Scalable and Consistent Virtual Worlds

The Proposed Time Management based on Constrained Communication Model

Umar Farooq and John Glauert

School of Computing Sciences,

University of East Anglia,

Norwich, UK.

University of East Anglia

Outline

- Introduction
- Previous Work
- The proposed Time Management approach
- Illustrations, Simulations, and Comparison
- Conclusions
- Future Work
- Questions

Introduction

- ***"A VE/Virtual World is an integrated/unified persistent 3D graphical simulation of real and imaginary contents, where avatars feel immersed through tele-presence in shared workspace, though geographically distributed both at infrastructure as well as application levels. Users collaboratively create and manipulate contents of the world they inhabit in."*** (based on definition of Rosedale et al. [1])

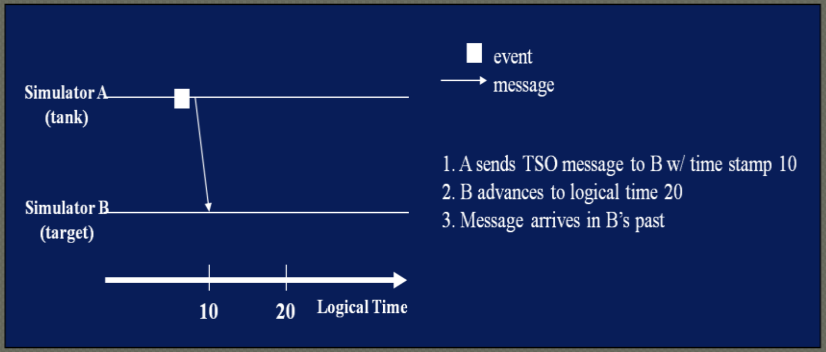
- A VE is an application of simulations (Fujimoto[2])

Introduction

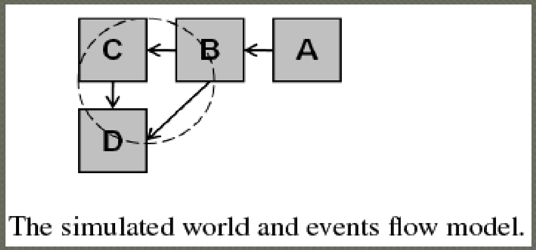
- Key Issues
  - Scalability : Splitting and distributing the world among a set of servers (has issues with conservative approaches)
  - Load Distribution
  - Consistency (Synchronisation/Time Management)

Synchronisation Problem

- Synchronisation maintains the temporal order of events in a system (called local causality constraint).
  - Examples: Flag Set Application, Fire/Destruction scenario
- Lookahead: determines a safe range of events to process
- LBTS (Lower Bound on Time Stamps) : to get a consistent environment time advance must be <= LBTS



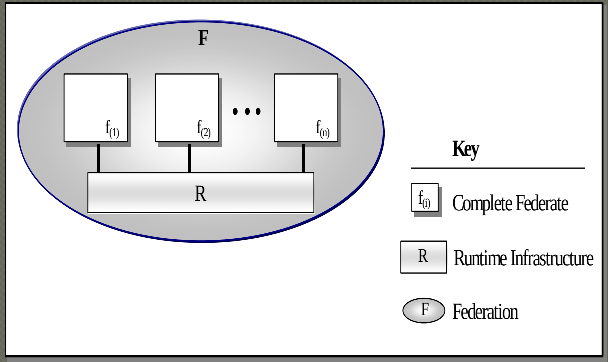
Violation of local causality constraint [3][4]



Flag Set Application

Synchronisation Problem

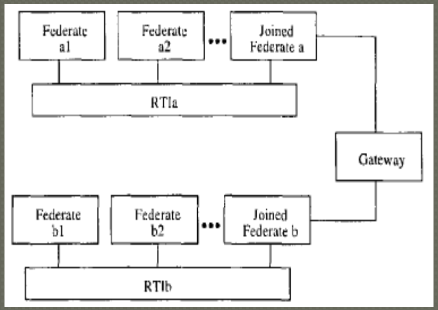
- The aim is to avoid receiving events in a federate past.
- The High Level Architecture (HLA) combines federates (individual simulator) into a federation (a collection of simulators).
- In HLA, the federates and RTI realise this together.



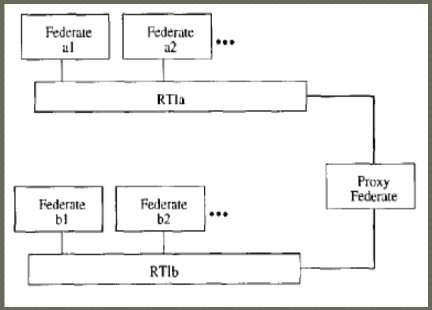
An HLA Federation [3][4]

Previous Work

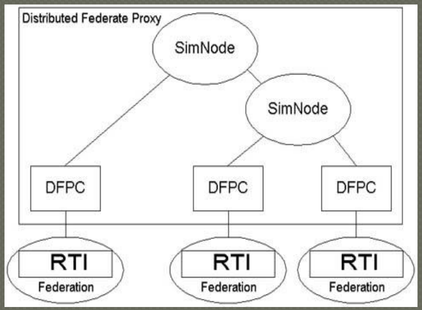
- Centralised Approaches
  - The HLA, for interoperability of simulators [2]
  - Interoperability among federations (a federation community)
    - Federation Gateway, Federate Proxy etc. [5]
- Distributed Approaches
  - Distributed Federate Proxy [6]



Federation Gateway



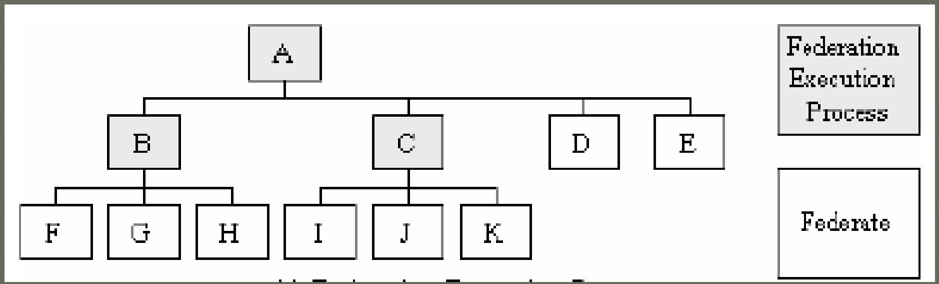
Federate Proxy



Distributed Federate Proxy
[Cramp et al. [6]]

Previous Work

- Hierarchical Approaches [7]
- Decentralised (Peer-to-Peer) Approaches [7]
- Not suitable for large scale virtual worlds with conservative behaviour.



Hierarchical HLA extension [Kim et al. [7]]

The Proposed Time Management Approach

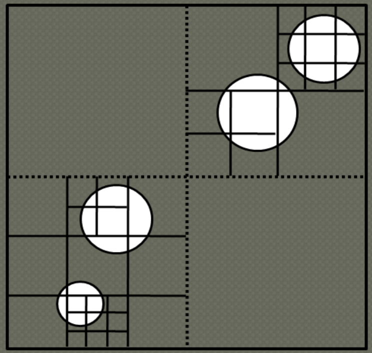
- Decentralised control, flat structure, and uses direct consultation with adjacent federates (sharing physical boundaries).

- It uses a restricted communication model based on inherent properties of virtual worlds (A region synchronises itself with regions that share boundaries with it).

- Objective:
  - To maintain local causality constraint.
  - To minimise number of intermediate hops and thus delay, complexity, number of messages communicated, and level of blockage.
  - To maximise scalability and interactive user experience.

The Proposed Time Management Approach

- A federate is a server executing a region.
- A federation (defined with respect to a federate) is a collection of federates that share boundaries with it.
- It uses a push strategy to reduce potential communication overhead and temporary blockage.
- A federate ensures that any destined messages are delivered (in sequence) before sending its LBTS value.



A hierarchical model based on JoHNUM strategies [8]

The Proposed Algorithm

```
Data: LocalQueue, LBTS, Lookahead, AdjacentLBTSValues
// Initialisations
// In general, the set of adjacent federates might change dynamically based on split and merge operations
int n = Number of adjacent federates;
int AdjacentLBTSValue[n];
for (i = 0; i < n; i++) do
|   AdjacentLBTSValue[i] = 0; // changes dynamically with the LBTS value sent by adjacent federate i
end
int LBTS = -1;// In order to force distribution of an initial LBTS value
int NewLBTS = 0;
Insert initial event(s) to LocalQueue; // used for synchronisation with other federates
// Main loop of program for safe processing
while (System is running) do
|   // Update LBTS value if necessary
|   NewLBTS = Min_{i=0}^{n-1}(AdjacentLBTSValue[i]);// determines minimum of LBTS values of adjacent federates
|   if (LocalQueue has Events) then
|   |   NewLBTS= Min(NewLBTS, Timestamp of earliest LocalQueue event);
|   end
|   if (NewLBTS > LBTS) then
|   |   LBTS = NewLBTS;
|   |   Send (LBTS + Lookahead) value to the adjacent federates;
|   end
|   // Check for an event that is safe to process
|   if (LocalQueue has Events and Timestamp of earliest LocalQueue event ≤ LBTS) then
|   |   Process Event; // Remove the event and may generate new internal and external events
|   |   Schedule internal and external events if any; // External events are sent to adjacent federates via messages
|   else
|   |   Go to Sleep;
|   end
end
```

Algorithm 1. The Proposed Decentralised Time Management Approach

Illustrations

- Simple illustration of a time advance
- Time advance for two independent federations without a common federate.
- Two federations with a common federate
- Temporarily blocking states of federates

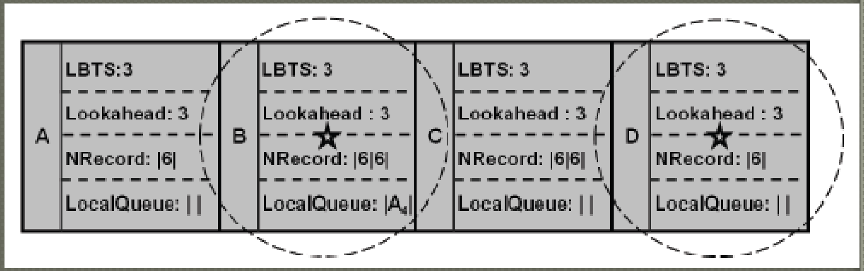
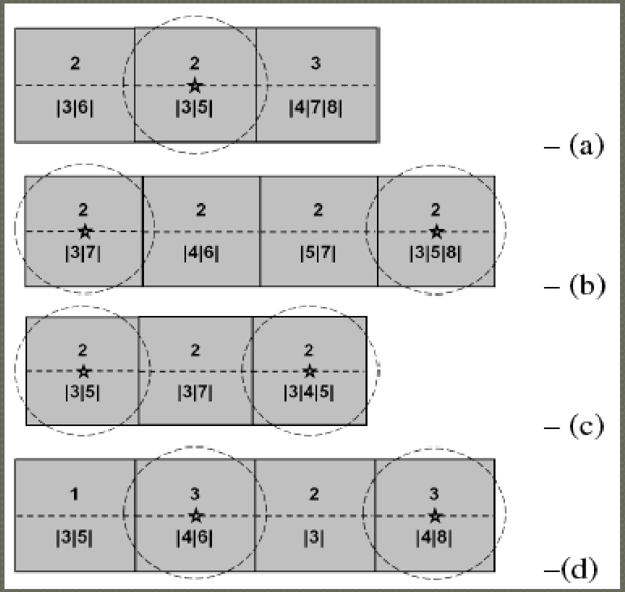


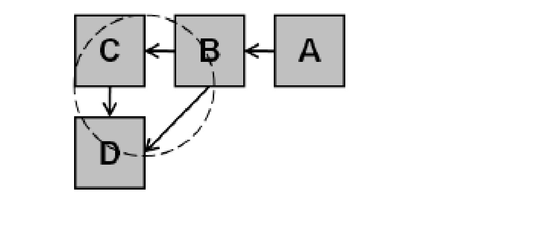
Illustration of a time advance



Illustrating the relevant concepts

Simulations

- Using a simple flag set application, randomly selected events, and delays in delivering messages.



The simulated world and events flow model.

Flag Set Application

Step	Region A					Region B					Region C					Region D (Observer)			
	Flag	LBTS	NRecord B	Local Queue	Event (s) Generated	Flag	LBTS	NRecord A,C,D	Local Queue	Event Generated	Flag	LBTS	NRecord B,D	Local Queue	Event (s) Generated	Flag	LBTS	NRecord B,C	Local Queue
-	0	-1	0	I_1	Empty	0	-1	0,0,0		Empty	0	-1	0,0		Empty	0	-1	0,0	
0	0	0	0	I_1	Empty	0	0	0,0,0	Empty	Empty	0	0	0,0	Empty	Empty	0	0	0,0	Empty
1	1	1	3	$[I_1]$	A_4	0	0	0,3,3	Empty	Empty	0	0	3,3	Empty	Empty	0	0	3,3	Empty
2	1	3	3	Empty	Empty	0	0	3,3,3	Empty	Empty	0	3	3,3	Empty	Empty	0	3	3,3	Empty
3	1	3	3	Empty	Empty	0	3	3,3,6	Empty	Empty	0	3	3,6	Empty	Empty	0	3	3,3	Empty
4	1	3	3	Empty	Empty	0	3	3,6,6	A_4	Empty	0	3	3,6	Empty	Empty	0	3	3,6	Empty
5	1	3	3	Empty	Empty	1	4	4,6,6	$[A_4]$	B_7	0	3	3,6	Empty	Empty	0	3	3,6	Empty
6-11	1	9	9	Empty	Empty	1	6	9,6,9	Empty	Empty	0	6	7,6	Empty	Empty	0	6	7,6	Empty
12	1	9	9	Empty	Empty	1	6	9,6,9	Empty	Empty	1	7	7,9	$[B_7]$	C_{10}	0	6	7,6	Empty
13	1	9	9	Empty	Empty	1	9	10,9,9	Empty	Empty	1	7	7,9	Empty	Empty	0	7	7,9	C_{10}
14	1	9	9	Empty	Empty	1	9	10,10,9	Empty	Empty	1	9	9,9	Empty	Empty	0	7	7,10	C_{10}
15-17	1	12	12	Empty	Empty	1	10	12,12,10	Empty	Empty	1	10	12,10	Empty	Empty	0	7	7,10	C_{10}
18	1	12	12	Empty	Empty	1	10	12,12,10	Empty	Empty	1	10	12,10	Empty	Empty	B	7	9,10	$[B_7]$, C_{10}
19	1	12	12	Empty	Empty	1	10	12,12,10	Empty	Empty	1	10	12,10	Empty	Empty	C	10	12,10	$[C_{10}]$

A simulation run of the proposed Time Management algorithm for flag set application

Simulations

- Non synchronised scenarios
  - D considering its own time information
  - D considering C (but not B) in addition to its own time information
  - Both violates the local causality, and allow C_{10} to process before B_7

Step	Region A					Region B					Region C					Region D (Observer)			
	Flag	LBTS	NRecord B	Local Queue	Event (s) Generated	Flag	LBTS	NRecord A,C,D	Local Queue	Event (s) Generated	Flag	LBTS	NRecord B,D	Local Queue	Event (s) Generated	Flag	LBTS	NRecord B,C	Local Queue
-	0	-1		I_1	Empty	0	-1		Empty	Empty	0	-1		Empty	Empty	0	-1		Empty
0	1	1		[I_1]	A_4	0	-1		Empty	Empty	0	-1		Empty	Empty	0	-1		Empty
1-3	1	1		Empty	Empty	0	-1		Empty	Empty	0	-1		Empty	Empty	0	-1		Empty
4	1	1		Empty	Empty	1	4		[A_4]	B_7	0	-1		Empty	Empty	0	-1		Empty
5-11	1	1		Empty	Empty	1	4		Empty	Empty	0	-1		Empty	Empty	0	-1		Empty
12	1	1		Empty	Empty	1	4		Empty	Empty	1	7		[B_7]	C_{10}	0	-1		Empty
13	1	1		Empty	Empty	1	4		Empty	Empty	1	7		Empty	Empty	C	10		[C_{10}]
14-17	1	1		Empty	Empty	1	4		Empty	Empty	1	7		Empty	Empty	C	10		Empty
18	1	1		Empty	Empty	1	4		Empty	Empty	1	7		Empty	Empty	B	7		[B_7]

A simulation run of non-synchronised scenario (Scenario 1) for the flag set application

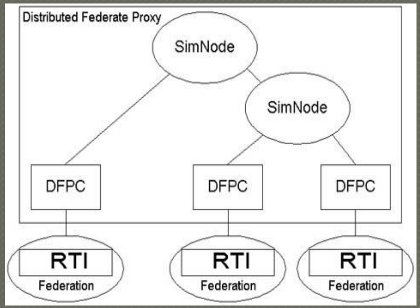
Simulations

- Decentralised/Peer-to-peer scenario
  - Achieves the same result and considers the entire set of LBTS values for an LBTS computation
  - Introduces longer delays and increases exchange of messages over network

An abstract Comparison

- Using Distributed Federate Proxy architecture as a reference, a number of potential parameters are highlighted for comparison in given table.

- The proposed approach is also well scalable and doesn't block the whole system.



Distributed Federate Proxy [Cramp et al. [6]]

Serial Number	Levels in Hierarchy	Algorithm	Number of hops	Complexity	Delay
1	2	Hierarchical/Decentralised	3	4*X	4*Y
		The Proposed Approach	0	X	Y
2	3	Hierarchical/Decentralised	5	6*X	6*Y
		The Proposed Approach	0	X	Y
3	4	Hierarchical/Decentralised	7	8*X	8*Y
		The Proposed Approach	0	X	Y

Comparison of existing hierarchical/distributed methods with the proposed Time Management Mechanism

Conclusions

- We proposed a Time Management approach and illustrated it with a number of examples.
- It is proved with a simple simulation model that it achieves the temporal order of events. An abstract model is used to compare it with the existing mechanisms.

Future Work

- Our future work includes the implementation of proposed Time Management approach together with JoHNUM infrastructure.
- It also include the detailed analysis and comparison with the help of further simulation and actual implementation.

References

[1] P. Rosedale and C. Ondrejka, Enabling Player Created Online Worlds with Grid Computing and Streaming, Gamasutra, 2003, 1-5.

[2] R.M. Fujimoto, Parallel and Distributed Simulation Systems, Proc. of the 2001 Winter Simulation Conference, 2001, 147-157.

[3] R.M. Fujimoto, Time Management in the High Level Architecture, Simulation, 71(6), 1998, 388-400.

[4] R.M. Fujimoto, Parallel and Distributed Simulation Systems (Wiley Interscience, 2000).

[5] W. Cai, S.J. Turner and B.P. Gan, Hierarchical Federations: An Architecture for Information Hiding, Proc. of the 15th Workshop on Parallel and Distributed Simulation, California, USA, 2001, 67-74.

[6] A. Cramp, J.P. Best and M.J. Oudshoorn, Time Management in Hierarchical Federation Communities, Proc. of the 2002 Fall Simulation Interoperability Workshop, Florida, USA, 2002.

[7] J.H. Kim and T.G. Kim, Hierarchical HLA: Mapping Hierarchical Model Structure into Hierarchical Federation, M&SMTSA, 2006, 75-80.

[8] Umar Farooq and John Glauert, "Joint Hierarchical Nodes based User Management (JoHNUM) Infrastructure for the Development of Scalable and Consistent Virtual Worlds", Proc. of the 13th IEEE/ACM Symposium on Distributed Systems and Real Time Applications (DSRT'09), Singapore, 2009.

Questions?

Thanks!