Experiment on Cristian’s and Berkeley Time Synchronization Algorithms

- Rahul Sehgal

12/6/2007

Introduction:

Centralized Algorithm:

1) Cristian’s algorithm. (*timeserver*)

2) Berkeley algorithm. (*coordinator*)

Concept of Simulation Time

- Time changes on the basis of state change.
- Time always advances
- Time doesn’t increases by the fix amount

Cristian's Algorithm

- External clock synchronization method.
- A process is the *time server* in the system.
- External time source *(Coordinated Universal Time)* is used as reference for synchronizing computer clocks with real time.
  - UTC is an international standard.
  - Standard bodies which, disseminate UTC signal by radio, telephone and satellite. One of the body is Geostationary Operational Environmental Satellites. (GEOS)

Cristian's Algorithm … contd

- Other processes request for current time by sending request message (req_message) to *time server.*
- *time server* sends a reply by attaching current time with each reply message.
- Message suffers:
  - Transmission delay

Apparatus for Cristian's Algorithm :

- *time server* maintains a *message_queue* for arriving requests from other processes.
- A delay value is generated at *message_queue*, due to message queue delay (*states for which request was in queue*)
- *Other issues : clock skew and clock drift*
  - *clock skew :* The difference in time values of two clocks.
  - *clock drift :* The difference between speeds of two clocks.

Implementation for measuring performance….contd

- In *run()* step,
  - The messages are delivered in the increasing order of delay, to *time server*.
  - *timeserver* computes *delay_at_rqst_queue* (states for which the message was in queue)
  - *timeserver* sends a reply message to requesting process.
  - Requesting process receives the message and updates its time as follows:

    current time = Tserver + (T_1 – T_0) / 2,

  Tserver = server time returned by time server.

  In simulation engine T_1 – T_0 = *message_queue_delay*

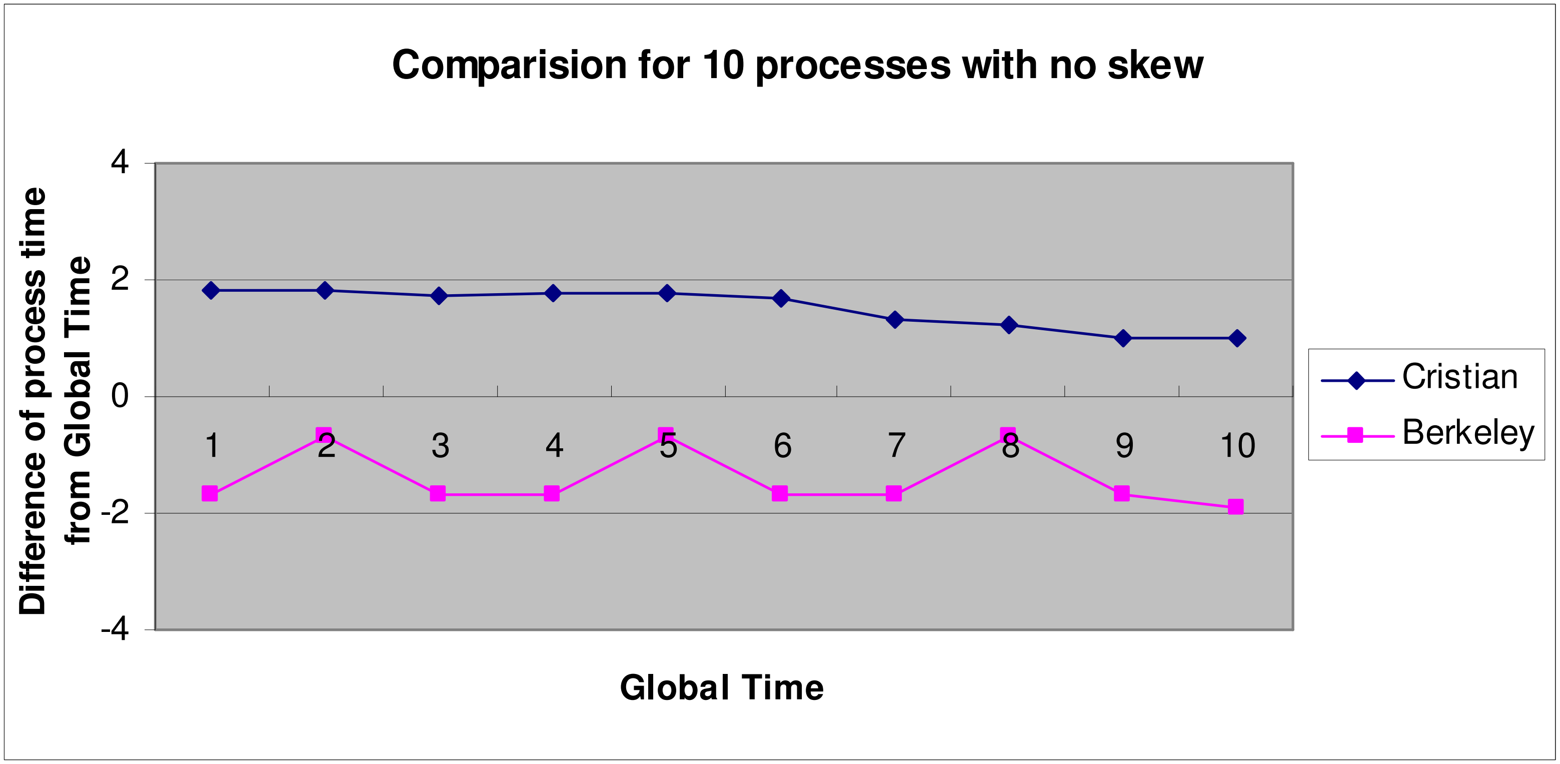
Berkeley Algorithm:

- *coordinator* polls other processes.
- Finds difference between its time and time of selected processes
- Then takes an average including its own time.
- Coordinator predicts time of other process with an error.
- Inform polled processes about correction (this message has error included)

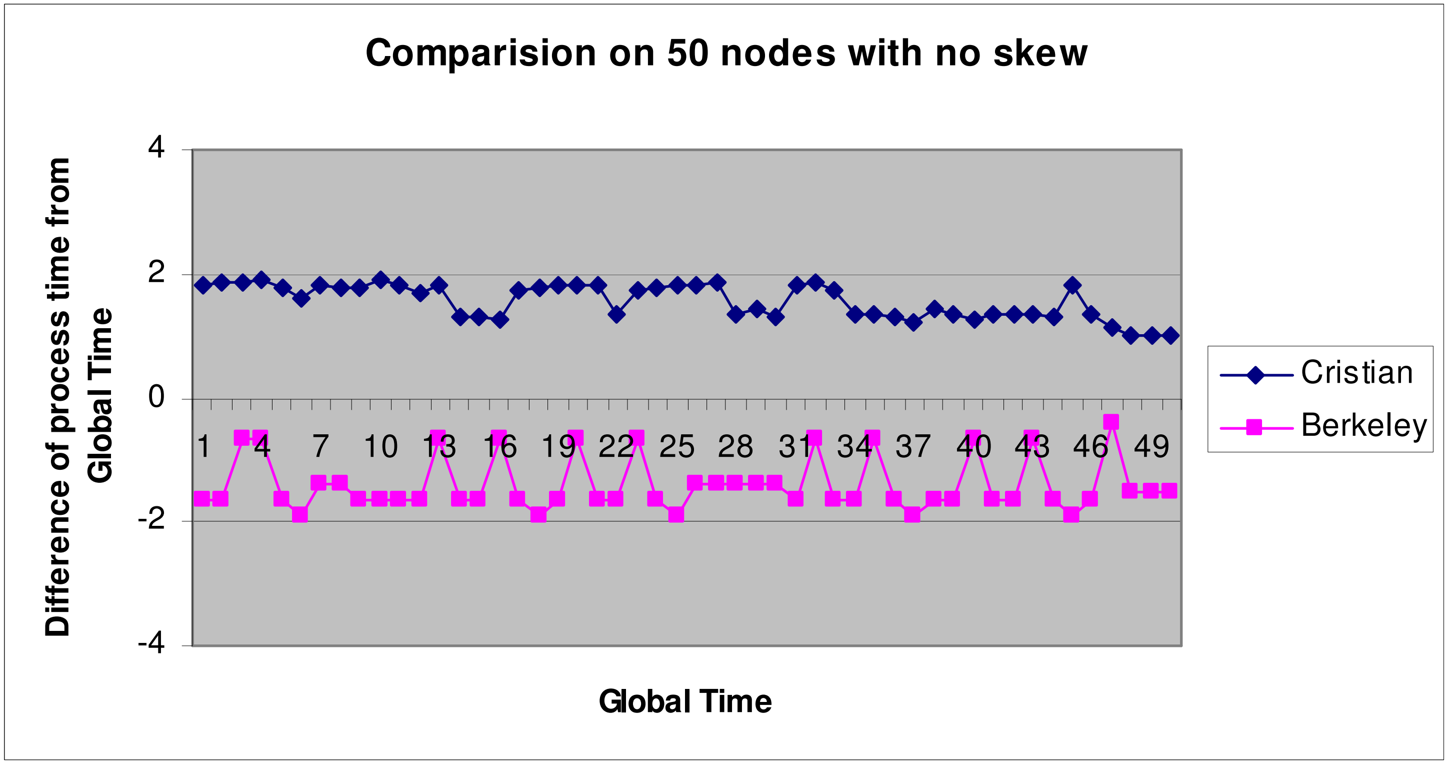
Apparatus:

- *coordinator* polls processes and sends them request for there time
- processes sends reply.
- processes differ in their times
- *Coordinator predicts* error for each process.
- *coordinator* puts an upper bound errors .
- Two test environments are:
  - *with minimal (or no) errors*
  - *with errors*

Comparison between Cristian and Berkeley Algorithms

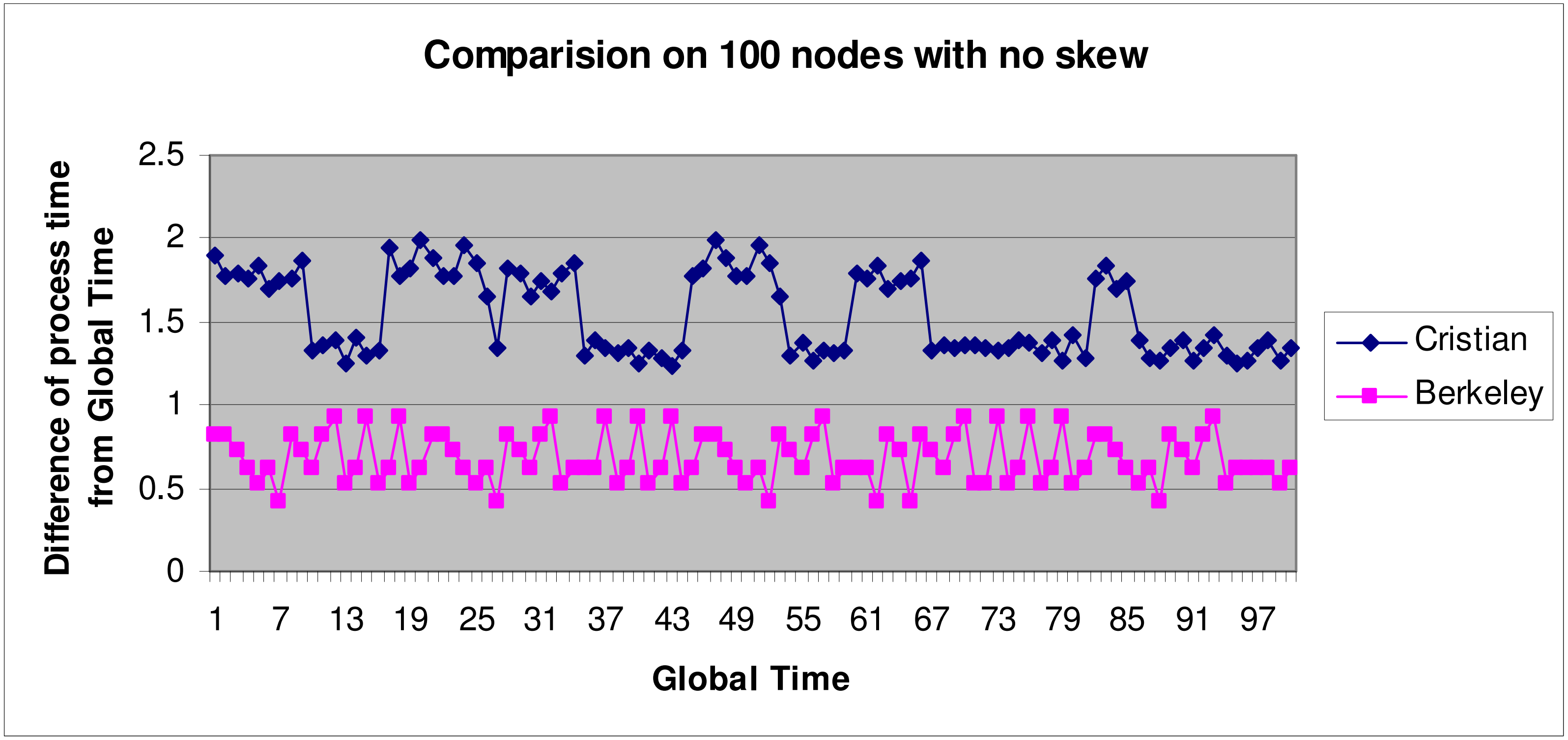


Comparison between Cristian and Berkeley Algorithms

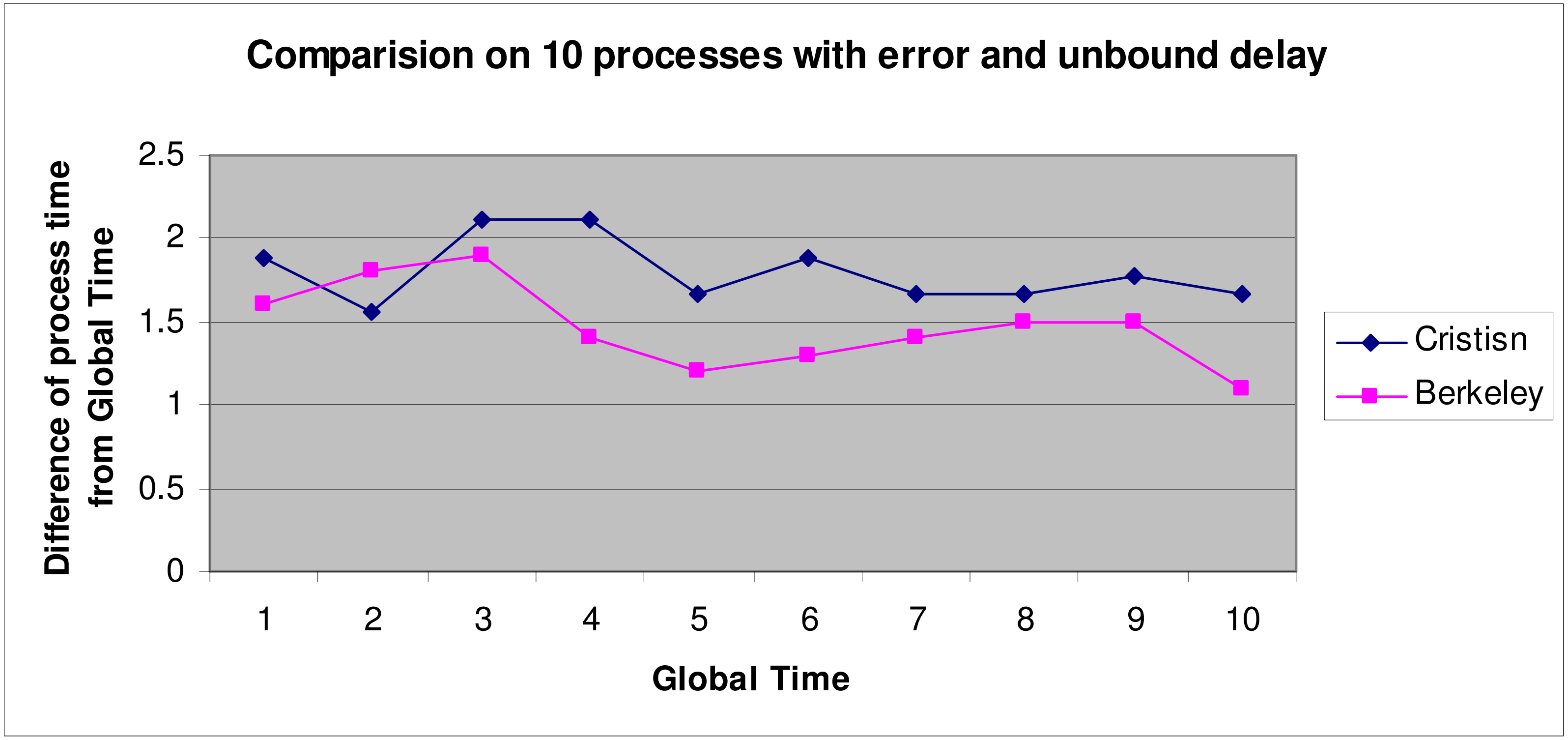


12/6/2007

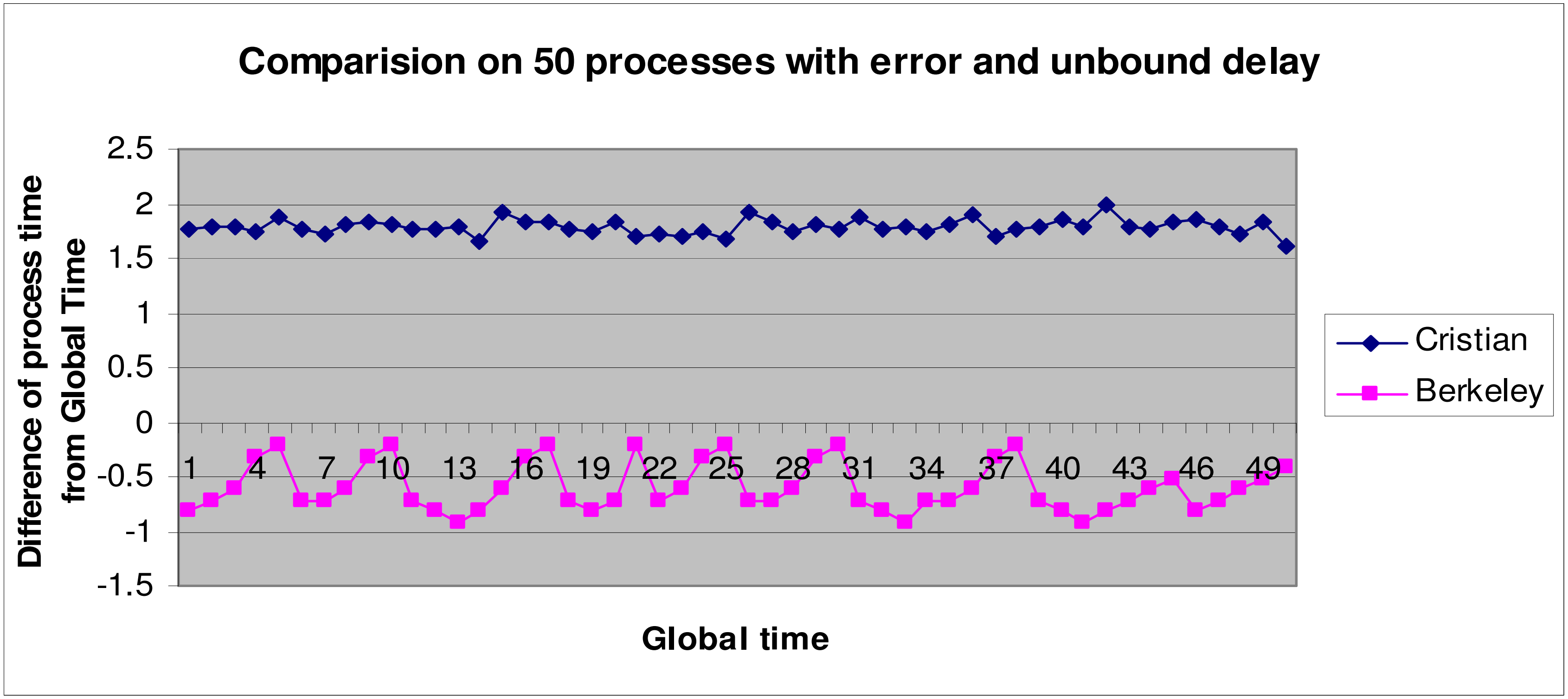
Comparison between Cristian and Berkeley Algorithms



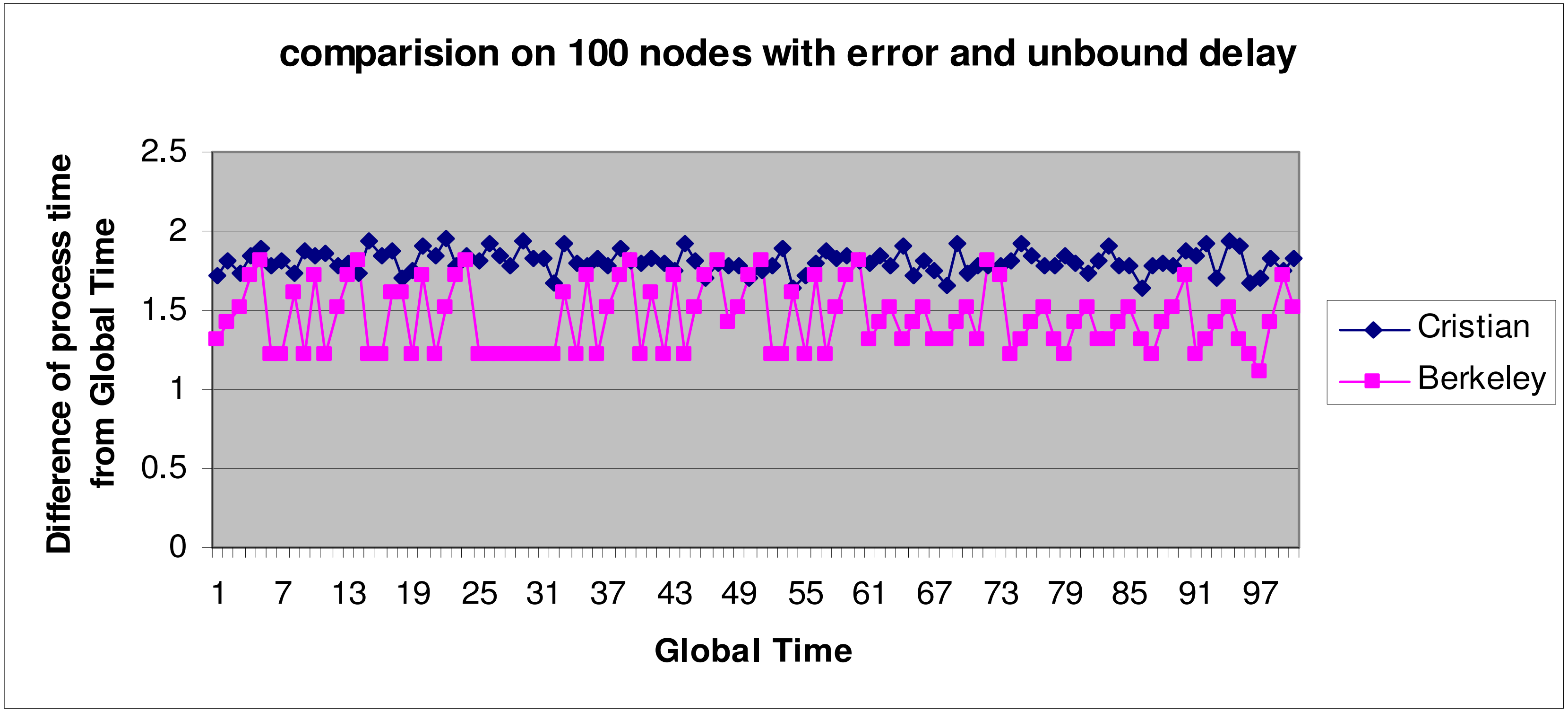
Comparison between Cristian and Berkeley Algorithms



Comparison between Cristian and Berkeley Algorithms



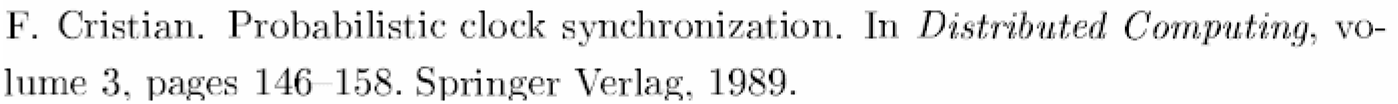
Comparison between Cristian and Berkeley Algorithms

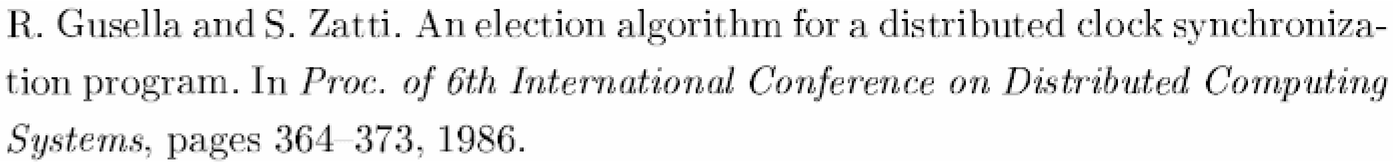


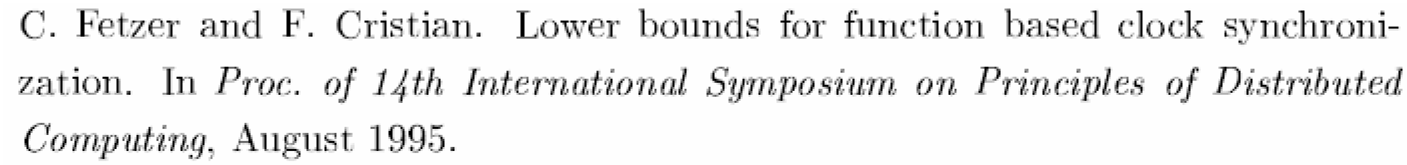
Future Work

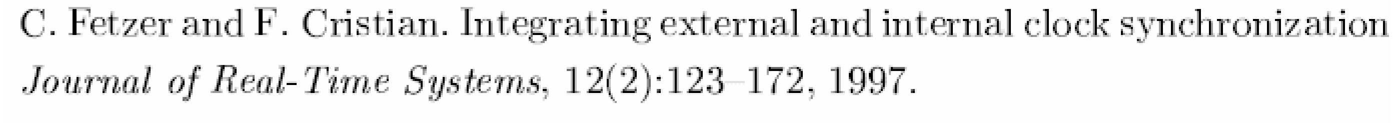
- Enhance design for fault tolerance in distributed system.
- Measure the performance of the system by combining Cristian’s and Berkley time synchronization algorithms.

References:


F. Cristian. Probabilistic clock synchronization. In *Distributed Computing*, volume 3, pages 146–158. Springer Verlag, 1989.


R. Gusella and S. Zatti. An election algorithm for a distributed clock synchronization program. In *Proc. of 6th International Conference on Distributed Computing Systems*, pages 364–373, 1986.


C. Fetzer and F. Cristian. Lower bounds for function based clock synchronization. In *Proc. of 14th International Symposium on Principles of Distributed Computing*, August 1995.


C. Fetzer and F. Cristian. Integrating external and internal clock synchronization *Journal of Real-Time Systems*, 12(2):123–172, 1997.

References:

- http://deneb.cs.kent.edu/~mikhail/classes/aos.f07/Materials/SinhaPhysClocks.pdf
- http://deneb.cs.kent.edu/~mikhail/Research/TR-KSU-CS-2007-04.pdf

Discussions

Any thoughts and questions…..