0 0 0 0 0]
Process9 executes internal event e91 at 35:41,vc: [0 0 0 0 0 0 0 0 1]
All connection established for client = 2
All connection established for client = 3
Process1 sends message m11 to process2 at 35:41, tuples sending = (1,1), vc: [1 0 0 0 0 0 0 0 0]
Process6 executes internal event e65 at 35:41,vc: [0 0 0 0 0 5 0 0 0]
Process8 executes internal event e86 at 35:41,vc: [0 0 0 0 0 0 0 6 0]
Process4 executes internal event e46 at 35:41,vc: [0 0 0 6 0 0 0 0 0]
Process2 executes internal event e23 at 35:41,vc: [0 3 0 0 0 0 0 0 0]
Process3 executes internal event e32 at 35:41,vc: [0 0 2 0 0 0 0 0 0]
Process9 executes internal event e92 at 35:41,vc: [0 0 0 0 0 0 0 0 2]
Process6 executes internal event e66 at 35:41,vc: [0 0 0 0 0 6 0 0 0]
Process8 executes internal event e87 at 35:41,vc: [0 0 0 0 0 0 0 7 0]
Process4 sends message m47 to process8 at 35:41, tuples sending = (4,7), vc: [0 0 0 7 0 0 0 0 0]
Process2 sends message m24 to process1 at 35:41, tuples sending = (2,4), vc: [0 4 0 0 0 0 0 0 0]
Process3 executes internal event e33 at 35:41,vc: [0 0 3 0 0 0 0 0 0]
Process1 executes internal event e12 at 35:41,vc: [2 0 0 0 0 0 0 0 0]
Process9 sends message m93 to process3 at 35:41, tuples sending = (9,3), vc: [0 0 0 0 0 0 0 0 3]
Process7 executes internal event e72 at 35:41,vc: [0 0 0 0 0 0 2 0 0]
Process6 sends message m67 to process7 at 35:41, tuples sending = (6,7), vc: [0 0 0 0 0 7 0 0 0]
Line 1607, Column 1
Tab Size: 4
Plain Text