オーバレイネットワークによる 大規模マルチプレイヤオンラインゲームの開発

Development of Massive Multiplayer Online Game by Overlay Network

情報工学専攻　西川 憲三
NISHIKAWA Kenzo

要約: 現在の大規模なオンラインゲームはクライアント・サーバ型で構成されるのが一般的で, 処理がサーバに一極集中し運営が困難になっている. 本研究ではオーバレイネットワークを用いてオンラインゲームをユーザに分散管理させ, 大規模なオンラインゲームを開発する手法を提案する.

キーワード: オンラインゲーム, オーバレイネットワーク, 分散処理

1　はじめに

ネットワークのブロードバンド化によりオンラインゲームが一般的になった. 現在の大規模なオンラインゲームはクライアント・サーバ型で構成され, 大量のアクセスがサーバに一極集中し莫大な負荷がかかる. 運営は莫大な負荷に耐えうる広帯域なネットワークの整備や, 高い計算能力を持つサーバのクラスタリングなど非常にコストのかかる敷居の高いものとなっている.

本研究ではオンラインゲームをオーバレイネットワーク上のユーザマシンに分散し管理させ, 大規模なオンラインゲームの運営の敷居を下げることを目的とする.

2　前提知識

2.1　オンラインゲーム

オンラインゲームは, ユーザが操作するデータがゲームを共有している他の全てのユーザにリアルタイムに反映されるアプリケーションである. このリアルタイム性を実現するために, 短時間に細かなパケットを頻繁に通信することになるため, オンラインゲームは通信量に気を使う必要がある. オンラインゲームで扱うデータは, オフラインで管理するデータとネットワークで共有するデータ (Global Status Data=GSD) の 2 種類に分けられ, リソースや冗長な処理をユーザマシンにオフラインで管理させ, GSD を極力少なくすることによって通信量を減らすことができる. また, オンラインゲームは一貫性が重要である. どちらが早いのか, 強いのかといった甲乙の判定を公平に行い, どのユーザにも同じ結果を返す必要がある. この判定は 1 つのマシンが行うことによって実現できる.

2.2　分散ハッシュテーブル

オンラインゲームを従来のクライアント・サーバ型から Peer-to-Peer(P2P) での実装に変更できれば, 高価な中央サーバを必要としないため, 運営の敷居を下げることができる. しかし, 無造作に接続された P2P ではゲーム中に必要となるデータを管理している特定のノードを検索することが困難である. そこで, 分散ハッシュテーブル (Distributed Hash Table=DHT) を用いる. DHT は少ない接続数と検索ステップ数で全てのノードを検索することができる構造化オーバレイネットワークである. 中央サーバーを必要とせず, すべてのノードが小さい範囲のルーティングと, 小さい 1 セットのデータの記憶を担う. また, DHT は検索対象のノードを特定するために, ハッシュテーブルを用いて検索するキーに対応するハッシュ値を管理したノードを決定する. すべてのノードはキーに対して最も近い隣接ノードを知っており, バケツリレーにノードを経由して検索対象のノードを発見することができる. 図 1 に DHT アルゴリズムの一種である Chord の接続例を示す. Chord はノードを n ビットで表現する ID で割り振り, 昇順で時計周りにリング状に繋ぐ. ハッシュのキーをノードの ID と関連付けることで, それぞれのノードが管理するキーが一意に定まる. データを格納する際にキーをハッシュ化することによってハッシュ関数のランダム性によりキーはノードに均等に近い割合で割り当てられる. また, 検索に関してノードは自身の ID から $2^k (1 \leq k \leq n-1)$ 先の ID に対してショートカットリンクを持つことでノード数 n に対して $O(\log n)$ のホップ数での検索を実現する [4].

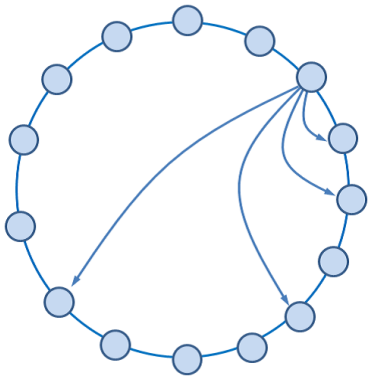

図 1　Chord

3　既存の研究

3.1　Zoned Federation Model(ZFM)

奈良先端科学大学院大学の飯村卓司氏による “オーバレイネットワークを用いた MOG インフラストラクチャの提案”[1] という研究がある. この研究は本研究と同様に P2P を用いてゲームを分散することを目的としている. 研究の中で Zoned Federation Model(ZFM)

を提案している．ZFM はゲーム全体の GSD を Zone という単位に分割し，ユーザに分散し管理させることでオンラインゲームを分散する．ユーザはゲーム中に必要となる Zone を管理するノードにアクセスすることでゲームをプレイすることができる．

3.2 ZFM の構成

Zone は DHT 上に分散して格納される．ゲームをプレイするノード (ユーザ) は Zone を管理するノードにアクセスし Zone に参加する．Zone に参加しているノードの中から Zone のデータの変更権限を唯一持つ Owner が 1 つ選ばれ，Owner は Zone 内のゲームの進行役を担う．それ以外のノードは処理を Owner に依頼する Member となる．Zone に参加ノードが存在しない場合，Owner は必要ないため存在せず，最初に参加するノードが Owner となる．それ以降に参加するノードは Member となり，Owner とネットワークで直接接続する (図 2)．そうすることでリアルタイム性を実現する．

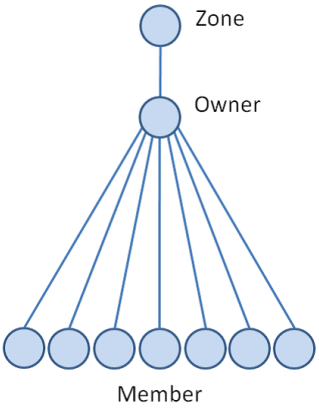

図 2　ZFM の構成

3.3 GSD の処理

Zone 内の GSD の処理はすべて Owner が行う (図 3)．Owner は 1 つなので，ゲームの一貫性が保たれる．

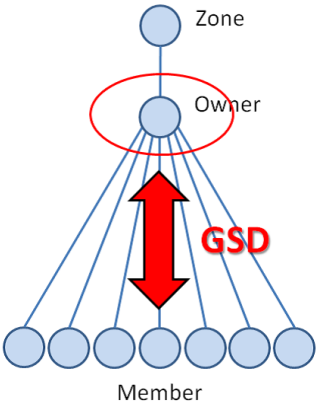

図 3　GSD の処理

3.4 ZFM の問題点

ZFM は Zone 内に小規模なクライアント・サーバを作り上げる．そのため ZFM では特定の Zone にノードの参加が一極参加した場合，Owner は処理することができない．特にイベントの発生する場所や，必ず通らなければならない場所などはこの現象が起こりやすいと考えられる．

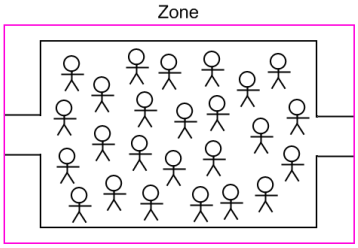

図 4　ひとつの Zone への一極集中

4 本研究

4.1 Two Hierarchical ZFM(2HZFM) の提案

ZFM で問題となっていた一極集中の原因は，Zone を 1 つのノード (Owner) が処理しているため，Zone の限界が 1 つのノードの処理量や通信量の限界と直結しているためだと考えられる．そこで本研究では，Zone を 2 層化して管理する Two Hierarchical ZFM(2HZFM) を提案する．2HZFM によって Zone に参加できる人数を増やすことで，より柔軟に大規模なオンラインゲームへの対応を可能にする．

4.2 GSD の分別

まず，2 層化する考えとして GSD を 2 つに分類する．1 つは，チャットや移動情報など，その場限りで共有する一時的なデータであり，Auto Global Status Data(AGSD) と呼ぶ．もう 1 つは，スコアやステータス情報などゲームの進行に永続的に影響を与えるデータであり，Static Global Status Data(SGSD) と呼ぶ．オンラインゲームでのプレイヤの動作は移動やチャットのような一時的なデータが大半を占めるので，GSD の大半は AGSD であり，SGSD は瞬間的なリアルタイム性を必要としないという特徴を持つ．

4.3 プレイヤのグループ化

次にプレイヤのグループ化を考える．一般にオンラインゲームではプレイヤは少数のグループで行動する．そこで，移動やチャットなどの AGSD はゲーム内で密接に関係しているグループ内でのみ通信されればよいと考え，全体に通信する SGSD と分離し，2 層化して管理する．

4.4 Zone の構成

2HZFM では Zone のデータを管理するノードは統括的な処理も行うことから Administrator と呼ぶ．まず，2HZFM は ZFM と同様に最初に参加するノードが Owner となり，後に参加するノードは Member として Owner に接続していく．そしてある程度参加人数が増えたところで新たに Owner を任命する．以降に参加す

るノードは Member として新たな Owner に接続していく．この動作を繰り返すことで，Zone 内のノードをグループ化する (図 5)．

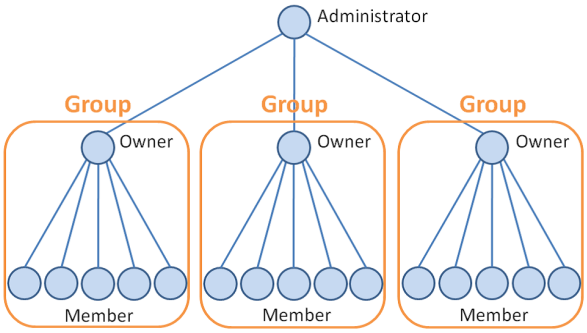

図 5　2HZFM の構成

4.5　GSD の処理

Zone 内の GSD の処理について説明する．　まず，AGSD はグループ間でのみ通信されれば良いので，それぞれのグループの Owner が独立して処理を行う．また，SGSD は全体に影響を与えるので，Member から Owner を通じて Administrator に送られ，Administrator が処理し，結果は逆に Owner を通じてすべての Member に伝えられる．つまり GSD を Administartor と Owner によって 2 層に管理する．それぞれの GSD は関係するノードに対して 1 つのノードが処理するので，ゲームの一貫性が保たれる．

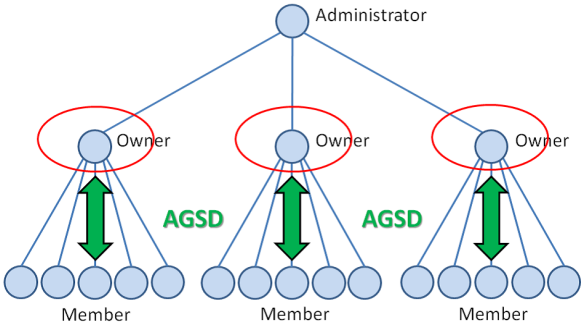

図 6　AGSD の処理

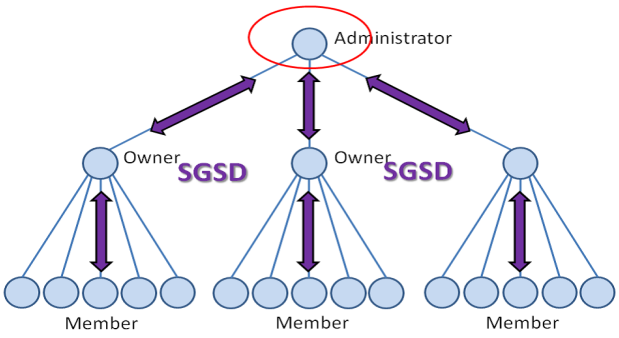

図 7　SGSD の処理

5　実験と評価

5.1　評価基準

2HZFM は Zone を 2 層化することによって ZFM よりも負荷分散に優れるが，リアルタイム性で劣ると考えられる．しかしオンラインゲームはいかに高速に数値計算を行うかというようなプログラムではなく，オンラインゲームに十分なリアルタイム性が実現されていれば良い．ゲームの応答時間に関する一連の研究がある (文献 [2, 3, 5, 6, 7])．これらの研究によるとオンラインゲームで人間がストレスを感じない応答時間は 200ms 程度が限界であるとされている．つまり，200ms 以下で応答し，最大限に負荷分散を行うモデルが理想の形であるといえる．2HZFM が ZFM よりもその理想の形に近いということを示す．

5.2　オンラインゲームの開発

オンラインゲームには様々なジャンルが存在し，ありとあらゆるゲームを開発し，2HZFM がそれらすべてに有効であると実証するのは不可能である．そこで本研究では実際にオンラインゲームを開発し，2HZFM が ZFM よりも負荷分散能力に優れ，十分なリアルタイム性を実現できるという例を示す．オンラインゲームは，ハドソン社のボンバーマンをモチーフにして開発した (図 8)．開発環境は Microsoft Visual C++，グラフィック描画 API は DirectX，ネットワーク通信 API は WinSock，DHT アルゴリズムは Chord を用いた．

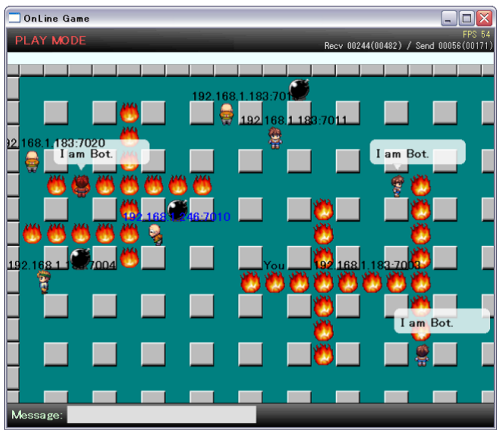

図 8　ゲームのスクリーンショット

5.3　シミュレーション環境

マシンは 10 台を用いる．すべてのマシンはシミュレーションでボトルネックとならない程の十分なスペックを有している．マシンは 100Base-TX で HUB で繋がれている．1 つのマシンで複数のプログラムを実行することで大規模なシミュレーションを行う．

5.4　Zone のシミュレーション

まず，1 つの Zone に対するシミュレーションを行う．Zone に 1 人ずつプレイヤを参加させ，通信量や応答時間がどのように変化していくかを計測する．プレイヤは 1 秒ごとに移動する Bot として起動し，安定した負荷を与える．2HZFM のグループの最大人数は 10 人に設定した．

5.4.1 Owner の負荷の評価

Owner の通信量 (送信, 受信の合計) を計測した結果が図 9 のようになる．Owner は Member の GSD に対して他のすべての Member に送信するので Member の数に対して 2 乗に比例して増加するが，2HZFM はグループに 10 人を越えると参加が次の Owner に移るので，1 つの Owner の通信量は 10 人で伸びなくなる．

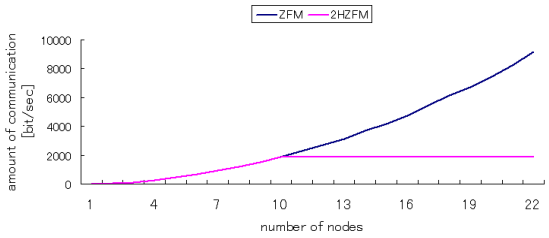

図 9　Owner の通信量

5.4.2 AGSD の応答時間の評価

AGSD の応答時間の評価の結果は図 10 のようになった．ボトルネックとなる Owner の処理が Member の数の 2 乗に比例して増加するので，応答時間も同様に伸びていく．2HZFM の場合は先ほどと同様に 10 人で応答時間が伸びなくなる．

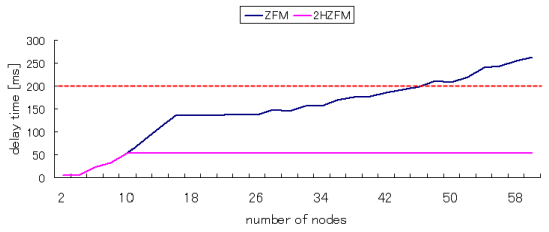

図 10　AGSD の応答時間

5.4.3 SGSD の応答時間の評価

SGSD の応答時間の評価の結果は図 11 のようになった．ZFM の SGSD は AGSD と同様に処理されるので AGSD の結果から変化しない．2HZFM の場合は Owner を通じて Administrator に通信され，Owner を通じて Member に返されるのでパスの長さが 2 倍になる．ところが結果は 10 人程度までは 2 倍に増加するが，Owner の負荷が 10 人で抑えられる 2HZFM に対して 2 乗に比例して Owner の負荷が増える ZFM はボトルネックとなって参加人数が 30 人程度で応答時間が逆転した．パスが 2 倍でも負荷が分散されることによって人数が増えるとかえって応答時間がはやくなるという結果になった．

5.5 ゲーム全体のシミュレーション

次にゲーム全体に対するシミュレーションを行う．ゲームにプレイヤが 100 人参加した状態でノードの負荷がどのように分布するかを測定する．より実際のプレイヤの操作に近い Bot として起動する．

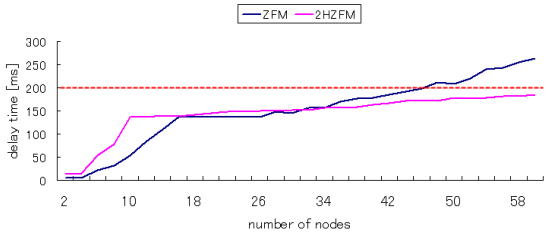

図 11　SGSD の応答時間

5.6 負荷分布の評価

負荷分布を計測した結果が図 12 になった．ZFM では Zone に大量のノードが参加した場合 Owner に負荷が偏るが，2HZFM の場合は 10 人に制限されるので Owner の負荷が抑えらる．結果から 2HZFM のほうが効率よく負荷が分散されていることがわかる．

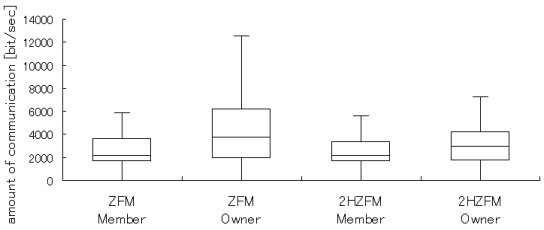

図 12　負荷バランス

6 研究の成果

研究の成果として ZFM を改良しプレイヤの一極集中に対応できる 2HZFM を提案した．2HZFM によってより柔軟に大規模なオンラインゲームに対応できることを可能にした．

参考文献

[1] 飯村卓司 **“オーバレイネットワークを用いた MOG インフラストラクチャの提案”** 奈良先端科学技術大学院大学情報科学研究科修士論文 2005 年 3 月.

[2] G.Armitage: Sensitivity of quake3 players to network latency. In ACM SIGCOMM Internet Measurement Workshop 2001 Work-in-progress posters Session, Nov.2001.

[3] G.J.Armitage: An experimental estimation of latency sensitivity in multiplayer quake3. In Proceedings of the 11th IEEE International Conference on Networks ICON 2003, 2003.

[4] I.Stoica, R.Morris, D.Karger, M.F.Kaashoek and H.Balakrishnan: Chord: A Scalable Peertopeer Lookup Service for Internet Applications Proceedings of SIGCOMM'01, pp.149-160, 2001.

[5] L.Pantel and L.C.Wolf: On the impact of delay on real-time multiplayer games. In Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, pp.23-29. ACM Press, 2002.

[6] P.Quax, P.Monsieurs, W.Lamotte, D.D.Vleeshauwer and N.Degrande: Objective and subjective evaluation of influence of small amounts of delay and jitter on a recent first person shooter game. In Proceedings of ACM SIGCOMM Workshop Network and System Support for Games NetGame-4, Aug. 2004.

[7] T.Beigbeder, R.Coughian, C.Lusher, J.Plunkett, E.Agu and M.Claypool: The effects of loss and latency on user performance in unreal tournament 2003. In Proceedings of ACM SIGCOMM Workshop Network and System Support for Games NetGames-4, Aug.2004.