6R-5

Peer to Peer 型ネットワークゲームにおける不正処理防止方法の提案

萩原一昌† 十川基†† 斉藤裕樹† 戸辺義人†

†東京電機大学工学部情報メディア学科 ††工学研究科情報メディア学専攻
{kazu-h,hajime,hsaito,yoshito}@unl.im.dendai.ac.jp

1 はじめに

近年，大多数のプレイヤが参加するネットワークゲームが普及している．現在ではクライアントサーバ型が主流であるが，運営団体のコスト削減や安定性，拡張性向上等の様々な利点のため P2P 型化が注目されている．しかし P2P 型化することで運営団体が提供する信頼性のあるサーバが存在しなくなるため，課金や認証，不正防止等が問題である．本稿では P2P 型化することで起こりうる不正を，ノードを階層化することによって防止する手段を提案する．

2 P2P 型ネットワークゲームにおける不正対策

2.1 本研究で考える不正

ネットワークゲームにおける不正には様々なものがあるが，本稿ではデータパケットをサーバに送信することによって，ゲームデータを直接改竄し，通常プレイでは起こり得ないような不正ユーザにとって有利な状況を作りだす行為を不正行為とする．ネットワークゲームでは一人のプレイヤの行為がゲーム空間全体に影響するためこの問題を放置するとユーザに対するゲームの信頼感の低下を招く．

2.2 対象とするゲーム

本システムで扱う P2P ネットワークゲームは既存のゲームの種類としては大多数の参加者を許容し，1～10ms 単位での遅延を問題としない MMORPG(Massively Multiplayer Online Role Playing Game)を扱う．トポロジとしてはメッシュ型の P2P ネットワークではなく，オーバレイネットワークを使って作られたネットワーク上で動くハイブリット型の P2P ネットワークゲーム(図 1) [1] を扱う．具体的にはゲームデータを保存するストレージは DHT を使い各 P2P ノードに分散させ，データの計算処理はゲーム空間を小さなエリアに分けてゲームに参加しているノードがランダムで担当エリアを割り当て，部分的なサーバになることで実現する．このような P2P ネットワークゲームで不正を検出するための要件とは「部分サーバが不正を起す場合」である．プレイヤノードが不正を働く場合でも部分サーバが正常に不正チェックしているならばゲーム全体で不正が起こることはない．したがって以降はいかに部分サーバの不正を防止するかという話に限定する．

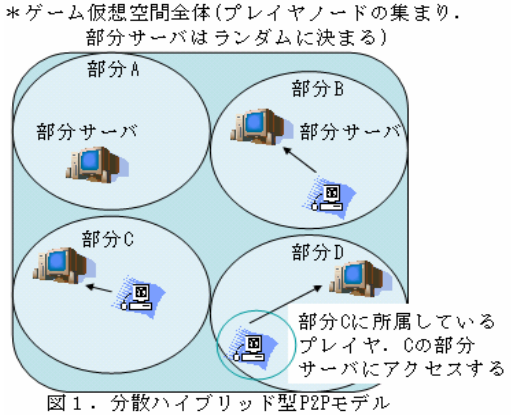

図１．分散ハイブリッド型P2Pモデル

2.3 不正を検出するための前提条件

1．正常なノードが存在し，すべての処理をこなす環境ならばチートは発生しない．

この場合を通常のゲーム機に例えると，コントローラの部分だけをプレイヤノードに任せる状態と同じで，いくらコントローラを改造してもメモリを改竄するような不正はできない．

2．不正をするノード数の割合は少ない．

正常なノード数のほうが不正を行うノード数よりも多いので，P2P ネットワーク上で正常なノードを協調動作させて不正をしているノードを発見する．

本システムはネットワークゲームを P2P 型化することによって，クライアントノード等必ずしも正常動作するとは限らないノードに任す演算部分を増やしたときに増加する不正発生数を極力減少させることを目的としたものである．

3 不正検出のための要求機能

必ず正常動作するサーバの存在しない P2P 環境において，不正を抑制し信頼性を向上させるシステムとして LAVER(leveled right of access for unreliable server)を提案する．

以下に LAVER の特徴を述べていく．

- 階層化機能

サーバは(図 2)のように階層機構を持っている．サーバノードの上にさらにサーバノードが存在している場合や，下位にサーバノードが存在している場合がある．この階層構造の規模はネットワークゲームに参加しているプレイヤ数等の規模によって変動する．

- 書き換え権限譲渡機能

書き換え権限を持っている上位サーバノードは下位サーバノードに対して分散データを書き換える権限を譲渡することができる．譲渡された後は権限が無効になるまで DHT に分散されたデータを書き換えることができるようになる．

- 多数決判定機能

書き換え権限を持つサーバには複数のクライアン

Cheat Protect Architecture for P2P Multi Player Network Game
† Kazumasa Hagiwara
‡ Hajime Sogawa
† Hiroki Saito
† Yoshito Tobe
Dep of Info. System and Multimedia Design. Tokyo Denki University (†)
Dep. of Info. and Media Engineering . Tokyo Denki University(‡)

トが接続している．そのノードは常にサーバが正しい動作をしているか判定していると仮定し，もしも不正を行っていると判定されたなら接続しているサーバのさらに上位のサーバに権限を剥奪して欲しいというメッセージを送信することが出来る．クライアントの半数以上がサーバノードを不正だと認識したなら上位サーバはサーバノードが不正を行っていると決定する．

- 書き換え権限剥奪機能

上位サーバによって不正だと決定したら，上位サーバは不正をしている下位サーバから書き換え権限を剥奪し，直ちに分散データに書き込めない様にする事が可能である．そしてサーバノードが離脱した場合に次期サーバとなるノードにサーバの交替を促す．

- データロールバック機能

不正サーバが既に書き換えてしまった改竄ゲームデータを正常な時点のデータに巻戻す機能．

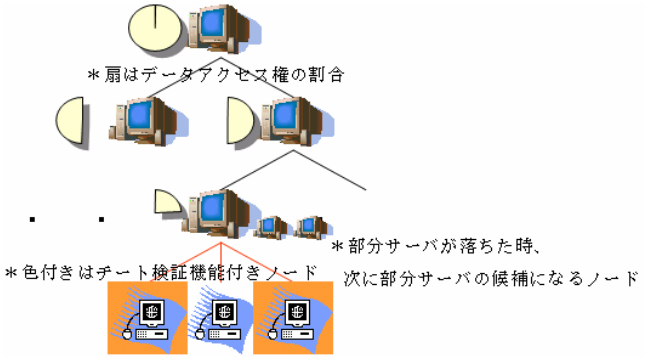

図２．書き換え権限を階層化したモデル

4 測定と評価

本システムを実装する上で重要な問題としてクライアントが接続するサーバが切り替わった瞬間に発生する遅延がゲームプレイ中にどの程度影響するかがある．そこでサーバが DHT ネットワークにゲームデータを保管しサーバが DHT ネットワークからゲームデータを取得する際に発生する遅延と， TCP コネクションを張り直す際に発生する遅延を測定した．

4.1DHT を利用してデータを引き継ぐ際に要する遅延

本稿で提案した DHT を使ってゲームデータを引き継ぐ際にかかる遅延を測定した．測定環境として DHT の実装には Bamboo[2]を使い DHT ネットワークには PlanetLab [3]上に構築されたネットワークを使用した．データをアクセスする前に DHT アクセスに要する時間を測定し，その結果データ量と DHT ネットワークの規模が大きくなってもアクセス時間がさほど増えずに済み，プロセス起動時間を含めても概ね 1～2s 程度でアクセス(put_t,get_t)出来ることが判った．

次にゲームに適用した測定としては(図 3.1)の様にサーバ切断命令の後に旧サーバが DHT にデータを保管して成功したら新サーバに DHT にデータを格納することに成功したと伝え，新サーバが DHT からゲームデータを取得してクライアントが接続させゲームを再開させる．この作業{保管＋取得}にかかる遅延を測定した結果 3～4s 程度かかるという結果になった．しかしこの時間はある程度短縮することが可能である．例えば図 3.2 のようにデータ保管成功まで待たなくても良く，その前に取得し始めてもアクセスは成功する．さらに旧サーバが切断される寸前のデータは信頼性がないものとみなし旧サーバはサーバ切断命令の直後にデータ保管作業をする必要はない(図 3.3)と考えられる．したがって旧サーバが切断する寸前のゲームデータを引き継ぐ場合にかかる時間は最適化できた場合(図 3.2)の{put_t+get_t 以下 get_t 以上}となる．さらに上位サーバからの権限剥奪により切断される場合，切断する寸前のゲームデータは改竄されている恐れがあるので，データを破棄して少し前にロールバックしたデータでも許されるので(図 3.3){ put_t+get_t 以下0以上}となる．この時間{put_t, get_t}はアルゴリズムの最適化と DHT の実装方法により変化する．

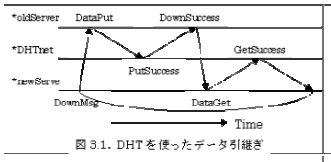

図 3.1. DHT を使ったデータ引継ぎ

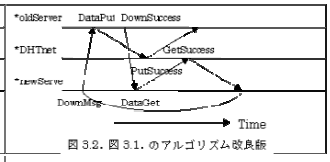

図 3.2. 図 3.1. のアルゴリズム改良版

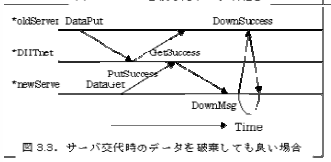

図 3.3. サーバ交代時のデータを破棄しても良い場合

4.2 TCP コネクション再確立に要する遅延

同一ネットワークにおいて 300ms 間隔でクライアントにデータを送るサーバを意図的に切断し，ゲームデータを引き継がずに新規サーバを立てクライアントが TCP コネクションを確立するために必要な時間を測定した．その結果，概ね 50～300ms 程度の時間で張りなおすことが出来た．最大で 300ms の遅延になってしまう理由は 300ms 間隔で送っていた影響であるので実際の TCP コネクション再確立に要する遅延は 50ms 程度である．つまりコネクションを張りなおすという点ではユーザがほとんど遅延を感じずにサーバを再確立が可能なことが確認できた．

5 まとめと今後の課題

本稿では LAVER の骨組みと，そのシステムの性能を決めるサーバ交替時に発生する遅延時間の点から実現可能性を考察した．今後の課題として本稿で提案したシステムに最適化したアルゴリズムの決定，適したトポロジを自動的に組めるような実装，待機部分サーバの冗長性の最適化を決定をする等がある．

参考文献

[1]Takuji Iimura, Hiroaki Hazeyama, Youki Kadobayashi　Zoned Federation of Game Servers: a Peer-to-Peer Approach to Scalable Multi-player Online games
ACM SIGCOMM 2004 Workshop on NetGames

[2]The Bamboo DHT http://bamboo-dht.org

[3]PlanetLab http://www.planet-lab.org