

2010 年度冬の LA シンポジウム [S12]

仮想世界上での近距離合意問題

榎本慎太郎*

1 はじめに

近年ゲーム業界でネットワークを利用したマルチプレイヤーオンラインゲームが普及した。その中でも MMORPG (Massively MultiPlayer Online Role-Playing Game) というジャンルのゲームが確固としたゲームジャンルとして成長を遂げている。このゲームはコンピュータが提供する仮想現実内で複数のプレイヤー達がお互いに干渉しながら RPG を行うというゲームである。しかしクライアント/サーバモデルのアーキテクチャで作られているためにサーバに負荷が集中するのでスケーラビリティに問題がある。

そこで P2P 型の MMORPG という研究が進められている。これはサーバを使わずに P2P ネットワーク上で MMORPG を実現しようとする試みである。但しこの研究にはいくつかの課題がある [FTT10]。課題のひとつとして唯一性というものがある。これは各ノード間でゲーム進行に矛盾を発生させないために必要である。例えば一人にしか獲得できないアイテムがある場合、そのアイテムを複数人が獲得してしまったり、消滅してしまうことを避ける必要がある。つまりそのアイテムの所有者は唯一でなければならない。そこで分散システムの合意問題を解くアルゴリズムを応用したい。しかし、従来の合意問題では分散システム上全てのノードとのやりとりを必要とするためスケーラビリティが低い。

そこで、仮想現実内ではアイテム等に座標情報があることを利用してイベント発生地点の近距離にいるノードのみで合意問題を解決するプロトコルを提案する。

2 P2P 型 MMORPG

MMORPG を P2P ネットワークで行う場合、プレイヤーの処理以外は相互に協調して行う必要がある。MMORPG を P2P で実現する利点は負荷の一点集中をさけることができるのとサーバに掛かっていたコストを削減できることである。しかし実現のためにはサーバが行っていたことを P2P の分散システム上で実現しなければならない。具体的には以下のような処理である。

1. イベントの処理
2. ゲームデータの保存
3. クライアント間の仲介
4. NPC (敵モンスターや村人) の動きの演算

本研究ではこのうちイベントの処理にあたる部分を扱う。イベントとは、プレイヤーがアイテムを取る、特定のモンスターが出現する等である。本研究では特にアイテムを取るイベントについてを扱う。

アイテムを取るイベントとはアイテムの所有者を合意するイベントである。この時、アイテム所有者は必ず一人に合意しなければならない。そしてゲームとしての性質上、公平性を考慮し、不正が起こる可能性を極力減らす必要がある。これは分散システムの合意問題を解くアルゴリズムを使用することによって解決することができる。

3 近距離合意問題

従来では合意問題を解決しようとする分散システム上全てのノードに合意を取る必要があった。し

*東京電機大学大学院工学研究科

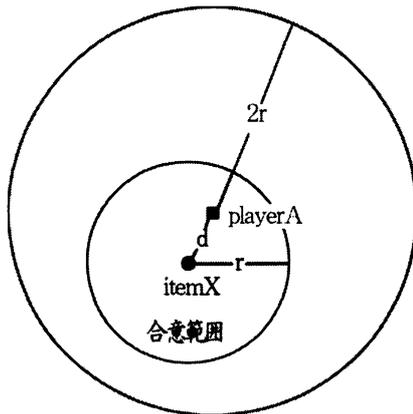


図 1: メッセージ送信範囲

かし、MMORPG は仮想現実として作られているのであり、同じ仮想世界にいても知らなくて関わらなくてよいイベントが存在する。たとえば、日本で100円を拾った人のことを、アメリカにいる人が知る必要はないのである。そこで仮想空間上でアイテムの近距離にいるノードだけで合意問題を解くプロトコルを提案する。

ここでは、二次元座標上のあるアイテムに対して、半径 r 内の Player たちが合意することを考える (図 1)。アイテムから半径 r の範囲を合意範囲と呼ぶ。しかし、アイテムは担当するひとつの Peer を持つことはなく、各 Player を持つ Peer のどれかが管理を担当する。そのため、アイテムから r 以内のノードを求めるばあい、その範囲に対して合意をとる必要が生じる。そこでイベントを発生するノードは自分から距離 $2r$ 以内にいるノードへメッセージを送ることとする。itemX とノードとの距離を d とするとイベントを起こす可能性のあるノードは必ず $d \leq r$ である。そしてノードの位置から最も遠いイベントを発生できる位置への距離は $d+r$ である。このことより $d+r \leq 2r$ であるので、 $2r$ という距離は合意範囲にいるノード全てにメッセージの届く十分条件である (図 1)。

さて、本研究はゲームなので特定のノードが有利になるようなアルゴリズムを使うことが好ましくな

い。さらに合意値はどのプレイヤーかに左右される心配のない有効なランダムな値であることが望ましい。従来の多数決や、一番小さい値に合意するというアルゴリズムでは不公平が生じる場合がある。そこで次のアルゴリズムを使用する。

[アルゴリズム] 各ノードはそれぞれ n 未満のランダムな二つの値 x, y を持つ。 x は抽選番号である。 y は当選番号を決めるのに使う。各ノードは合意に参加する全てのノードへ x, y を送信する。各ノードは自分を含む全てのノードの y の合計を求めて n で割った余りを求める。この余りが当選番号となる。当選番号以上で最も小さい抽選番号を持つノードが当選者となる。もし当選番号以上の抽選番号を持つノードがなければ、最も小さい抽選番号を持つノードが当選となる。この時、各ノード間で抽選番号と当選番号が共有されていれば矛盾は起きない。次にプロトコルを示す。

[プロトコル 1]

1. 各ノード i はランダムに $0 \leq x_i, y_i \leq n$ を選び、全ノードに送信する。
2. $Y = \sum y_i \pmod{n}$ とする
3. Y を超える最も小さい x_j 、なければ最小の数を送ったノードを選ぶ。

n は十分大きいこととする。なお、この n は参加者数を m とした場合、 $n > m^3$ とすると、十分高い確率で重複が起こらない [BZ89]。

但し、これは非同期なシステムで考えるとき、最後に x_i, y_i を選んだ方が有利となるため工夫が必要である。

さて、プロトコル 1 を応用して近距離合意問題を解くプロトコルを提案する。プロトコルの前提条件は次のようにする。

1. 大域時計を使用できる。
2. ノードがどのタイミングで合意範囲に出入りしてもよい。

3. 各 Player は半径 $2r$ 内の Player とやりとりできる。

この前提で次の合意問題を解く。

1. どのノードが途中でプロトコルを止めても、合意範囲内にノードが一つでも残っていれば合意ができる。
2. 途中でノードが参加しても合意できる。しかし、有利にはならない。
3. 途中で止めた人に合意することはない。

さて、まず初めに、次のプロトコルを示す。

[プロトコル 2]

立候補フェイズ アイテムを取得したい Player は次のことを行う。

1. ランダムな値 x, y を決定する。
2. 公開鍵 $e(\cdot)$ と他のノードが応答するのに十分な時間が開くように締切時刻 dl を決定する。
3. 立候補メッセージとして次のメッセージを半径 $2r$ 内のノードに送る「プレイヤー ID, アイテム ID, 送信時刻, $e(\cdot)$, $e(x)$, $e(y)$, dl 」

ここで、複数の立候補フェイズのメッセージが発行された場合、最も最初の開始メッセージの締切時刻を有効とする。

合意フェイズ 締切時刻を過ぎたら、立候補した Player は次を行う

1. 十分な長さのある合意フェイズの締切時刻 dl_2 を決定する
2. 受信したメッセージ m_0, m_1, \dots に対して (m_0, m_1, \dots は立候補メッセージ) プレイヤー ID の集合 PID を求める。
3. 次に合意用メッセージとして次を送る「プレイヤー ID, 送信時刻, x, y, dl_2 」

集計フェイズ 合意フェイズで一番早い締切時刻になったら、受信を締め切る。

1. 自分で計算した PID と受信した合意用メッセージ c_1, \dots, c_k を比較する。もし、ある合意用メッセージに含まれる $PID(c_j)$ が PID に含まれない場合 u 合は落選としてプロトコルを終了する。
2. c_1, \dots, c_k のうち、 $PID(c_j)$ が PID と等しいものを有効合意用メッセージ c'_1, \dots, c'_k とする。
3. 得られた有効合意用メッセージにおける x_j, y_j において、プロトコル 1 を利用してノードを一つ選ぶ。

しかし、このプロトコルでは合意用メッセージを送る時点で合意されるべきノードが確定していない。そのため、選ばれるはずのノードが抜けてしまった後で、抜けたノードを選ぶことに合意してしまう可能性がある。また、複数人が自分が取得したと思いついでいる場合の回避方法がない。

これを回避するため、合意用メッセージを送らなければ選ばれない一方、送る時は選ばれる前提になるようにプロトコルを改造する必要がある。

プロトコル 2 を改良したプロトコル 3 を次に示す。

[プロトコル 3]

立候補フェイズ アイテムを取得したい Player は次のことを行う。

1. ランダムな値 x, y を決定する。
2. 公開鍵 $e(\cdot)$ と他のノードが応答するのに十分な時間が開くように締切時刻 dl を決定する。
3. 立候補メッセージとして次のメッセージを半径 $2r$ 内のノードに送る「プレイヤー ID, アイテム ID, 送信時刻, $e(\cdot)$, $e^2(x)$, $e^2(y)$, dl 」

ここで、複数の立候補フェイズのメッセージが発行された場合、最も最初の開始メッセージの締切時刻を有効とする。

合意フェイズ 締切時刻を過ぎたら、立候補した Player は次を行う。

1. 十分な長さのある合意フェイズの締切時刻 dl_2 を決定する
2. 十分な長さのある確認フェイズの締切時刻 dl_3 を決定する
3. 受信したメッセージ m_0, m_1, \dots に対して (m_0, m_1, \dots は立候補メッセージ) プレイヤー ID の集合 PID を求める。
4. 次に合意用メッセージとして次を送る「プレイヤー ID, 送信時刻, $e(x), e(y), dl_2, dl_3$ 」

集計フェイズ 合意フェイズで一番早い締切時刻になったら、受信を締め切る。

1. 自分で計算した PID と受信した合意用メッセージ c_1, \dots, c_k を比較する。もし、ある合意用メッセージに含まれる $PID(c_j)$ が PID に含まれない場合は落選としてプロトコルを終了する。
2. c_1, \dots, c_k のうち、 $PID(c_j)$ が PID と等しいものを有効合意用メッセージ c'_1, \dots, c'_k とする。
3. 得られた有効合意用メッセージにおける $e(x_j), e(y_j)$ において、プロトコル1を利用してノードを一つ選ぶ

確認フェイズ ノードがひとつ選ばれたら確認フェイズに入る。

1. 選ばれたノードは次の確認メッセージを送る。「プレイヤー ID, アイテム ID, 送信時刻, $x, y, \{c'_1, \dots, c'_k\}$ 」
2. 選ばれてないノードは確認メッセージを受け取ることで合意結果を確定する。

再合意フェイズ 締切時間になっても選んだノードから確認メッセージが届かなければ再合意フェイズに入る。

1. 再合意フェイズに入ったノードは再びプロトコル1を利用する。ただしここでは、前回選んだノード除いてノードを選ぶ。
2. 選んだノードを使い再び確認フェイズに入る。

このプロトコルでは集計フェイズの後に選ばれたプレイヤー自身が自分が選ばれたと分かっていることを他のノードに知らせている。さらに、複数人が選ばれたと主張した場合も、 c'_1, \dots, c'_k を照らしあわせることで矛盾しているものを特定することができる。もし、選ばれたプレイヤーが名乗り出ないときは再び集計フェイズからやり直すことで、次の合意者を決定できる。

4 まとめと今後の課題

P2P 型の MMORPG 向けの MMORPG の仮想現実上で近距離のノードのみで合意問題を解くプロトコルを作成した。このプロトコルを使うとアイテムから半径 r 以内のノードだけでアイテムの所有者を合意することができる。今後はこのプロトコルを論理的に証明し、実際にプログラムとして実装する。

参考文献

- [BZ89] J. Bar-Ilan and Dror Zernik. Random leaders and random spanning trees. *Lecture Notes in Computer Science*, Vol. 392, pp. 1–12, 1989. Distributed Algorithms.
- [FTT10] Lu Fan, Phil Trinder, and Hamish Taylor. Design issues for peer-to-peer massively multiplayer online games. *Int. J. Adv. Media Commun.*, Vol. 4, pp. 108–125, March 2010.