Read:

1. A.D. Kshemkalyani and A. Misra, The Bloom Clock to Characterize Causality in Distributed Systems, The 23rd International Conference on Network-Based Information Systems (NBiS-2020), pp. 269-279, Springer, 2020. https://www.cs.uic.edu/~ajayk/ext/NBiS2020.pdf
2. L. Ramabaja, The Bloom Clock, *CoRR*, vol. abs/1905.13064, 2019. [Online]. Available: http://arxiv.org/abs/1905.13064 (read but beware, it has many errors)

The references to equations below are with respect to Reference [1] above.

## Experiment

AIM

The goal of the project is to study the probability of false positives, accuracy, precision, and the false positive rate of the Bloom clock. You may use any programming language and its libraries. (For example, Python multiprocessing).

A POSSIBLE APPROACH

One way to study this is to simulate asynchronous message-passing among n processes. Each process is simulated by a thread and the events of each process (internal, send, and receive events) are also simulated. In the simulation, each process maintains the Vector Clock (VC) as well as the Bloom Clock (BC). A process generates internal and send events with a certain probability (a controllable parameter, which can also disallow internal events) and at a certain frequency (rate), say 1 event/ms. Initially, set probability of internal event to 0, thus, there are only send events which induce corresponding receive events. The events at process P_i get queued in the process queue Q_i along with the simulation time timestamp, which is processed by the simulating thread. If it is a send event, its destination P_j is chosen at random from among the other n-1 processes. A corresponding receive event, (along with the sender's VC and BC timestamps) and along with the simulation time timestamp is enqueued in Q_j and processed by the thread simulating P_j. The simulation time timestamp of a receive event, as chosen by the sender, can be set to the sum of the send event simulation time timestamp plus a very small delta.

The queue Q_i determines the schedule of events occurring at process P_i. The thread simulating process P_i dequeues events in simulation timestamp order, and simulates the Vector clock and Bloom clock updates for that event. In essence, you have to ensure fair scheduling; for example, you can implement round-robin execution among the threads.

In addition, each thread P_i maintains a count of the number of simulated events it has processed after dequeueing from the local queue Q_i. Also maintain a global sequence number (GSN) for each event based on its order of occurrence across all processes.

EXPERIMENTS

For selected values of *n* (number of processes), e.g., 100, 200, 300, (and 500, 1000 if your computer permits), for the following values of *m* (size of Bloom Clock): 0.1n, 0.2n, 0.3n, and *k* (number of hash functions): 2,3,4, run the simulation and record the VC values and BC values of each event.

1. Consider the event with GSN = 10n. Let this be event y. Let z be instantiated by event with GSN from event 10n+1 to n^2 + 10n. For different values of m and k, plot the probability of positives

($pr_p$) given by Equation (4) on the Y-axis, as a function of the event number (GSN) on the X-axis. You may consider each 10th event for z. Note, this graph may look like a scatter-plot.

2. Repeat for different values of n.
3. Repeat (1), this time plotting the probability of false positives ($pr_{fp} = (1-pr_p)pr_{\backslash\text{delta}(p)}$ ) on the Y-axis. Use different symbols for actual positives and for actual negatives.
4. Repeat (3) for different values of n.
5. Repeat (1), this time plotting the **actual** accuracy, precision, and false positive rate estimate *fpr* (in separate graphs and tables) given by Equation (6) on the Y-axis. Let z be the event from *10n+1* to *n^2 + 10n*, in steps of *k* or a small multiple of *k* or a small multiple of 10, or 100. For the n^2 events, there are (n^2)(n^2 - 1)/2 pairs by letting each event of the execution slice be y and each other event as z. To calculate the accuracy, precision and fpr, you will need to use the actual VC and BC of each event in the execution slice. You may have to plot/tabulate separate graphs for different *k*. For this step, it may be more convenient to tabulate your results.
6. Repeat (5) for different values of *n*.
7. **[EXTRA CREDIT]** Repeat steps (5) and (6), this time plotting/tabulating the **estimated** accuracy, precision, and false positive rate using only BCs of events in the execution slice, using Equations (10), (11), and (12).

Now vary the probability of internal event (vs. send event) from 0 to 0.5 (recall: baseline case was with no internal events), and to 0.9, for internal events. Repeat simulations all the above.

NOTE: For Items 1,2,3,4, for z, you may have to experiment with the upper bound from n^2 + 10n to a lower or higher value so that a meaningful trend in the graphs can be determined.

DELIVERABLES

Submit a detailed project report (PDF), typeset preferably using Latex. It is recommended that your plots be in Gnuplot or some other professional tool (not Microsoft) converted to PDF figures. Your report should be professionally prepared.

Document all the design choices you made in the project, and how you implemented the main procedures.

You may plot not just graphs, but other forms of charts such as bar charts and histograms and pie charts, as you think are useful, for the experimental results. If more meaningful, you may also tabulate data. **As there will be a lot of data in the results to be presented, it is very important to present meaningful data in the results.**

Analyze and explain the trends and observations you make about your data in all the above cases. For example:

- How do each of the metrics (**actual** and **estimated** Accuracy, Precision, False Positive Rate, and $pr_{fp}$) vary with changes in n, keeping m, k, probability(internal event) constant?
- How do each of the metrics vary with changes in m, keeping, n, k, probability(internal event) constant?
- How do each of the metrics vary with changes in k, keeping n, m, probability(internal event) constant?
- How do each of the metrics vary with changes in probability(internal event), keeping n, m, k constant?

- How do the above observations change as the size of the execution slice for which you collect timestamps (BC and VC) varies?
- **[EXTRA CREDIT]** How much of an approximation are equations (10), (11), and (12) to the actual values computed using Equation (6)?

Note: The scientific approach to taking readings in simulations/experiments is as follows. For each setting of the parameters, take the readings for several runs (for example, 3 or 5 runs) and report the average. If there is noticeable variation in the readings (for the identical setting of the parameters), in addition to the average, report the standard deviation also for each setting of the parameters.