第33卷 第17期 计 算 机 工 程 2007年9月
Vol.33 No.17 Computer Engineering September 2007
·开发研究与设计技术· 文章编号：1000—3428(2007)17—0280—03 文献标识码：A 中图分类号：TP393

基于双边假设检验的NEO改变帧长度机制

石祥滨[1,2]，刘 芳[2]，赵 鑫[2]
(1. 沈阳航空工业学院计算机学院，沈阳 110034；2. 辽宁大学信息科学与技术学院，沈阳 110036)

摘 要：使用双边假设检验来检测更新信息到达时间间隔的变化情况，根据当前网络状况精确计算帧长度调整值以改进原NEO协议简单比较及粗略调整帧长度的缺陷。将该协议中改变帧长度的投票权转交给游戏引擎，避免强制玩家参与投票。实验结果表明，改进后的帧长度随网络状况平稳变化，更接近真实网络状态，在一定程度上提升了协议性能。

关键词：网络游戏；欺骗；NEO；双边假设检验

Adjusting Frame Duration Mechanism of NEO Based on Two-sided Hypothesis Test

SHI Xiang-bin[1,2], LIU Fang[2], ZHAO Xin[2]
(1. Department of Computer Science and Engineering, Shenyang Institute of Aeronautical Engineering, Shenyang 110034;
2. School of Information Science and Technology, Liaoning University, Shenyang 110036)

【Abstract】 This paper uses two-sided hypothesis test to check the change of update arriving interval and computes the adjusting value of frame duration accurately according to current network condition, so that the deficiency of original NEO protocol's simple comparing and imprecise adjusting is improved. The voting right is delivered to game engine to avoid forcing players to vote. Simulation results show that the improved frame duration can be adjusted to adapt to network condition steadily which is closer to real network condition, and the protocol performance is enhanced to some extent.

【Key words】 online games; cheating; new event ordering (NEO); two-sided hypothesis test

1 概述

MMOG(massively multiplayer online games)不仅改变了传统计算机游戏的玩法，也从根本上改变了对计算机游戏安全性的要求。不同于传统的网络安全，网络游戏有其特有的安全问题——欺骗[1]。欺骗不仅降低了游戏的可玩性，也威胁到游戏开发者的收入。

传统的MMOG都是基于C/S模式的，由于游戏同步、兴趣管理等都需要服务器集中控制，因此可伸缩性以及单点失败是C/S模式固有的问题。P2P作为一种分布式计算模式可以提供很好的伸缩性、减少信息传输的延迟及消除服务器端瓶颈。P2P结构可以解决C/S模式中已有的问题，但是由于其开放特性也使欺骗问题日益严峻。MMOG的安全方案主要有防欺骗和欺骗检测两类。防欺骗主要在游戏协议中嵌入安全策略，安全性较高；欺骗检测相对简单，只需检测某些特定参数或者玩家行为是否合法即可。

文献[2]提出了分布式网络游戏中的第一个防欺骗协议：LS协议，该协议只能防止隐藏正确信息和预见未来欺骗，且行动提交严格按照两阶段提交进行，游戏按最慢玩家的速度前进，性能较差。为此，文献[2]又提出了AS协议，该协议在LS协议中添加了SOI，在一定程度上提高了LS协议的性能。但由于AS协议仍然使用两阶段提交，不能彻底改善性能。与LS协议相关的防欺骗协议还有PLS和Ghost等。这些协议分别从不同角度改进了LS协议，并能在一定程度上缓解LS协议的性能缺陷，但仍没能从根本上解决问题。文献[3]提出的NEO(new event ordering)协议则不同于LS协议。NEO通过加密而不是两阶段提交保证更新信息的安全性。NEO的安全性能要高于AS，能防止5种类型的欺骗，其他性能也更优越。

尽管NEO协议有上述优点，但也存在缺陷。它允许玩家动态改变游戏参数以实时适应网络状况的变化。游戏参数的动态改变通过强制玩家手动投票实现。但是强制投票影响了游戏的响应性和可玩性。因此，本文将投票权移交给游戏引擎，该引擎使用双边假设检验计算更新信息到达时间间隔的变化程度。当变化显著时，发起改变帧长度的投票，如有半数以上终端同意时依据当前网络状况改变帧长度。投票完全由游戏引擎自行完成，无需玩家参与。此外，与本文相关的MMOG中同步等问题已经在文献[4]中得到了解决。

2 NEO协议简介[3]

(1)基本NEO协议

游戏按帧进行，每一帧玩家向其他玩家发送更新信息。NEO使用帧限制玩家发送更新信息的最大延迟，迟到的更新信息无效。f是每一帧的长度，由游戏开人员自行设定。第r帧中玩家A的更新信息格式为

$$M_A^r = E(S_A(U_A^r)), K_A^{r-1}, S_A(V_A^{r-1})$$

基金项目：辽宁省自然科学基金资助项目(20052007)；辽宁省教育厅攻关计划基金资助项目(2004D116)

作者简介：石祥滨(1963－)，男，博士、教授，主研方向：分布式系统，网络游戏，数据库技术；刘 芳、赵 鑫，硕士研究生

收稿日期：2006-09-27 **E-mail：**sxb@lnu.edu.cn

其中，$E(x)$ 表示将 x 加密；$S_A(x)$ 表示 A 对 x 的签名；U_A^r 表示来自 A 的第 r 帧的更新信息；K_A^{r-1} 是 A 的第 $r-1$ 帧更新信息的密钥；V_A^{r-1} 是 A 投票决定第 $r-1$ 帧的投票结果位向量。更新信息的有效性通过分布式投票机制判断。如果某一玩家的更新信息及时到达，对该玩家投支持票，否则，投反对票。当且仅当大多数玩家对该更新信息投支持票时才接收该更新信息。每一帧玩家都计算所接收到的选票并决定各个更新信息的有效性。使用投票机制等待时间最长为 f，能有效减少延迟。当大部分玩家都接收到了更新信息和投票，NEO 就开始下一帧。

(2)游戏参数动态改变机制

为了更好地适应网络特性、解决网络拥塞，NEO 协议动态调整帧长度和更新信息发送频率。NEO 使用分布式投票决定是否调整，投票由玩家完成。动态改变帧长度可以提高对网络状况的反应能力，但增加资源开销。为了鼓励玩家参与投票的调整，投票信息被关联到更新信息数据包中，这样可以防止玩家忽略投票。因为忽略就是更新信息发送的停止。

1)调整帧长度：每一帧玩家发送更新信息。从该玩家的角度看，更新信息早到就可以缩短帧长度，而延迟则应增加帧长度。NEO 度量更新信息到达所用时间间隔，使用过去几帧的平均值判断是否需要调整帧长度。每个玩家根据自己的网络状况投票决定是否改变帧长度。

2)调整发送速率：NEO 协议衡量每个终端的丢包率和其他终端的迟到信息数。客户端通过故意丢掉对某一玩家的更新信息局部调整发送速率。客户端记录其他玩家丢包率的平均值，当大多数玩家投票对全局发送速率进行调整时，调整全局发送速率。

(3)NEO 协议的缺陷

NEO 协议中游戏参数的动态改变通过玩家投票决定。如果把游戏看作公共物品，按照奥尔森搭便车理论，在 NEO 的投票中，玩家越多，投票的自觉性就越差。这就是 NEO 投票的困境。NEO 采用强制方法解决这一问题。硬性强制和关联还会使玩家反感，降低游戏的可玩性。

3 强制投票的改进

本文将投票权转交给游戏引擎，避免了 NEO 协议改变游戏参数时玩家不能自觉投票的缺陷。本文使用概率论中双边假设检验计算更新信息到达时间间隔的变化情况。当网络情况变化显著时游戏引擎可以自行投票决定是否改变游戏参数，不需要玩家参与决策。

(1)投票机制的改进

各个终端实时记录更新信息到达时间间隔的变化情况。当某一终端通过计算发现更新信息的实际到达时间与游戏帧长度 f 相比变化显著时，发起投票。其他终端接收到投票后根据本机实际情况进行投票。如果有半数以上终端同意则根据当前网络状况改变帧长度。

以太网中数据包到达过程(时间 t 内到达 k 个帧的概率)服从泊松分布。由概率论可知，数据包到达的时间间隔服从指数分布，即数据到达所需时间的概率密度和分布函数分别为 $\varphi(t)=\lambda e^{-\lambda t}, t>0$ 和 $P(t)=1-e^{-\lambda t}, t>0$。

更新信息的实际到达时间间隔为 d (有效更新信息到达时间间隔统计均值)，则 $d\sim e(\lambda)$。更新信息到达时间的数学期望和标准差均为 $\frac{1}{\lambda}$。其中，λ 值由极大似然法获得，计算过程如下：

似然函数为 $L=\prod_{i=1}^{n}\lambda e^{-\lambda d_i}=\lambda^n e^{-\lambda\sum_{i=1}^{n}d_i}$

取对数 $\ln L=n\ln\lambda-\lambda\sum_{i=1}^{n}d_i$，微分可得 $\frac{\mathrm{d}\ln L}{\mathrm{d}\lambda}=\frac{n}{\lambda}-\sum_{i=1}^{n}d_i=0$；

因此，λ 的极大似然估计值为 $\lambda=\frac{n}{\sum_{i=1}^{n}d_i}=\frac{1}{\bar{d}}$。

如果将样本容量 n 取得充分大($n\geq 50$)，则根据列维定理统计量 $\mu=\frac{\sum_{i=1}^{n}d_i-nEf}{\sqrt{Df}\sqrt{n}}$ 近似服从正态分布 $N(0,1)$。

下面给出验证是否需要改变帧长度的过程。取 μ_0 值为当前帧长度 f；显著性水平 α =0.01(α 为假设检验中的一个临界概率值，通常取 0.01 或者 0.05，本文暂时取 0.01)；取 n 为 50 帧。

本文提出的 2 个对立假设如式(1)所示。

$$H_0:\mu=\mu_0=f;H_1:\mu\neq\mu_0 \tag{1}$$

检验更新信息到达时间间隔的计算过程如下：

更新信息到达时间间隔平均值(样本均值)为

$$\bar{d}=\frac{1}{50}\sum_{i=1}^{n}d_i$$

统计量为

$$|\mu|=\frac{|\bar{d}-\mu_0|}{\sqrt{Df}/\sqrt{50}}=\frac{|\bar{d}-f|}{\frac{1}{\lambda}/\sqrt{50}}=\sqrt{50}\lambda|\bar{d}-f|=\frac{\sqrt{50}|\bar{d}-f|}{\bar{d}}=\sqrt{50}\left|1-\frac{f}{\bar{d}}\right|$$

当 $\alpha=0.01$，按照 t 分布，$\mu\alpha/2=\mu_{0.005}=2.58$。当式(2)成立时认为 H_0 是错误的。

$$P(|\mu|>2.58)=P(\sqrt{50}\left|1-\frac{f}{\bar{d}}\right|>2.58)=\alpha=0.01 \tag{2}$$

如果计算证明 H_0 是错误的，则发起投票改变帧长度。投票信息附加在更新信息之后。对投票进行编号，如果有多个玩家使用同样的编号发起投票，则只响应最先接收到的投票。投票信息同样发给服务器(用于记录玩家静态信息和系统静态信息)。当有半数以上的客户端同意改变帧长度时，由服务器取各个玩家 $\bar{d}$ 的平均值作为新的帧长度。若 μ 为正，帧长度增加，为负，则减小。

投票算法

首先给出一些约定：投票信息同样放在更新信息中，新的更新信息格式如下：

$$M_A^r=E(S_A(U_A^r)),K_A^{r-1},S_A(V_A^{r-1}),\ VS,Vsn,V$$

其中，VS (voting sign)为是否包含投票信息的标志位；置 11 表示包含改变帧长度的投票信息；置 01 表示包含改变全局发送速率的投票信息；V_{sn} 为投票信息的序号；V 为帧长度改变方向标志位，为 1 表示 f 增加，为 0 表示 f 减小；f_{sn} 表示帧序号；Ny 为投赞成票的人数；N 为所有玩家数。

发起投票的算法

When(f_{sn} %50==1)

{ if($\sqrt{50}\left|1-\frac{f}{\bar{d}}\right|>2.58$)

{ VS =11; $V_{sn}=\lfloor fsn/50\rfloor$; if($(\bar{d}-f)>0$) V =1; else V =0; send M_A^r to other peers and server;}}

接收到投票信息的处理算法

if(M_i^r。VS ==11 & &V_{sn}!= M_i^r。V_{sn})

{ V_{sn}=$\lfloor f_{sn}/50 \rfloor$;

if($\sqrt{50}\left|1-\frac{f}{\bar{d}}\right| > 2.58$)

{if(M_i^r。V & &$(\bar{d}-f) > 0$) send yes and $\bar{d}$ to server;

else if($!M_i^r$。V & &$(\bar{d}-f) < 0$) send yes and $\bar{d}$ to server;

else send no to server; }}

服务器端的投票处理算法

if($Ny/N > 0.5$) $\bar{d} = \sum_{i=1}^{N} \bar{d}_i$; send $\bar{d}$ to all the peers

(2)调整发送速率的完善

原 NEO 协议通过跳过更新信息实现发送速率的调整。本文不直接跳过更新信息，而是将本来应该忽略的更新信息中的主要信息聚合在下一正常发送的数据包中，实现发送速率的调整。虽然迟到的更新信息无效，但这些信息可用于减小不一致性的发生。采取与调整帧长度相同的投票机制，由游戏引擎决定是否调整。

(3)改进后安全性分析

NEO 协议能够防止 5 种基本类型的欺骗。固定延迟欺骗和时间戳欺骗：通过限定帧长度实现；丢失更新信息欺骗：通过调整发送速率实现；不一致性欺骗：通过数字签名和状态比较解决；共谋：通过协议保证大多数人足够多，并加强 ID 的安全措施实现。

原 NEO 协议在一定程度上能够防止共谋欺骗的发生。改进后由于不需要玩家参与投票，则从投票层面上杜绝了共谋的发生，在防共谋欺骗问题上取得了较大进展。调整发送速率时，聚合方式的采用不但解决了拥塞问题，也减少了不一致性的发生。

由于强制投票影响玩家游戏，影响游戏的可玩性和响应性，因此本文使用双边假设检验计算更新信息到达时间间隔的变化。当更新信息到达时间变化确实具有统计意义时，由游戏引擎自行进行投票，而不用玩家参与。这样既能完成帧长度的改变，也可以减少玩家投票引发的延迟，或者游戏暂停。新的方法能够比较精确地计算出信息到达时间间隔的差异，实时反应网络变化，并精确修改游戏帧长度，弥补了原协议的不足。

4 性能测试与分析

本文使用网络仿真软件 NS-2 对 NEO 及其改进协议在调整帧长度方面的性能进行仿真实验。首先向 NS-2 的构件库中添加各个协议的定义，并设置必要的代理和链路连接。然后生成测试所需的星形拓扑。3 个玩家之间的通信延迟为 50ms，一个玩家的通信延迟为 225ms。玩家采用多播方式直接通信。本文的主要测试帧长度的变化稳定和真实特性。初始帧长为 100ms，帧长上限为 300ms，下限为 50ms；增加时帧长为 50 的倍数，减小时由附近整 10 向上取整。每个曲线均由 50 次实验的平均值绘出。

(1)网络延迟线性增加时帧长度的变化

首先比较网络延迟不断增加时 NEO 及其改进协议所表现出的延时特性。如图 1 所示，初始时原 NEO 协议的反映比较灵敏，因为原协议每 10 帧改变一次帧长度，而改进后的协议 100 帧才更新一次。但是随着时间的增加，改进后协议的优势非常明显：在原协议下，可以看到帧长随网络情况的变化抖动剧烈，变化幅度较大；而在改进后协议下帧长变化比较缓和，更接近于真实网络状况。由于本文使用双边假设检验验证网络变化情况，并根据当前网络情况计算出适当的帧长度而不是向原协议一样粗略选择，因此改进后的协议表现出较好的性能。

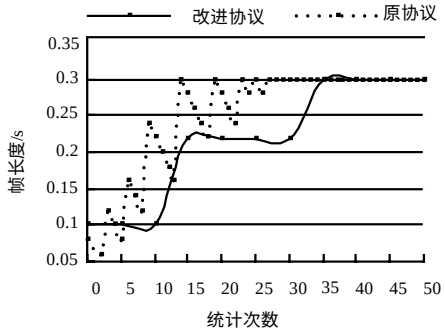

图 1 网络延迟线性增加时帧长度的变化

(2)网络延迟按泊松分布变化时帧长度的变化

在第 2 个实验中，可比较网络延迟按泊松分布变化时帧长度的变化情况。由图 2 可知，在原 NEO 协议下，帧长随网络变化上下波动比较剧烈，而且一直在高处徘徊、振荡。帧长如此抖动显然会影响游戏性能；而在改进后的协议中，帧长随着网络延迟平缓变化，基本上能反映并适应网络情况，有效提升了协议稳定性。在实际情况下，帧长度的变化将更加平稳。

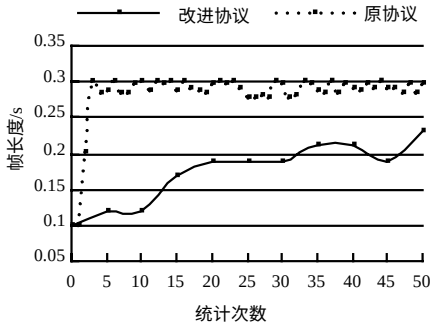

图 2 网络延迟服从泊松分布时长度的变化

由试验可知，在改进后的协议下，帧长度随网络变化适度调整，而且调整值通过精确计算而不是像原协议那样粗略地选择得出，保持了较好的特性。

5 小结

本文改进了 NEO 协议的改变帧长度机制。该机制将投票权转交给游戏引擎，并使用双边假设检验验证网络变化情况。试验证明，改进后的协议有效避免了原协议的缺陷，提高了协议性能及安全性。在今后的工作中应继续完善改变发送速率机制，尤其是数据包聚合机制的应用。

参考文献

1 Yan J, Randell B. A Systematic Classification of Cheating in Online Games[C]//Proc. of the 4th ACM SIGCOMM Workshop on Network and System Support for Games, Hawthorne, NY. 2005: 1-9.

2 Baughman N, Levine B. Cheat-proof Playout for Centralized and Distributed Online Games[C]//Proc. of the 12th IEEE INFOCOM Conference, Anchorage, Alaska. 2001: 104-113.

3 GauthierDickey C, Zappala D, Lo V, et al. Low Latency and Cheat-proof Event Ordering for Peer-to-Peer Games[C]//Proc. of the 14th International Workshop on Network and Operating Systems Support for Digital Audio and Video, NY, USA. 2004: 134-139.

4 石祥滨, 周兴海, 高 鹏, 等. 一种基于事件关联的 Timewarp 算法[J]. 小型微型计算机系统, 2006, 27(11): 2067-2071.