

A Parallel, Optimistically Synchronized VHDL Simulator Executing on a Network of Workstations*

Tim McBrayer, David Charley, Philip A. Wilsey, and Debra A. Hensgen
Dept. of Electrical and Computer Engineering
University of Cincinnati, Cincinnati, OH 45221-0030
{tmcbraye,dcharley,paw,dhensgen}@thor.ece.uc.edu

As VHDL grows in popularity, larger and larger models are designed in VHDL, requiring considerable system resources for simulation. A simulation's demand for resources can easily overload a single system in terms of processor power, physical memory, and disk space. A method of dealing with this problem is to design a simulator which uses the capabilities of a multiple-instruction multiple-data (MIMD) machine. This has been accomplished at the University of Cincinnati through the QUEST project on the ES-Kit, a MIMD machine with 16 Motorola 88000-based nodes [1]. However, the ES-Kit is not generally available and we are therefore modifying the QUEST simulator to run on a network of UNIX workstations.

The current design runs on a network of Silicon Graphics or SUN workstations. VHDL code is first processed by the Vantage compiler to produce a parsed tree in Vantage Intermediate Format (VIF). The VIF is then analyzed for (i) optimization information, (ii) processor assignment, and (iii) code generation (producing C++). The generated code uses an optimized version of the apE flux communication layer to manage communication across one of three communication layers (shared memory, a hardware supported high speed inter-workstation shared memory called SCRAMNET, and ethernet).

At present, many of the known optimizations to Time Warp have been incorporated as compile time switches into the simulator and new optimizations are also being implemented. For example, we have developed and are implementing rollback relaxation [3] and an efficient implementation of lazy reevaluation [2]. The communication support system is also being modified to sort the message queues in timestamp order and also to accelerate the delivery of antimessages. As part of this implementation we are also working to develop an efficient marking strategy for preempted transactions (a problem aggravated by the out of order message delivery possible in the current communication subsystem).

References

- [1] H. Carter, R. Vemuri, P. A. Wilsey, J. Aylor, R. Waxman, and T. Hartrum. High speed acceleration of VHDL simulation, synthesis, and ATPG: Overview of the QUEST project. In *VHDL Users' Group Spring 1991 Conference*, pages 85-90, Cincinnati, OH, April 1991.
- [2] A. Palaniswamy, S. Aji, and P. A. Wilsey. An efficient implementation of lazy reevaluation. In *Proc. of the 25th Annual Simulation Symposium*, pages 140-146, April 1992.
- [3] P. A. Wilsey and A. Palaniswamy. Rollback relaxation. Technical Report TR 135-2-92-ECE, Department of Electrical and Computer Engineering, University of Cincinnati, February 1992.

*This work is partially supported by the Air Force under order number 054458 and also by the Defense Advanced Research Projects Agency under order number 7056, monitored by the Federal Bureau of Investigation under contract number J-FBI-89-094.

A Parallel, Optimistically Synchronized VHDL Simulator Executing on a Network of Workstations

October 21, 1992

Timothy J. McBrayer
University of Cincinnati
E.C.E. Department, ML 30
Cincinnati, OH 45221-0030
(513) 556-0904
tmcbraye@thor.ece.uc.edu

P. A. Wilsey, University of Cincinnati
D. Charley, University of Cincinnati
D. A. Hensgen, University of Cincinnati

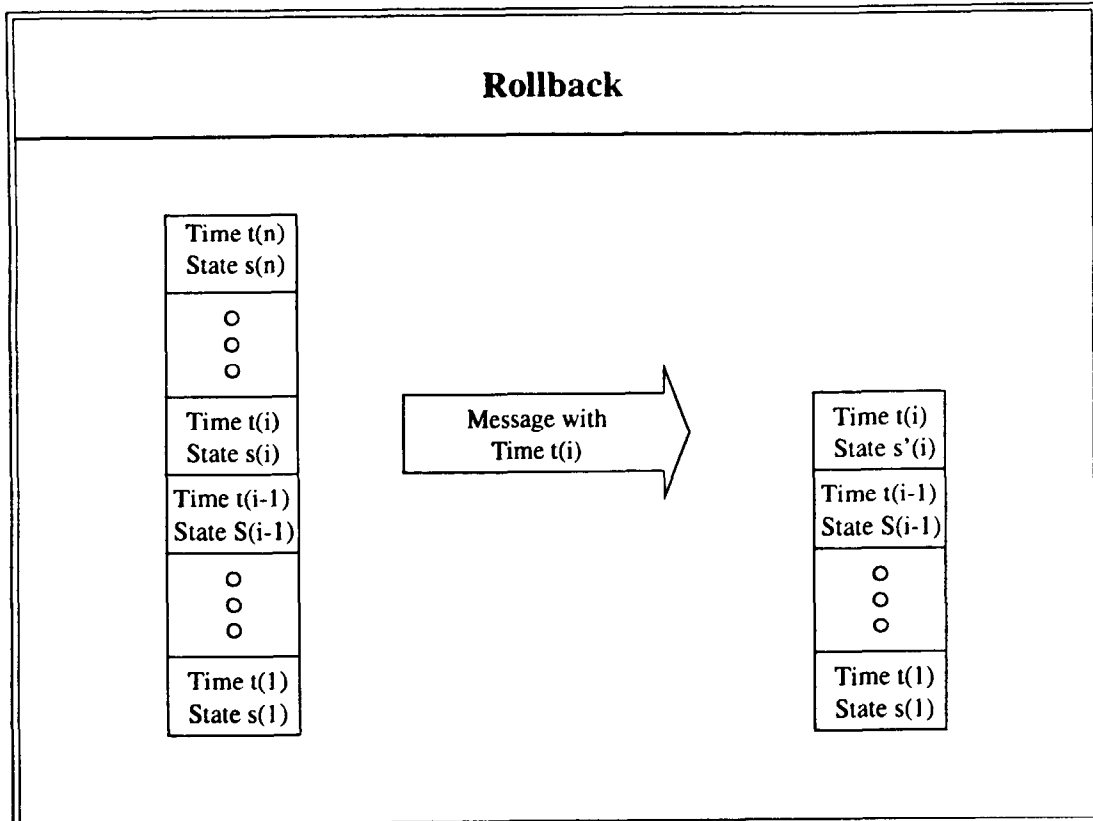
T. J. McBrayer

Parallel Simulation Synchronization

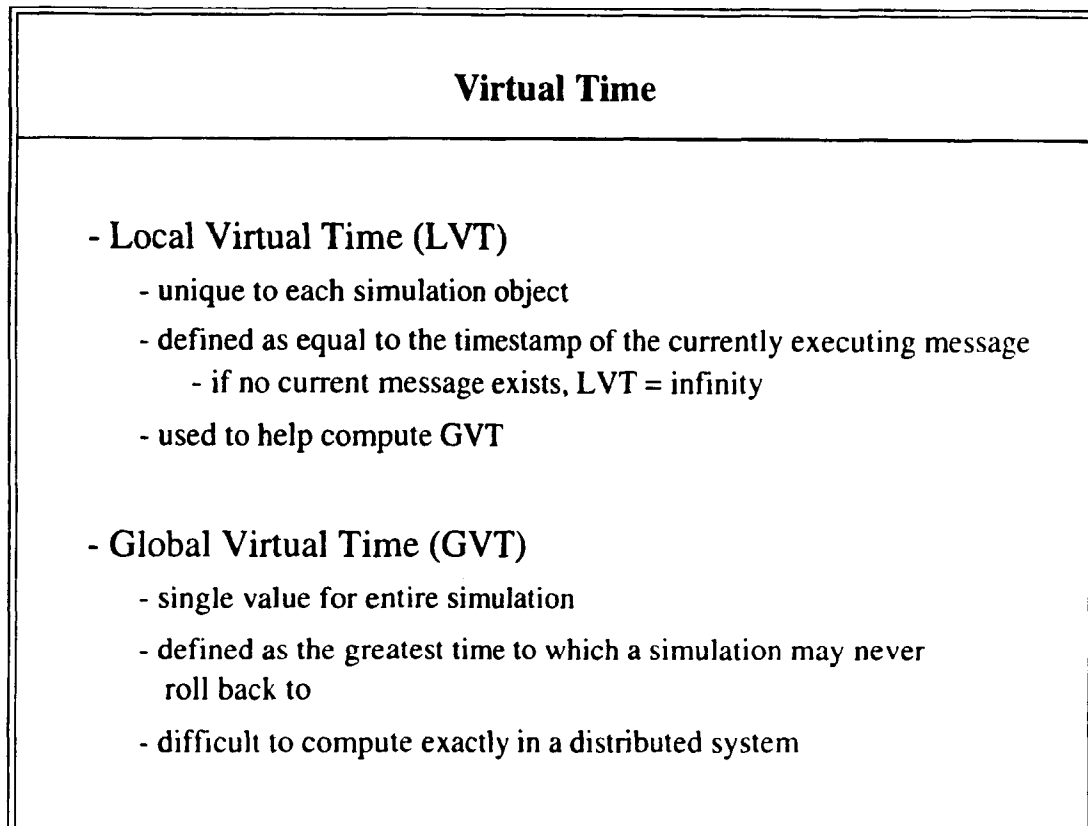
- Conservative
 - only objects with lowest simulation time execute at given real time
 - objects must wait for all input vectors before proceeding
 - processors may sit idle, wasting computational power
 - simulation can deadlock

- Optimistic
 - all objects can always execute
 - objects assume a default value for unreceived input vectors
 - processors/workstations never sit idle
 - simulation can never deadlock
 - incorrect assumption of input vectors cause **rollbacks**

T. J. McBrayer



T. J. McBraver



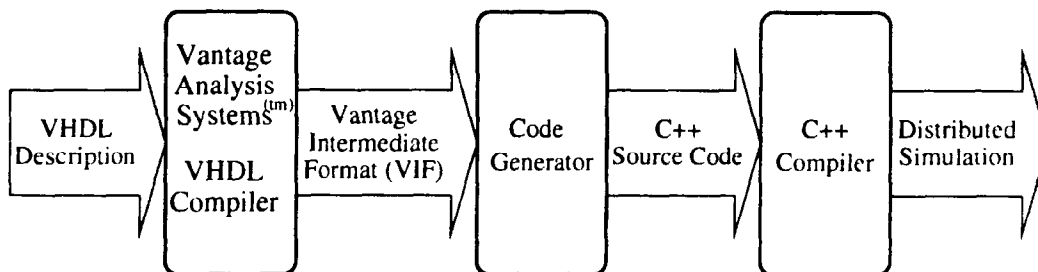
T. J. McBraver

Optimizations to Optimistic Synchronization

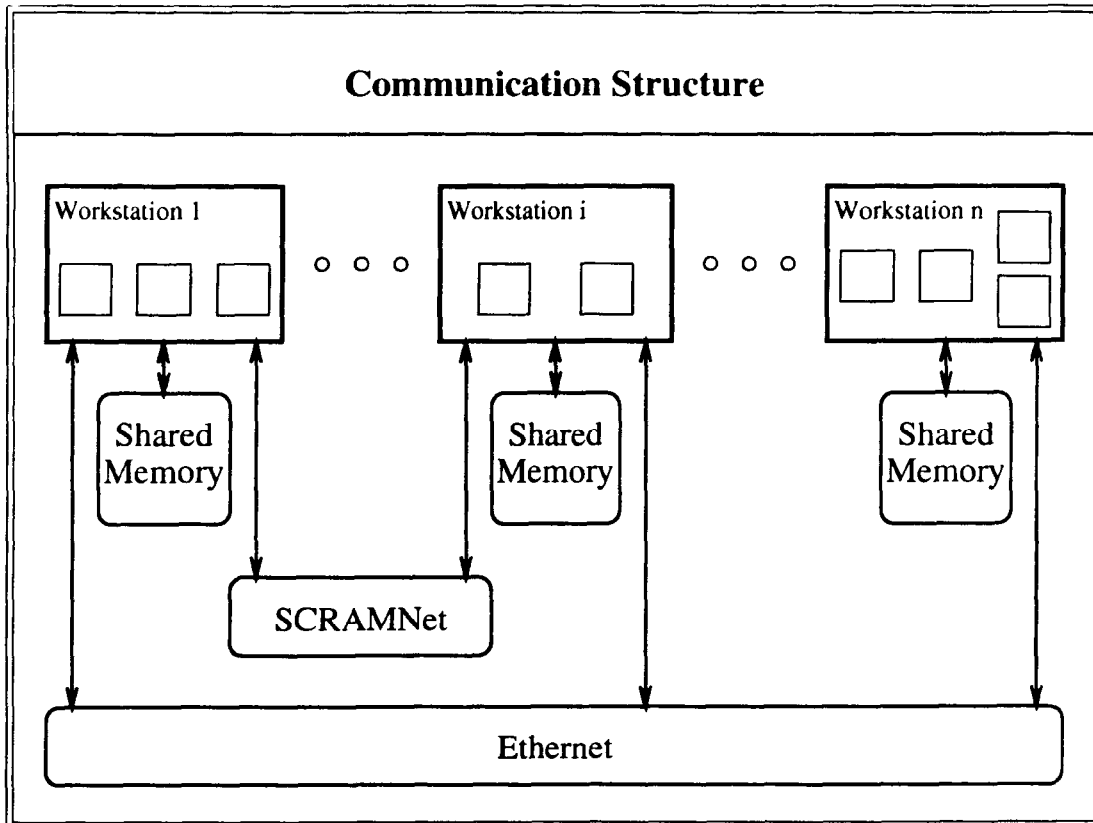
- Periodic State Savings
- Rollback Relaxation
- Lazy Cancellation
- Lazy Reevaluation

T. J. McBrayer

VHDL Simulation Build Sequence

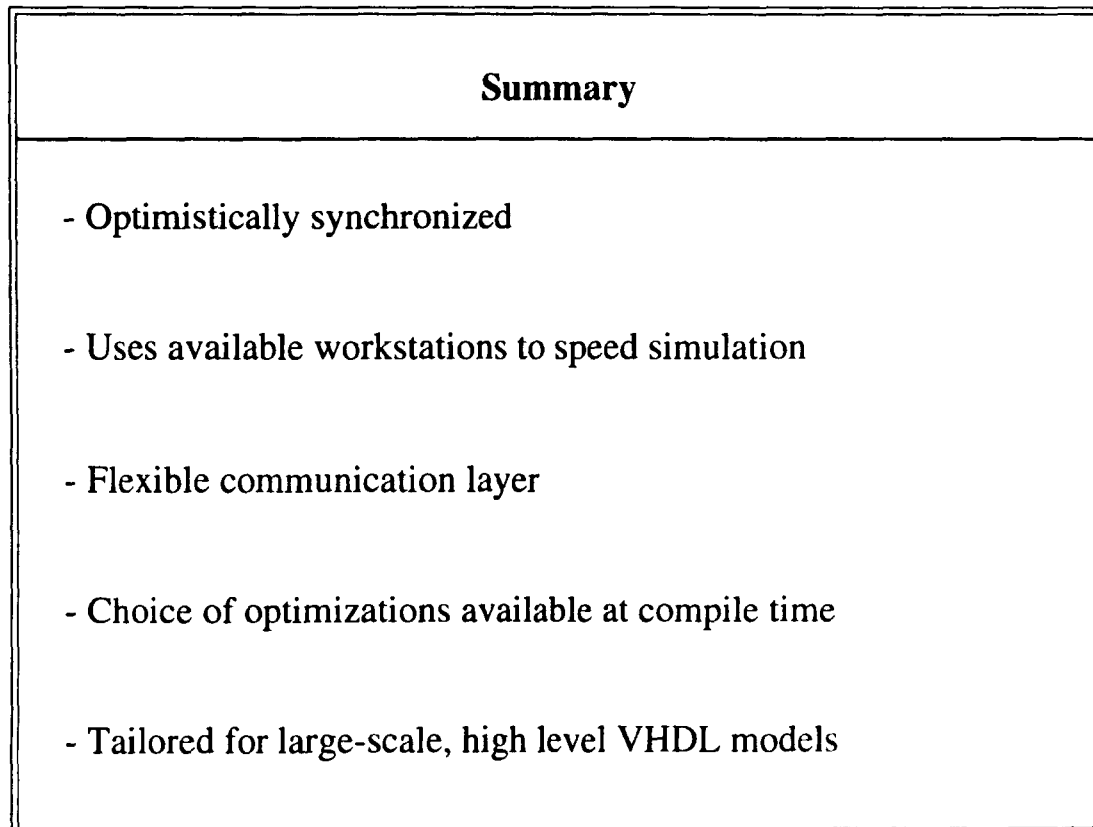


T. J. McBrayer



T. J. McBrayer

7



T. J. McBrayer

8