Skywind Inside » 帧锁定同步算法

帧锁定算法解决游戏同步

早期 RTS，XBOX360 LIVE游戏常用同步策略是什么？格斗游戏多人联机如何保证流畅性和一致性？如何才能像单机游戏一样编写网游？敬请观看《帧锁定同步算法》

《帧锁定同步算法》转载请注明出处：http://www.skywind.me/blog/archives/131

算法概念

该算法普遍要求网速RTT要在100ms以内，一般人数不超过8人，在这样的情况下，可以像单机游戏一样编写网络游戏。所有客户端任意时刻逻辑都是统一的，缺点是一个人卡机，所有人等待。

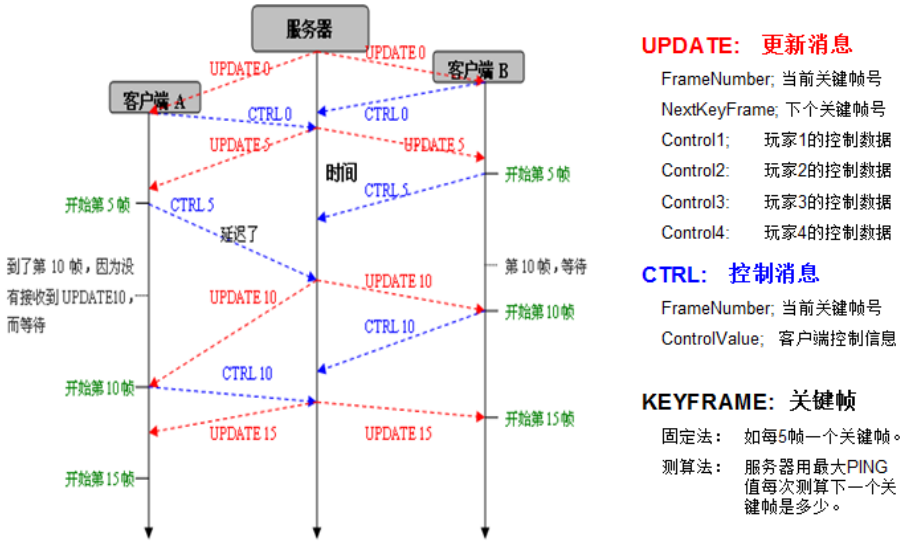
1．客户端定时（比如每五帧）上传控制信息。
2．服务器收到所有控制信息后广播给所有客户。
3．客户端用服务器发来的更新消息中的控制信息进行游戏。
4．如果客户端进行到下一个关键帧（5帧后）时没有收到服务器的更新消息则等待。
5．如果客户端进行到下一个关键帧时已经接收到了服务器的更新消息，则将上面的数据用于游戏，并采集当前鼠标键盘输入发送给服务器，同时继续进行下去。
6．服务端采集到所有数据后再次发送下一个关键帧更新消息。

这个等待关键帧更新数据的过程称为“帧锁定”
应用案例：大部分RTS游戏，街霸II(xbox360)，Callus模拟器。

算法流程

客户端逻辑：
1. 判断当前帧F是否关键帧K1：如果不是跳转（7）。
2. 如果是关键帧，则察看有没有K1的UPDATE数据，如果没有的话重复2等待。
3. 采集当前K1的输入作为CTRL数据与K1编号一起发送给服务器
4. 从UPDATE K1中得到下一个关键帧的号码K2以及到下一个关键帧之间的输入数据I。
5. 从这个关键帧到下 一个关键帧K2之间的虚拟输入都用I。
6. 令K1 = K2。
7. 执行该帧逻辑
8. 跳转（1）

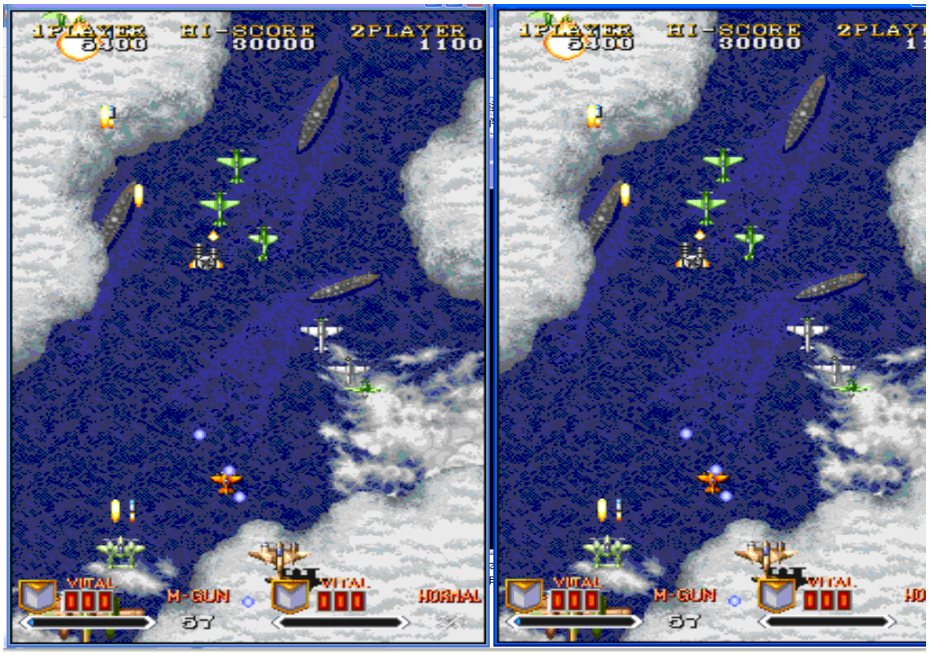
服务端逻辑：
1． 收集所有客户端本关键帧K1的CTRL数据（Ctrl-K）等待知道收集完成所有的CTRL-K。
2． 根据所有CTRL-K，计算下一个关键帧K2的Update，计算再下一个关键帧的编号K3。
3． 将Update发送给所有客户端
4． 令K1=K2
5． 跳转（1）



服务器根据所有客户端的最大RTT，平滑计算下一个关键帧的编号，让延迟根据网络情况自动调整。

算法演示

我根据该算法将街机模拟器修改出了一个可用于多人对战的版本，早期有一个叫做kaillera的东西，可以帮助模拟器实现多人联机，但是并没有作帧锁定，只是简单将键盘消息进行收集广播而已，后来Capcom在PSP和360上都出过街霸的联网版本，但是联网效果不理想。这个算法其实局域网有细就经常使用了，只是近年来公网速度提高，很容易找到RTT<50ms的服务器，因此根据上述算法，在平均RTT=100ms（操作灵敏度1/10秒），情况下，保证自动计算关键帧适应各种网络条件后，就能够像编写单机游戏一样开发网游，而不需状态上作复杂的位置/状态同步。



从上图的演示中可以看到，两个模拟器进程都在运行1941这个游戏，两边客户端使用了该算法，将逻辑统一在一个整体中。

最后这张图是运行KOF99的效果图，两边完美同步。

乐观帧锁定

针对传统严格帧锁定算法中网速慢会卡到网速快的问题，实践中线上动作游戏通常用“定时不等待”的乐观方式再每次Interval时钟发生时固定将操作广播给所有用户，不依赖具体每个玩家是否有操作更新：

1. 单个用户当前键盘上下左右攻击跳跃是否按下用一个32位整数描述，服务端描述一局游戏中最多8玩家的键盘操作为：int player_keyboards[8];

2. 服务端每秒钟20-50次向所有客户端发送更新消息（包含所有客户端的操作和递增的帧号）：

update=（FrameID，player_keyboards）

3. 客户端就像播放游戏录像一样不停的播放这些包含每帧所有玩家操作的 update消息。

4. 客户端如果没有update数据了，就必须等待，直到有新的数据到来。

5. 客户端如果一下子收到很多连续的update，则快进播放。

6. 客户端只有按键按下或者放开，就会发送消息给服务端（而不是到每帧开始才采集键盘），消息只包含一个整数。服务端收到以后，改写player_keyboards

————

虽然网速慢的玩家网络一卡，可能就被网速快的玩家给秒了（其他游戏也差不多）。但是网速慢的玩家不会卡到快的玩家，只会感觉自己操作迟迟而已。另一个侧面来说，土豪的网宿一般比较快，我们要照顾。

随机数需要服务端提前将种子发给各个客户端，各个客户端算逻辑时用该种子生成随机数，另外该例子以键盘操作为例，实际可以以更高级的操作为例，比如“正走向A点”，“正在攻击”等。该方法目前也成功的被应用到了若干实时动作游戏中。

指令缓存

针对高级别的抽象指令（非前后可以覆盖的键盘操作），比如即时战略游戏中，各种高级操作指令，在“乐观帧锁定”中，客户端任何操作都是可靠消息发送到服务端，服务端缓存在对应玩家的指令队列里面，然后定时向所有人广播所有队列里面的历史操作，广播完成后清空队列，等待新的指令上传。客户端收到后按顺序执行这些指令，为了保证公平性，客户端可以先执轮询行每个用户的第一条指令，执行完以后弹出队列，再进入下一轮，直到没有任何指令。这样在即时战略游戏中，选择 250ms一个同步帧，每秒四次，已经足够了。如果做的好还可以象 AOE一样根据网速调整，比如网速快的时候，进化为每秒10帧，网速慢时退化成每秒4帧，2帧之类的。

————

PS：可以把整段战斗过程的操作和随机数种子记录下来，不但可以当录像播放，还可以交给另外一台服务端迟迟验算，还可以交给其他空闲的客户端验算，将验算结果的 hash值进行比较，如果相同则认可，如果不通则记录或者处理，服务端如果根据游戏当前进程加入一些临时事件（比如天上掉下一个宝箱），可以在广播的时候附带。

（完）