网络虚拟环境
Networked Virtual Environments

周　忠
虚拟现实技术与系统国家重点实验室
北京航空航天大学计算机学院
zz@vrlab.buaa.edu.cn

Start

第12讲　一致性与可靠性

一致性

分布式系统中的时间、时钟和事件序

分布式系统中的时间与事件一致性

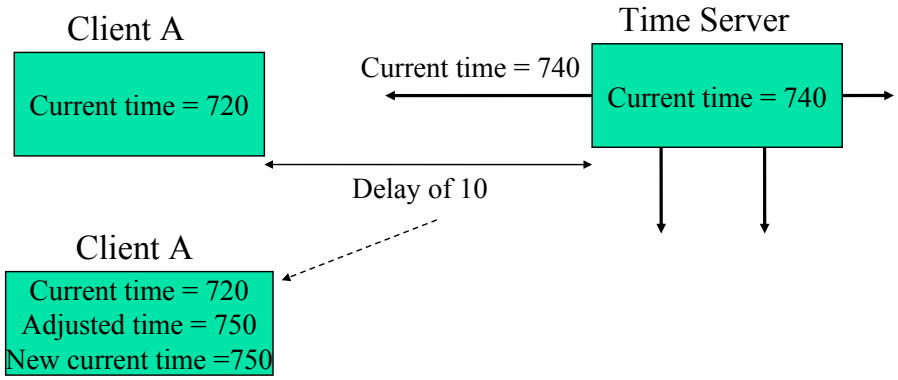
- 时间
- 时钟
- 事件序
- 时间感知

1. 统一时间的发布

- 时间的流逝不以人的意志为转移
- 时间服务器来发布全局的统一时钟
- 各节点根据全局时钟来调整自己的本地时钟
- 事件按照调整后的本地时钟排序

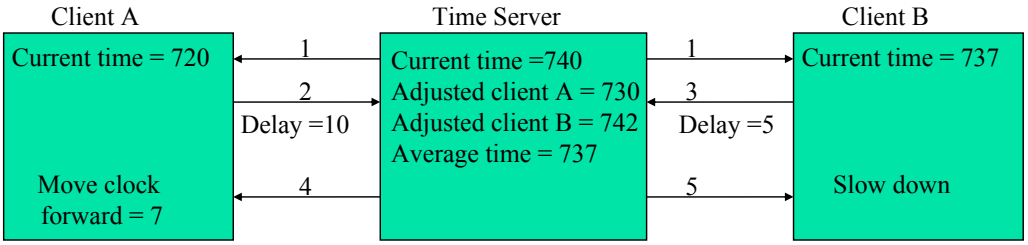
- NTP、Internet校准本地时钟等

时间服务器的后向调整



Cristian算法：时间服务器提供标准时钟
过快的时钟，减缓时钟推进速度
RTT修正；本地可通过插值进行时间推算

时间服务器的Berkeley算法



1. Current time = 740
2. My current time is 730
3. My current time is 742
4. Adjust forward
5. Adjust slowdown to accommodate 5

实现于BSD UNIX

Berkeley算法：时间服务器没有标准时钟，通过定期询问求出全局的平均时间

时间服务器方法

- 简单易行，通用性强
- 适用的条件
- 不适用的条件
- 网络抖动、发布频率…

2. 外部时间同步

- 第1种方法的改进，增加硬件同步源
- 同步信号触发
- GPS同步
- 光纤同步
- 外部网络同步

- 适用范围？

3. 事件一致性

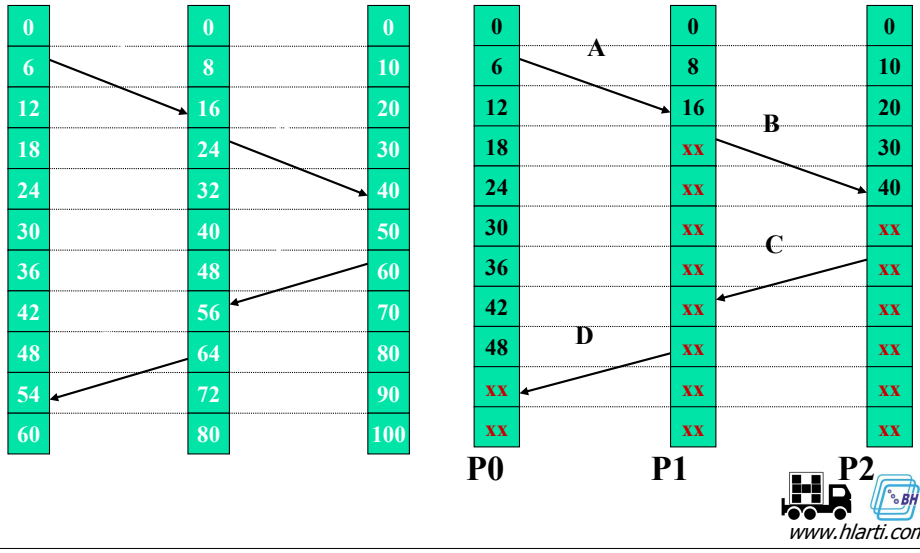
- Logical clock
- Logical time

- 事件一致性的核心在于实现各节点上事件的顺序一致性
- 事件戳 timestamp

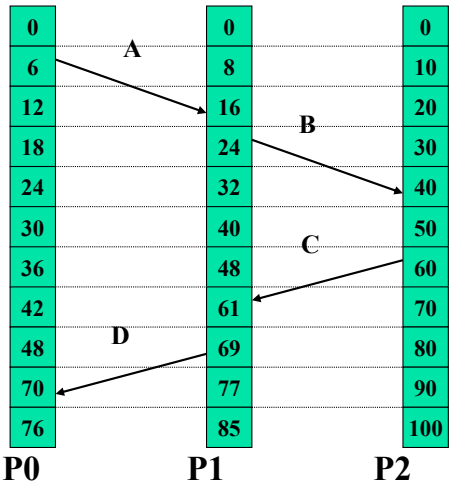
Lamport 1978

- 第1篇详细研究如何在分布式系统中进行事件排序的论文
  - "Time, clocks, and the ordering of events in a distributed system" [Lamport,1978]
- Lamport算法的核心是定义一个逻辑时钟
- Lamport逻辑时钟 [Lamport,1978], is "just a way of assigning a number to an event, where the number is thought of as the time at which the event occurred. … They may be implemented by counters with no actual timing mechanism".

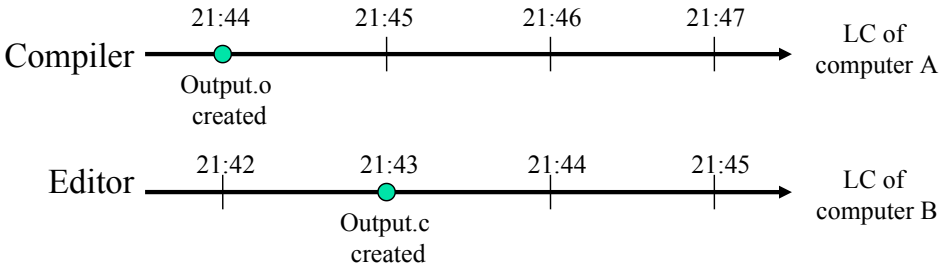
Lamport算法



Lamport算法



Example

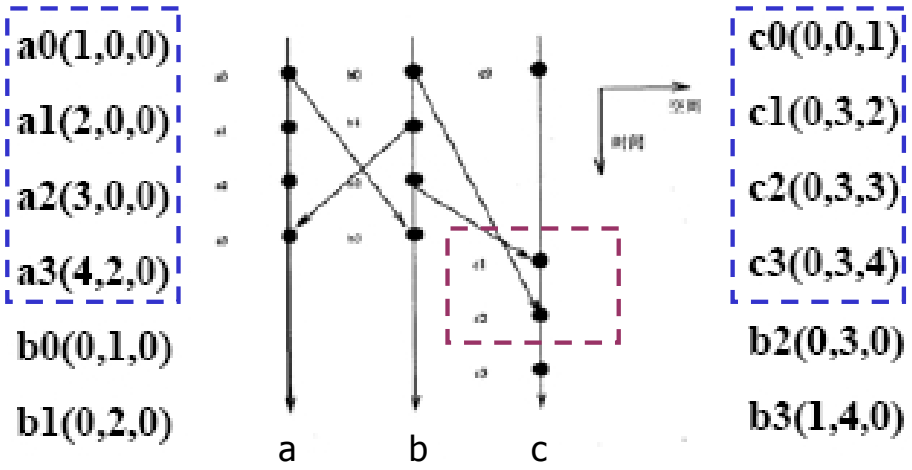


Lamport算法不能避免死锁

标量逻辑时钟 – 向量时钟

- 没有一个直接的全局逻辑时钟
- 在一个由n个并发进程构成的系统中，每个事件的逻辑时钟均由一个n维向量（n元组）构成，其中第i维（分量）对应于第i个进程的逻辑时钟Vi
- 维护方法：
  - 初值
  - 第i个进程在事件发生时，继承上一事件的逻辑时钟并将自身所对应的分量Vi增加一个步长
  - 任何进程Pi在发出任何信息m时都要将自己当前的逻辑时钟分量Vi一起发送出去
  - 如果是接收事件，且发送方为j，则比较自己继承的Vj和收到的逻辑时钟，并取其中较大者为自己的Vj

标量逻辑时钟 – 向量时钟



标量逻辑时钟 – 向量时钟

- 对于给定的时空视图其向量时钟的生成，算法可以发现系统是否存在因等待接收消息而引起的死锁，具体方法是：
  - i进程中接收a事件的向量时钟为$(...a_i.....)$，j为对应的发送事件b所属进程
  - j进程中b事件的向量时钟为$(...b_i.....)$，显然，按照定义$a_i >= b_i$
  - 若满足$a_i < b_i$，则存在一个因等待接收消息存在的死锁

标量逻辑时钟 – 向量时钟

- 在没有死锁的向量时钟系统中，设：
  - i进程中a事件的向量时钟为$(...a_i...a_j...)$
  - j进程中b事件的向量时钟为$(...b_i...b_j...)$
  - 若满足$a_i <= b_i$且$a_j < b_j$，则$a \rightarrow b$，否则$a \| b$
- 并发关系和发生在先关系仅反映2个事件之间的关系
  - 只有发生在先关系具有传递性，并发不具该特性
  - 如果有$a \rightarrow b$和$c \rightarrow b$，不能判断a和c之间的关系

分布式系统的事件处理

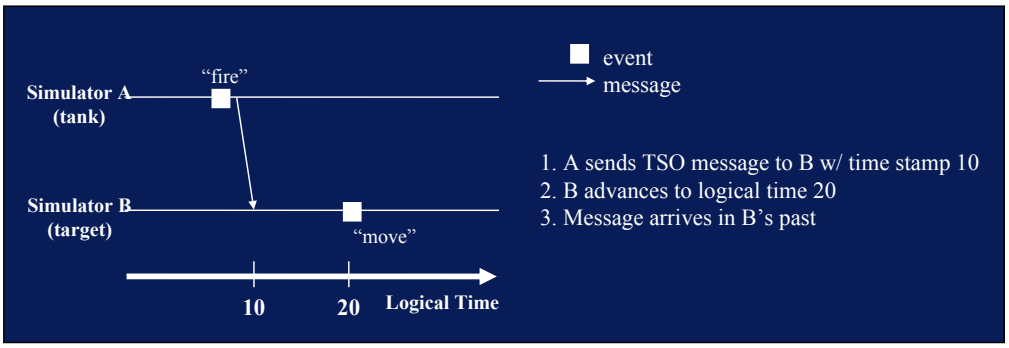
- 在1980's中期，大量的分布式处理系统在事件一致性方面开展研究
  - 一些可靠的事件处理排序算法已被申请专利
- 总结：这类事件一致性处理可以看作是因-果关系的一致性

4. 仿真时间

- 物理时间
  - 物理系统建模的物理时间
  - Example: 珍珠港事件仿真的物理时间为1941年12月7日00:00到18:00
- 逻辑时间（仿真时间）
  - 仿真节点的时间表达，一般为数值
  - Example:上述仿真中的逻辑时间取值区间为 double [0.0, 1080.0]，每个数字代表了物理时间某一分钟
- 墙上时间
  - 仿真中的真实世界的时间
  - Example: 上述仿真可能在2007年12月19日13:47到15:47的两个小时内进行

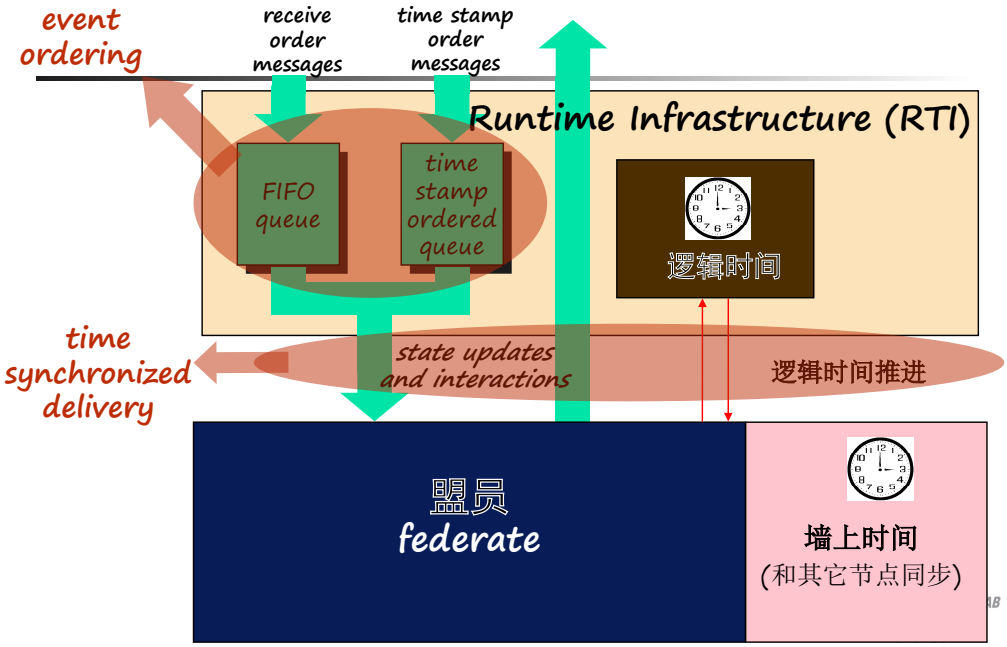
为什么是仿真时间？

- 两个离散事件仿真节点
- 光速的星云空间 – 网络速度的分布式系统



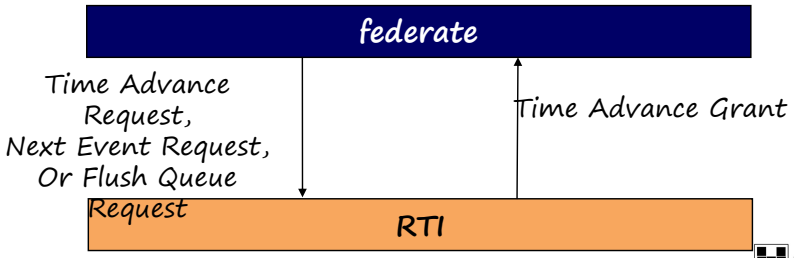
www.hlarti.com

HLA Time Management Services



HLA时间推进机制

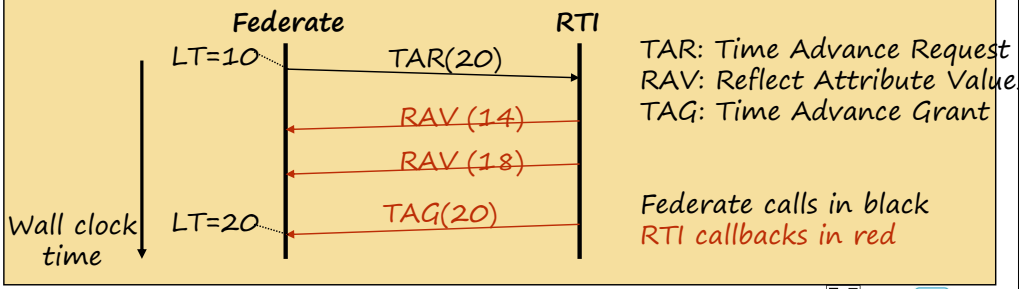
- 保守时间推进 TAR
  - 时间步进
- 乐观时间推进 FQR
  - 如果节点推进太快，通过时间弯曲机制 time warp 进行回滚
- 事件驱动的时间推进 NER



Time Advance Request(TAR)

- Federate调用Time Advance Request (T), 请求将逻辑时间LT推进到T
- RTI将所有time stamp ≤ T的 TSO消息发送给Federate
- RTI 确保所有time stamp ≤ T的TSO消息已经提交，调用Time Advance Grant (T)，将盟员推进到T

典型执行过程

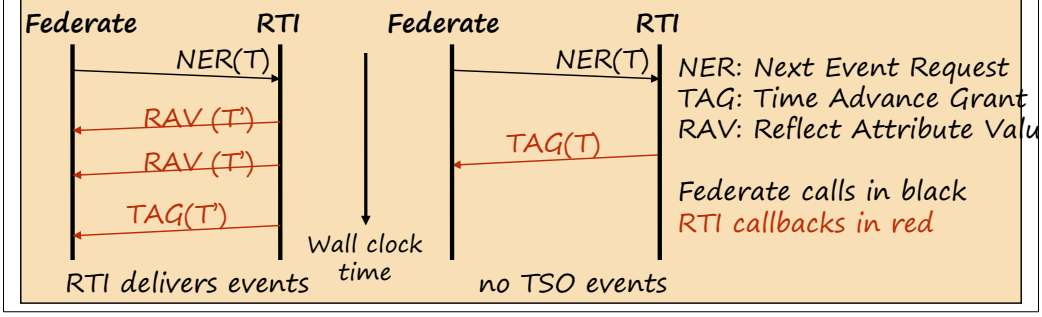


25

Next Event Request(NER)

- 目标： 所有的TSO消息都按TSO顺序处理
- Federate 调用 Next Event Request (T)
- If 下一条TSO消息的time stamp T' ≤ T
  - RTI 将所有时戳为 T'的TSO消息发送给Federate
  - RTI 调用 Time Advance Grant (T')
- Else RTI 调用 Time Advance Grant (T)

典型执行过程

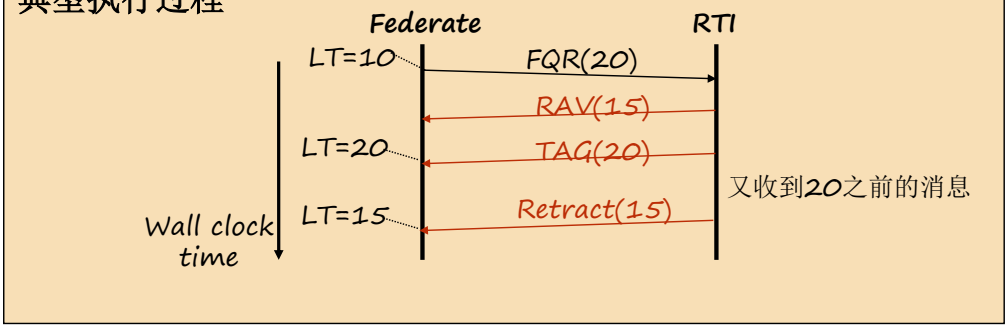


26

Flush Queue Request(FQR)

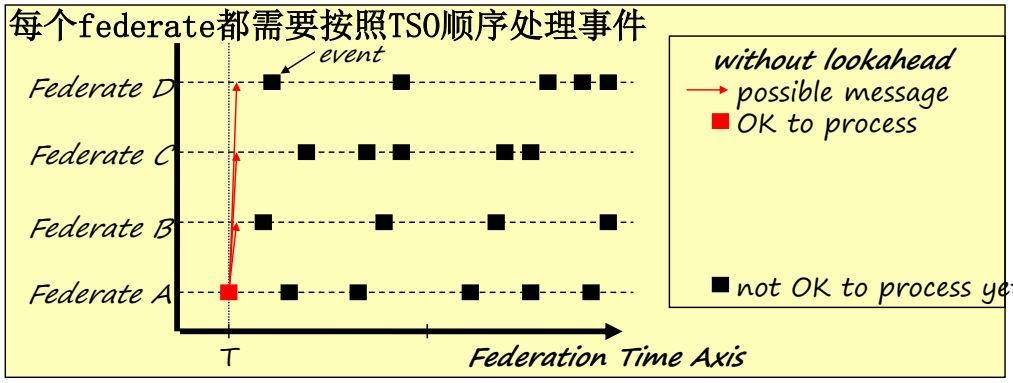
- 时间弯曲
- Federate需要完成回滚的任务

典型执行过程

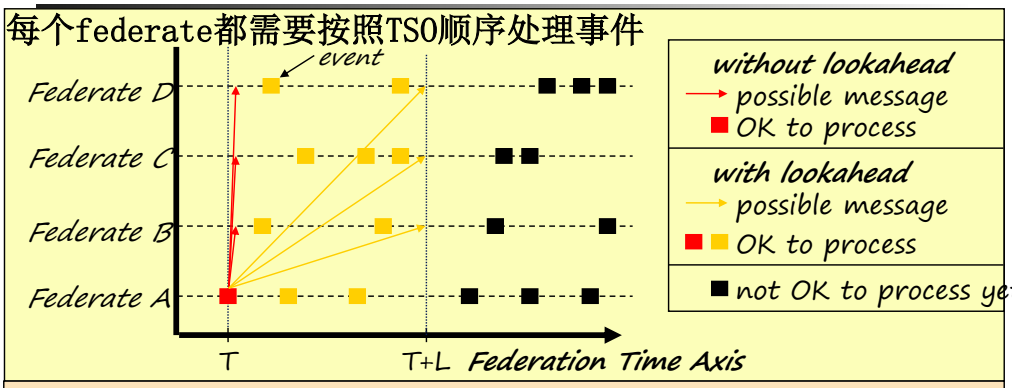


前瞻值Lookahead

NER只能并行处理相同时间戳的TSO消息



前瞻值Lookahead



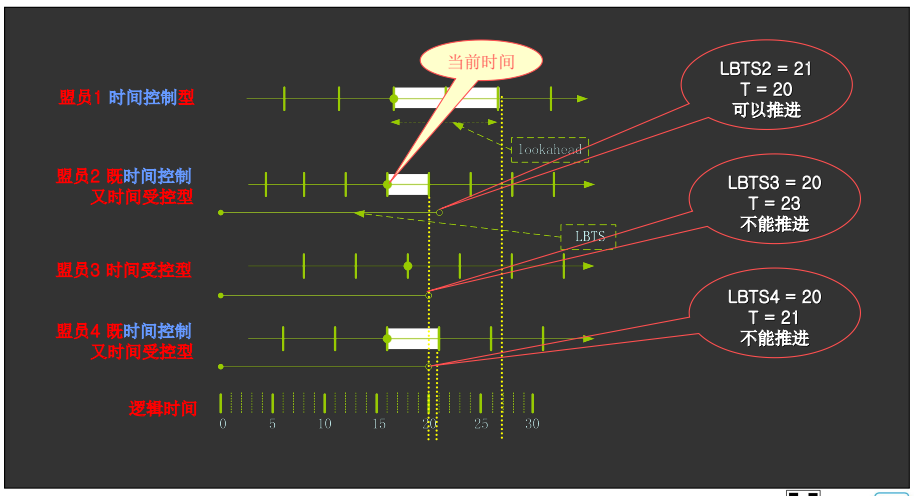
每个federate声明一个lookahead值 L; 该Federate发出的TSO消息必须符合 time stamp ≥ the federate's current time + L

Lookahead对TSO消息的并行处理是非常必要的 (乐观推进除外)

- 前瞻值 Lookahead
  - 盟员当前时间+前瞻值为盟员可以发送的时戳消息的时间戳下界
- LBTS (Lower Bound Time Stamp)
  - 盟员的LBTS：该盟员的最大时间推进值
  - 联盟的LBTS：所有盟员的LBTS的最小值

$$LBTSi = \begin{cases} \text{Min}(Tj + \text{lookaheadj}) & \text{如果盟员Fi是时间受控的成员，Fj是所有能给Fi发送TSO消息的盟员} \\ \infty & \text{如果盟员Fi不是时间受控的成员} \end{cases}$$

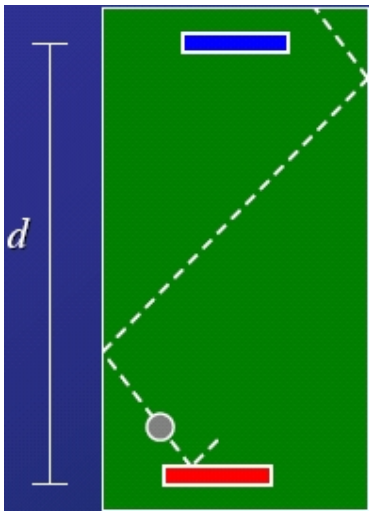
联盟的LBTS为：
LBTS = Min(LBTSi)，　　i = 1, 2, ……, n



5. 时间感知

- 主动实体：节点控制的自主对象
- 被动实体：无生命对象，需要对环境发出的事件作出反应
- 主动实体会与被动实体交互
- 需要根据最靠近的主动实体的网络延迟来绘制被动实体，以实现对最近主动实体的立即响应

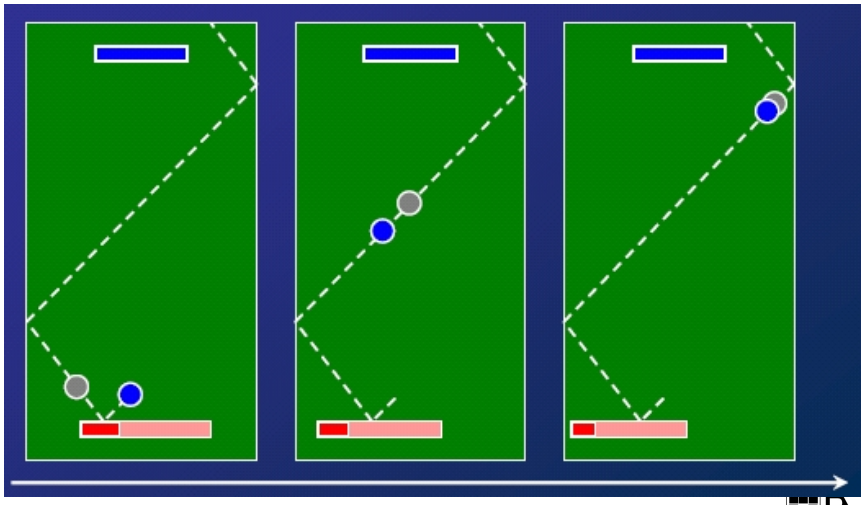
Example



- 两个主动实体：球拍
  - 运动不可预测
- 一个被动实体：壁球
  - 响应球拍，运动可预测
- 网络延迟为 d 秒
- **小球在最接近的用户节点上绘制**

33

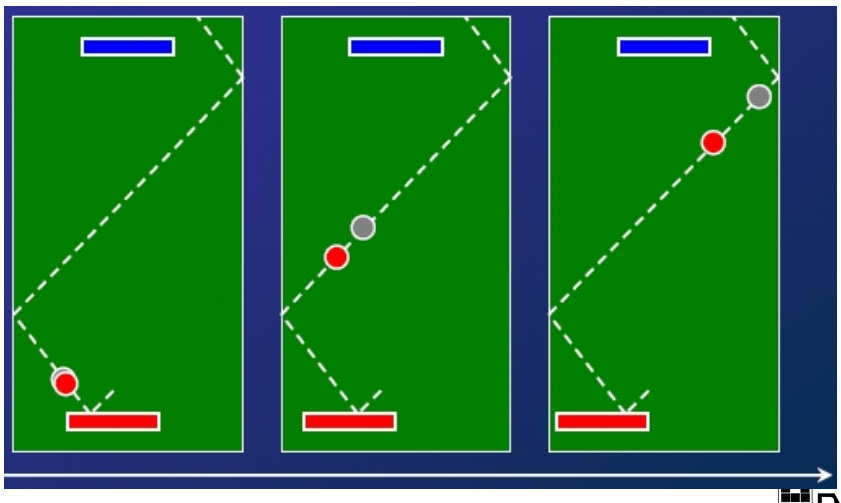
蓝方视角



www.hlarti.com

34

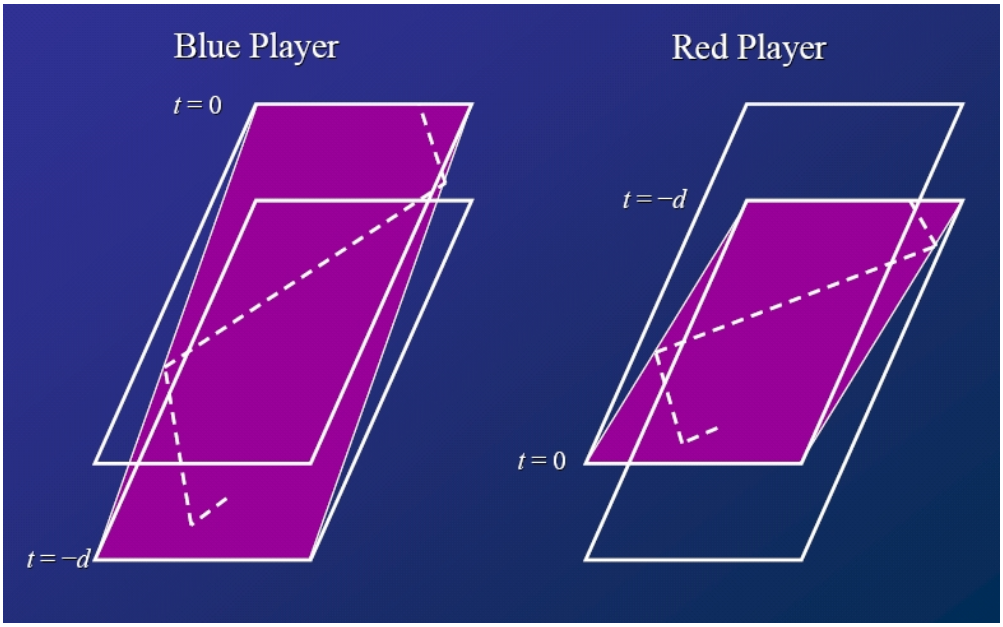
红方视角



www.hlarti.com

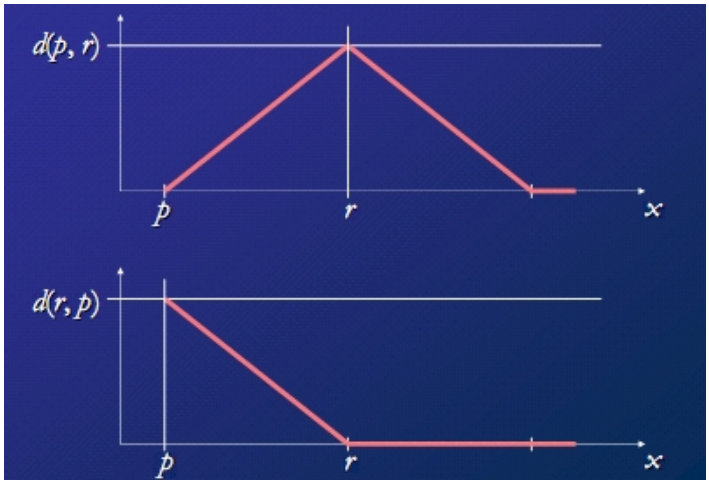
3 1/2维游戏区域

- 每个玩家的感知表示为(x,y,z,t)四维坐标系统
- x,y,z：空间坐标
- t：位置对应的绘制时间

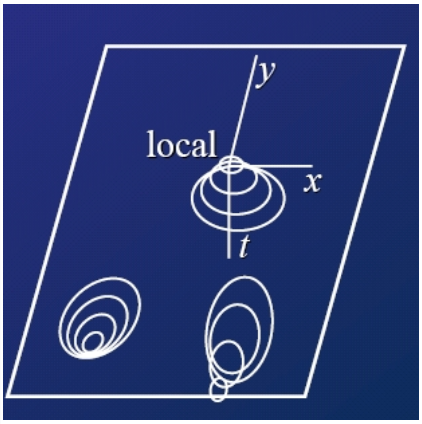


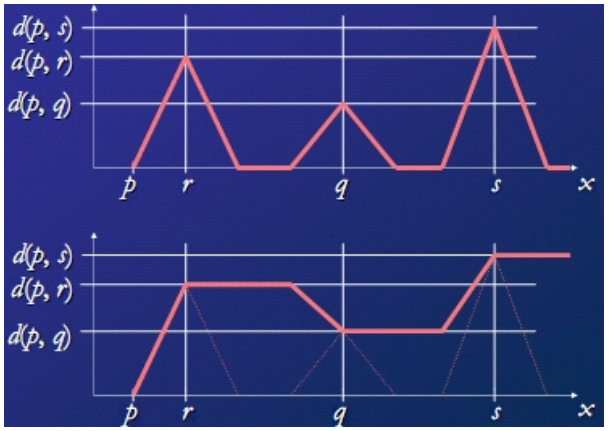
用户根据小球所在的t值来绘制位置。
通过这个思想可以实现球的连续移动

两种线性插值(全路径vs. 远程路径)



多用户NVE：时间等高线





相关问题

- 遥操作

www.hlarti.com

可靠性

背景

- 实时性 和 可靠性 是一对很难调和的矛盾
  - eg:TCP, UDP
- 可扩展性
  - TCP的拥塞控制策略过于保守

分布式虚拟环境中的通信特点

- 分布式虚拟环境中的通信特点
  - 多对多的组通信模型
  - 节点负载重
- 如何保证可靠？
  - 基于TCP
  - 可靠组播

基于TCP的应用层组播

- TCP Explode
  - 基于TCP实现可靠的组通信
  - 简单，却很实用
- 基于P2P的overlay multicast
  - 可扩展性好
  - 覆盖性和实时性较差

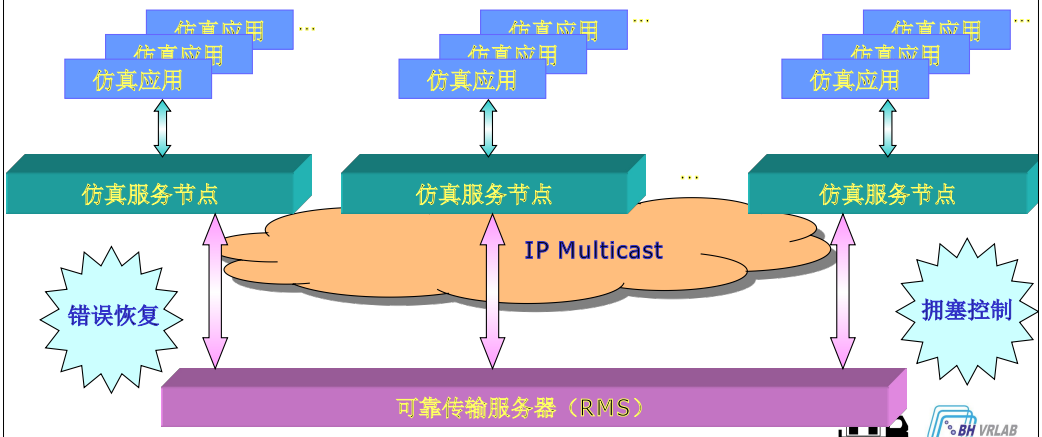
可靠组播协议

典型协议	研究机构	谁检测丢失、反馈方式	谁负责恢复	是否需知道组播拓扑	备注
SRM	University of Massachusetts	接收方、NAK	发送方	否	
RMTP	Purdue University	发送方、ACK	发送方、中间节点	动态维护	基于树状结构
TMTP	University of Kentucky	发送方、ACK	发送方、中间节点	静态事先指定	基于树状结构
FEC	University of Massachusetts	无	无	否	通过加入冗余信息进行恢复
PGM	已成为RFC	接收方、NAK	活跃路由器	路由器已知	基于路由器

组播，包括可靠组播是和应用密切相关的。

可靠组播服务模型

- 多对多 ⟹ 多对一



总结

- 事件一致性
- 时间一致性
- 可靠性