Computing Global Virtual Time

Issues and Some Solutions

Richard M. Fujimoto
Professor

Computational Science and Engineering Division
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0765, USA

http://www.cc.gatech.edu/~fujimoto/

Outline

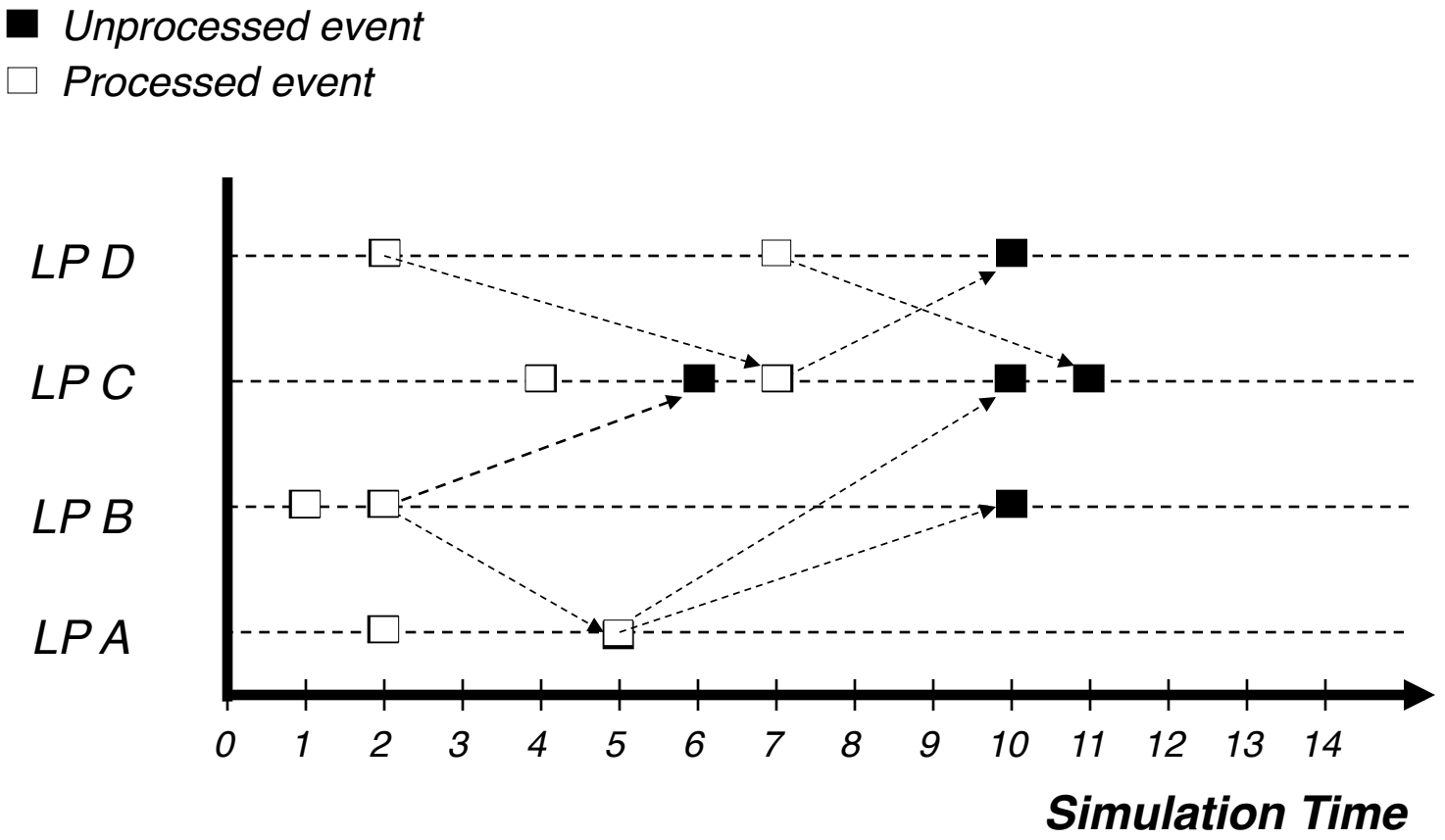
- GVT Computations: Introduction
  - Synchronous vs. Asynchronous
  - GVT vs. LBTS
- Computing Global Virtual Time
  - Transient Message Problem
  - Simultaneous Reporting Problem
- Samadi Algorithm
  - Message Acknowledgements
  - Marked Acknowledgment Messages

Global Virtual Time

GVT(t): lower bound on time stamp of any future rollback at wallclock time t

- Needed to commit I/O operations, reclaim memory
- Observation: All rollbacks are caused by receiving a message (or anti-message) in an LP's past
  - Transient messages or anti-messages that have not been delivered can cause rollback
  - LP may later generate a new positive event, but the time stamp of this event must be greater than or equal to an unprocessed (or partially processed) event in the LP.
  - LP may later send a new anti-message, but this only occurs after a rollback to time T where T is less than the time stamp of the antimessage.
  - Anti-messages in output queue not considered in GVT computation, only those that have been sent.
  - GVT can be computed as the minimum time stamp among *all* unprocessed or partially processed messages at wallclock time t.
- Computing GVT straightforward if an instantaneous snapshot of the computation could be obtained: compute minimum time stamp among
  - Unprocessed events & anti-messages within each LP
  - Transient messages (messages sent before time t that are received after time t)

What is the GVT in this Snapshot?



GVT vs. LBTS

Computing GVT is similar to computing the lower bound on time stamp (LBTS) of future events in conservative algorithms

- GVT algorithms can be used to compute LBTS and vice versa
- Both determine the minimum time stamp of messages (or anti-message) that may later arrive
  - Historically, developed separately
  - Often developed using different assumptions (lookahead, topology, etc.)
- Time Warp
  - Latency to compute GVT typically less critical than the latency to compute LBTS
  - Asynchronous execution of GVT computation preferred to allow optimistic event processing to continue

Synchronous vs. Asynchronous

- Synchronous GVT algorithms
  - LPs stop processing events once a GVT computation has been detected
  - LPs resume once GVT computation complete (e.g., new GVT value is distributed)
- Asynchronous GVT algorithms
  - LPs can continue processing events and schedule new events while the GVT computation proceeds “in background”
  - Asynchronous is preferable from performance standpoint

Asynchronous GVT

An incorrect GVT algorithm:

- Controller process: broadcast “compute GVT request”
- upon receiving the GVT request, each process computes its local minimum and reports it back to the controller
- Controller computes global minimum, broadcast to others

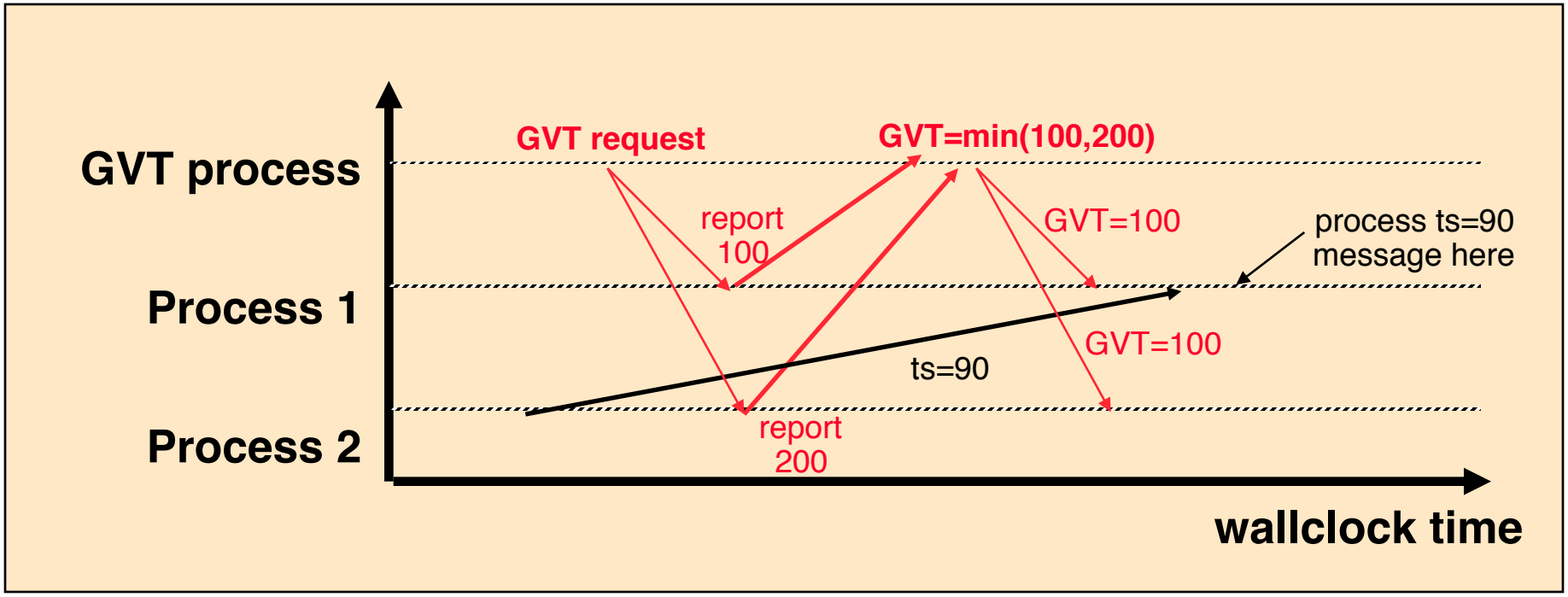
Difficulties:

- *transient message problem*: messages sent, but not yet received must be considered in computing GVT
- *simultaneous reporting problem*: different processors report their local minima at different points in wallclock times, leading to an incorrect GVT value

The Transient Message Problem

Transient message: A message that has been sent, but has not yet been received at its destination

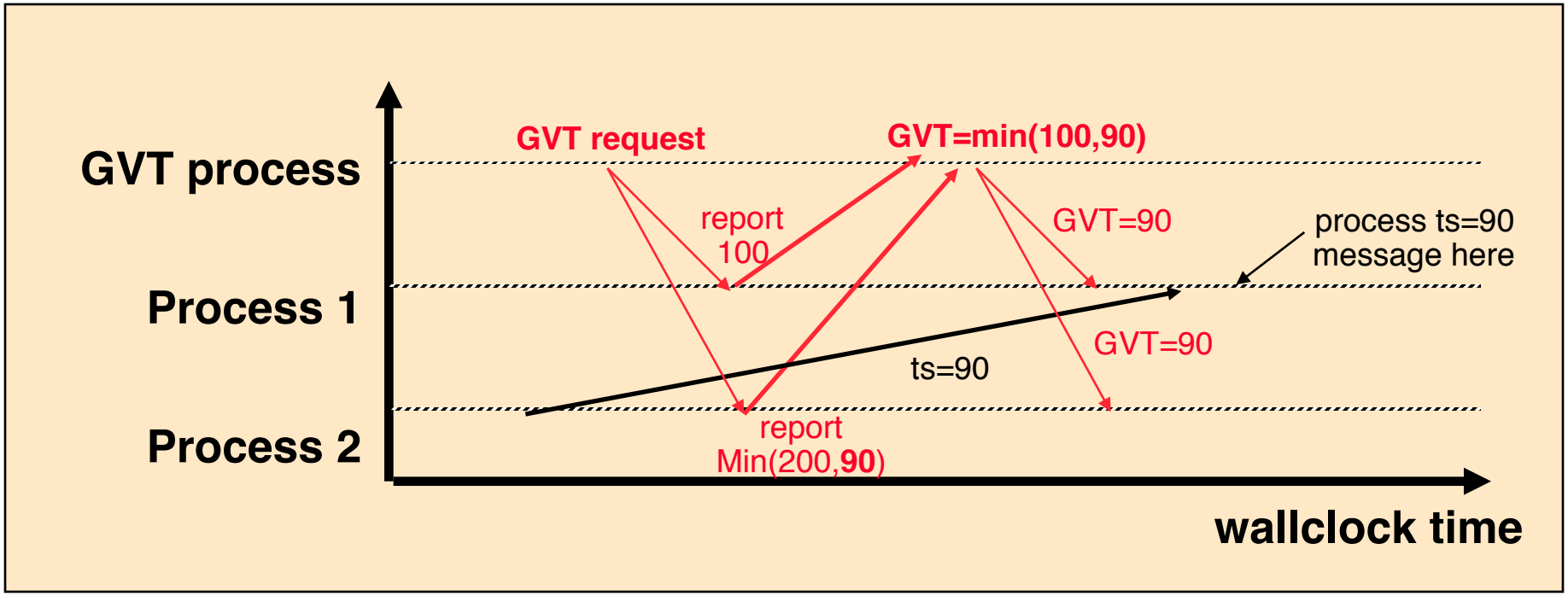
Erroneous values of GVT may be computed if the algorithm does not take into account transient messages



Transient Messages: A Solution

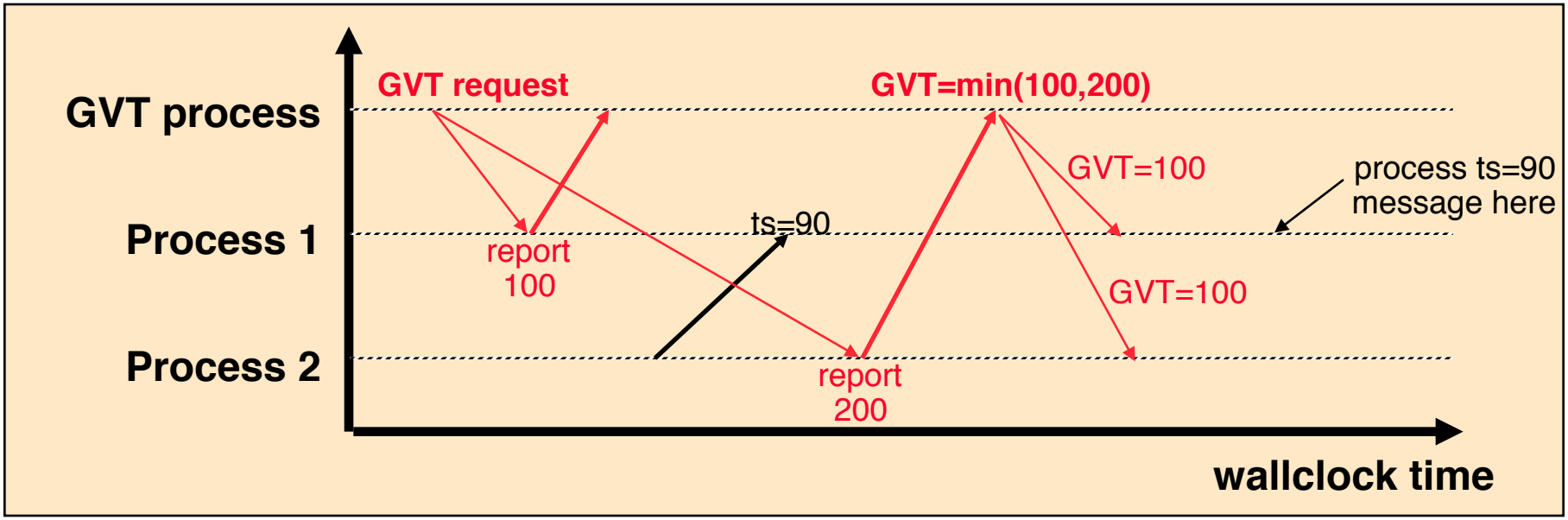
Send an acknowledgement message for each message

Report minimum of (local messages/anti-messages, time stamp of any unacknowledged messages)



The Simultaneous Reporting Problem

Erroneous values of GVT may be computed when processes report local minima at different points in (wallclock) time.



Process 1 can't account for time stamp 90 message

Process 2 assumes process 1 will account for the message

Do message acknowledgements solve this problem?

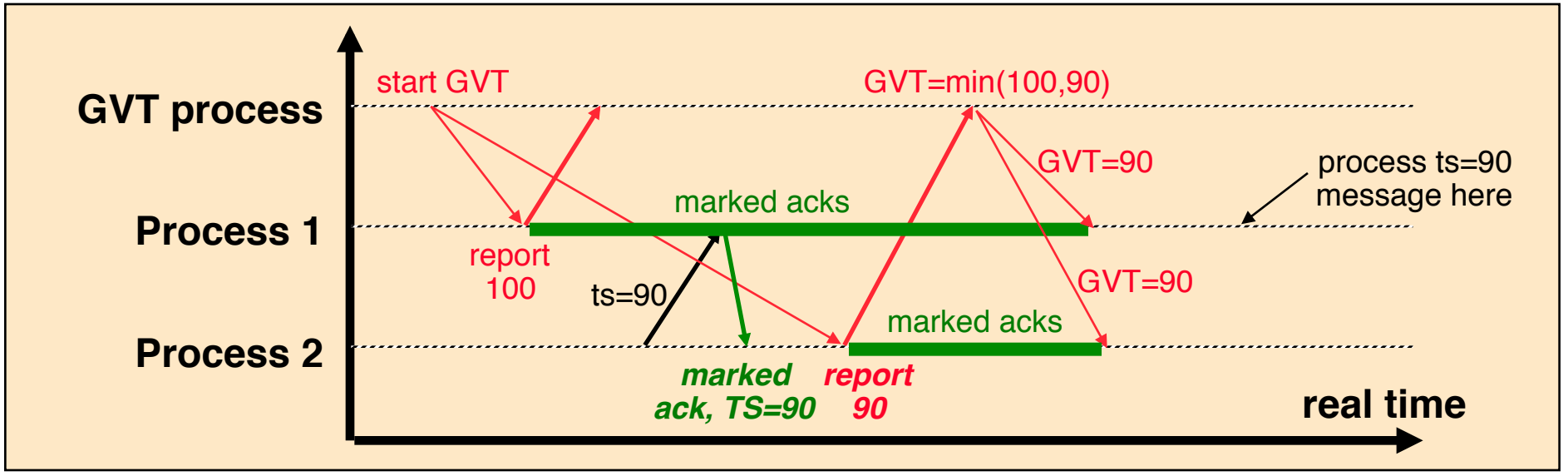
- No, at least not by themselves
- Solution: Mark acks that are sent after local min has been reported

Samadi Algorithm

- send an ack for each event messages & anti-messages received
- “mark” acks sent after the processor has reported its local minimum

Algorithm:

- controller broadcasts “start GVT” message
- each processor reports minimum time stamp among (1) local messages, (2) unacknowledged sent messages, (3) marked acks that were received
- subsequent acks sent by process are marked until new GVT is received
- controller computes global minimum as GVT value, broadcasts new GVT



Summary

- Global Virtual Time
  - Compute lower bound on time stamp of future rollbacks
    - Asynchronous GVT computation highly desirable to avoid unnecessary blocking
- Samadi Algorithm
  - Transient message problem: Message acknowledgements
  - Simultaneous reporting problem: Mark acknowledgements sent after reporting local minimum
  - Requires acknowledgements on event messages