

リアルタイムグループウェアにおけるイベント順序制御の一考察

徳永 博之 関野 公彦 久保田 創一 佐藤 栄

NTT 情報通信研究所

対戦型ネットワークゲームのようなリアルタイムグループウェアでは、使用する回線の違いなどに起因する計算機間の通信遅延時間の差が利用者間の有利不利を招く。通信遅延時間の大きい回線を利用する者が送信したイベントは通信遅延時間の小さい回線を利用する者が後から送信したイベントに追い越されることがあり、不利な条件での利用を余儀なくされる。我々は、このイベントの送信順序と処理順序の逆転現象が計算機間の通信遅延時間の差により生じることに着目し、イベントの送信順序と処理順序を一致させ、同一条件でのアプリケーションの利用を保証するイベント順序補正機構を試作し、有効性の検証を行った。

A Study on Event Order Inversion on Real-time Groupware

Hiroyuki Tokunaga, Kimihiko Sekino, Souichi Kubota, and Sakae Sato

NTT Information and Communication Systems Labs.

The increasing performance and decreasing cost of computers and communications have made it more common for computers to be connected via networks. This is also increasing the demand for groupware; software that connects computers on networks with each other, and facilitates collaboration among its users. Some real-time groupware, such as network games, voting systems, etc., requires fairness among users. However, fairness is not guaranteed by most existing groupware products. The difference in communication lines to which each user is connected may cause unfairness. An event sent over high-speed communication line may overtake an event on a low-speed communication line. We call this the 'event inversion problem'. The problem must be solved to support real-time groupware which demands fairness among users. We focus on the difference of communication latencies between users which may cause event inversion. We have implemented a 'real-time event sorter' which sorts the received events by their timestamps to guarantee that the receiving order matches the sending order, and have verified the validity of the proposed method.

1. リアルタイムグループウェア

1.1 リアルタイムグループウェアとは

一般にグループウェアは、時間と空間の2つの観点から分類される¹⁾が、ここでは空間関係に関わらず、即時に処理結果が全利用者に反映されるものをリアルタイムグループウェアと呼ぶことにする。

グループウェアにより利用者間で共有している情報(共有データ)には、一貫性を保障するために利用者の操作に対する排他制御が必要となる。

我々は、この排他制御をユーザレベルで意識する必要があるかどうかにより、リアルタイムグループウェアを次の2つに分類した。

明示的排他制御 (Explicit Mutual Exclusion) 議長を置く等、利用者間で操作が競合しないように、利用者間で明示的に排他制御を行うグループウェア。利用者は順番が来るまで共有データの操作は行えない。

暗黙的排他制御 (Implicit Mutual Exclusion) 利用者が操作の競合を意識せずに操作するグループウェア。利用者は他の利用者に関係なく、共有データの操作を行える。

我々は、リアルタイムグループウェアにおいて、暗黙的排他制御を行なう必要が有るアプリケーションを対象としている。このようなリアルタイムグループウェアの例としては、リアルタイム対戦型ネットワークゲームやネットワークオークション等が考えられる。

1.2 必要な機能と従来技術の問題点

1.2.1 イベント順序制御

§1.1で述べたリアルタイムグループウェアでは、利用者間で共有データ空間への操作の競合が生じることがある。

例えば、対戦型のシューティングゲームで一方のプレイヤーが他方のプレイヤーを攻撃しようとしているとすると(図1)。攻撃側は発砲を、防御側はシールドをイベントとして相互に通知し合う。このとき、攻撃側では相手に命中したと判断し、一方、防御側ではシールドで防御したと判断してしまふと、両者の判断が食い違い、状態の整合が取れなくなってしまう。

これは、両計算機でのイベントの処理順序が一致していないことに起因しており、このような状態の不一致を避けるために、リアルタイムグループウェアでは実時間に基づいてイベントの処理順序を一致させるイベント順序制御が必要となることがある。

またリアルタイムグループウェアでは、複数の利用者から、非同期にイベントが発生する。これらのイベント

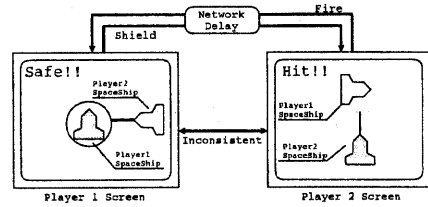


図1 利用者間のイベント同期

Fig. 1 Event Synchronization among Participants

は、各計算機の通信遅延時間の差により、送信先の計算機に到着するまでに他のイベントに追い越されて、送信順と受信順(処理順)の逆転が起こりうる。

このイベントの送信順と受信順の逆転は、通信遅延時間の大きい回線を使用する利用者小さい回線を使用する利用者間に有利不利を生じさせる原因となる。例えば、対戦型リアルタイムネットワークゲームでは、基本的に同一条件での対戦を前提としており、この意味からもイベント順序制御が必要となる。

1.2.2 イベントの応答時間に関する制約

イベントの順序逆転を検出し、補正するには小さい通信遅延時間を持つ計算機から送信されたイベントが大きい通信遅延時間を持つ計算機から送信されたイベントを待つようにすればよい。

また、通信遅延時間は常に一定ではなく、ネットワークのトラフィックや計算機の負荷等により、揺らぎが存在する。この揺らぎを含めて、完全にイベントの順序を補正するためには、揺らぎの特性を考慮し、十分な待ち時間を設定する必要がある。

さらに、リアルタイムグループウェアでは、実時間に沿って処理が行われているため、イベントの順序補正のための待ち時間は最小限に抑える必要がある。

2. モデル

2.1 用語の定義

定義1 (イベント) 共有データ空間に対する操作の要求や結果の通知を行うトランザクション。

定義2 (共有データ空間) 複数の利用者間で共有されているアプリケーションのデータ空間のこと。共有データ空間に対して行った操作は、同じデータ空間を共有する他の利用者にも反映される。

イベント同期やイベント順序逆転の補正を行うためには、ネットワークを流れるイベント間に順序を定義する必要がある。

我々は、「同一のグループウェアを実行する複数の計算機の時刻が同期している」ことを前提とし、送信時刻によりイベントを順序付け、次のように定義する。

定義 3 (イベントの順序関係) 任意の2つのイベント E_i, E_j ($i, j \in \mathcal{N}, i \neq j$) の当該計算機におけるイベントの送信時刻, または受信時刻が t_i, t_j のとき, 順序関係 \prec_{EVENT} を次のように定義する.

$$t_i < t_j \iff E_i \prec_{EVENT} E_j$$

2.2 アプリケーションモデル

我々は, オープンなネットワーク上での多人数での利用を想定していることから, アプリケーションの実行モデルとしてクライアント/サーバモデルを採用することとした. サーバとしてはアプリケーションプログラム本体, もしくは, クライアント間にイベントを中継するルーティングプログラムを用いる形態を想定した.

3. イベント順序制御

3.1 順序補正原理

定義 4 (イベント順序逆転) 2つの送信計算機から送信されたイベント E_1, E_2 が $E_1 \prec_{EVENT} E_2$ の順序関係を満たし, かつ, 受信計算機におけるイベント E_1, E_2 の順序関係が $E_2 \prec_{EVENT} E_1$ を満たすとき, イベント順序逆転という.

定義 5 (最大通信遅延時間) クライアント計算機 C_i ($i = 1, \dots, N$) とサーバ計算機 S との間の通信遅延時間を D_i としたとき, D_i の最大値 $D_{max} = \max D_i$ を最大通信遅延時間と呼ぶ.

定義 6 (通信遅延時間差) 同一サーバ計算機 S と通信を行うクライアント計算機 D_i ($i = 1, \dots, N$) について, 最大通信遅延時間 D_{max} と各クライアント計算機 D_i との通信遅延時間の差 $\Delta_i = D_{max} - D_i$ ($i = 1, \dots, N$) を通信遅延時間差と呼ぶ.

今, 通信遅延時間が D_i, D_j ($D_i > D_j$) の2台のクライアント C_i, C_j から時刻 t_{s_i}, t_{s_j} ($t_{s_i} < t_{s_j}, t_{s_j} - t_{s_i} \leq D_i - D_j$) に送信されたイベント E_{C_i}, E_{C_j} はサーバ計算機にそれぞれ時刻 $t_{R_i} = t_{s_i} + D_i, t_{R_j} = t_{s_j} + D_j$ に到着し, イベントの順序逆転が発生する. そこで, クライアントでの送信前, もしくは, サーバへの到着後において, 通信遅延時間差 $\Delta = D_i - D_j$ に相当する時間だけイベントの処理を遅らせることにより, 順序逆転を補正することができる(図2). このイベントを待たせる時間をイベント待機時間と呼ぶ.

4. 実装

4.1 RGソケットの概要

我々はイベント順序制御を行なうモジュールとして, バックレソケットインタフェースと同様なインタフェースを持つRG (Real-Time Groupware) ソケットを実装した. RGソケットは図3に示した構

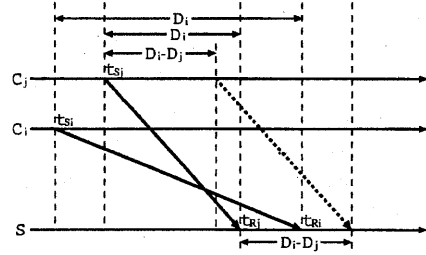


図2 イベント順序補正原理
Fig. 2 Principle of the Event Order Correction

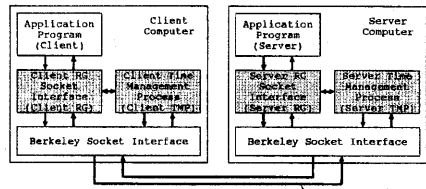


図3 RGソケットの構成
Fig. 3 Structure of the RG Socket

成を採る. 図3の網掛け部分がRGソケットの構成モジュールである.

RGソケットの各構成モジュールは次のような機能を提供する.

- クライアントRGソケット

上位APからの送信要求を受け, イベント順序補正に必要なタイムスタンプを付加する. 到着したイベントはそのまま上位APに渡す.

- サーバRGソケット

クライアントRGソケットの機能に加え, クライアントから到着したイベントをタイムスタンプ順にソートすることで順序補正機能を提供する.

- TMP (Time Management Process)

RGソケットが走行する全計算機間の時刻同期と通信遅延時間の計測を行う.

TMPは次の2つの機能を提供する.

- 時刻同期

RGソケットではタイムスタンプによるイベント順序補正を行うため, 全計算機間の時刻が同期していることが前提となる. TMPはNTP(Network Time Protocol)と同様の方式により, 時刻のずれの計測を行い, サーバ計算機と各クライアントの時刻オフセットをサーバRGソケットに提供する.

- 通信遅延時間計測

イベント待機時間を算出するためのサーバ-各クライアント間の通信遅延時間を計測する. 計測は時刻オフセットの計測時に同時に行う.

4.2 アルゴリズム

RG ソケットは次のアルゴリズムで機能する。

- (1) サーバ RG ソケットはクライアント RG ソケットの IP アドレスを TMP に登録する。
- (2) TMP は指定された IP アドレスの計算機とサーバ計算機間の時刻オフセット、通信遅延時間を計測する。
- (3) サーバ RG ソケットは TMP の通信遅延時間の計測結果から、最大通信遅延時間を検索し、各クライアントのイベント待機時間を設定する。
- (4) サーバ RG ソケットは到着したイベントのタイムスタンプに時刻オフセットを加算し、サーバ計算機におけるイベントの送信時刻を算出する。
- (5) イベントキューに到着イベントを加え、サーバ計算機での送信時刻でイベントをソートする。
- (6) 指定されたイベント待機時間が来るまで、イベントキュー内にイベントを蓄積する。
- (7) 上記、1から6を繰り返す。イベント待機時間を過ぎたイベントはアプリケーションから読みとり可能となる。

5. 実験

5.1 実験環境

RG ソケットの評価実験に用いたシステムと対象アプリケーションの構成を示す。

5.1.1 実験システムの構成

下記の実験システムを構成し、実験を行った。

- **サーバ計算機**
Sun SPARC station 20 (Sun Solaris 2.4)
- **クライアント計算機**
IBM PC/AT 互換機 (Microsoft Windows 95)
- **対象 AP**
ネットワーク対戦型ゲーム
- **ネットワーク環境**
 - Ethernet
サーバ計算機と同一セグメントの Ethernet で接続。
 - INS ネット 64 / アナログ回線
サーバ計算機と同一セグメントの Ethernet に接続された ISDN ルータに同期 64K / 28.8K モデムでダイヤルアップ IP 接続。

5.2 評価項目

RG ソケットの性能評価の観点次の2点とした。

- (1) **異なるネットワーク環境での有効性**
ISDN 回線同士等の同一の通信回線、及び、ISDN 回線とアナログ回線等の異なる通信回線の

組合せにおけるイベント順序補正能力の検証とネットワーク上を流れるイベントの発生頻度を変化させた場合のイベント順序補正能力を検証する。

- (2) **イベント待機時間の違いによるイベント順序補正能力の変化**

イベント待機時間を変化させた場合にイベント順序補正能力がどのように変化するかを検証する。

5.3 実験内容

§5.2の観点に基づいて行った実験内容を次に示す。なお、各実験とも、「イベント総数」、「最大通信遅延時間差」、「イベント順序逆転回数/率」、「イベント順序補正回数/率」を計測した。

5.3.1 実験 1

本実験項目では、通信回線の組合せによりイベント順序補正能力にどのような変化が起きるかを検証するためのものである。

実験した通信回線の組合せは表1の通り。なお、計測は、通信回線の各組合せ毎に、イベント順序補正を行った場合と行わなかった場合の2通り行った。表中の数字はクライアント計算機の台数を示す。

表1 通信回線の組合せ

Table 1 Combinations of Communication Lines

実験番号	1-1	1-2	1-3	1-4	1-5	1-6	1-7
Ethernet	2	-	-	1	1	-	1
ISDN	-	2	-	1	-	1	1
Analog	-	-	2	-	1	1	1

5.3.2 実験 2

本実験項目では、イベントの発生頻度の違いによるイベント順序補正能力の変化を調べるためのものである。

イベントの発生頻度の調整は、サーバ計算機に接続するクライアント計算機の台数を変化させることで行った。

実験したクライアント計算機の組合せは表2の通り。表中の数字はクライアント計算機の台数を示す。

表2 クライアント計算機の組合せ

Table 2 Combinations of Client Computers

	ISDN	ANALOG
実験 2-1	1	1
実験 2-2	2	2

5.3.3 実験 3

本実験項目では、イベント待機時間を変化させた場合

にイベント順序補正能力にどのような変化が起きるかを検証する。

実験には、Ethernet/ISDN/ANALOGを通信回線とするクライアント計算機を各1台、計3台使用して計測した。

実験したイベント待機時間の組合せは表3の通り。

表3 イベント待機時間の組合せ

Table 3 Combinations of Event Waiting Times

	イベント待機時間(msec)
実験 3-1	5(msec)
実験 3-2	300(msec)
実験 3-3	600(msec)
実験 3-4	$D_{max} \times 1(msec)$
実験 3-5	$D_{max} \times 2(msec)$
実験 3-6	$D_{max} \times 3(msec)$
実験 3-7	$D_{max} \times 4(msec)$
実験 3-8	$D_{max} \times 5(msec)$

5.4 計測結果

§5.3で示した実験の結果を表4～表6に示す。

なお、実験1,2についてはイベント待機時間は最大通信遅延時間差の5倍($D_{max} \times 5$)に設定した。また、実験3の結果はイベント待機時間の小さい順に並べた。遅延時間差、待機時間の単位はmsecである。

表4 実験1結果

Table 4 Result of Experiment 1

実験番号	遅延時間差 待機時間	イベント 総数	逆転回数 逆転率(%)	補正回数 補正率(%)
実験	4	639	2	0
1-1	20		0.31	0.00
実験	26	378	2	0
1-2	130		0.53	0.00
実験	79	248	46	9
1-3	395		18.55	19.57
実験	32	415	13	5
1-4	160		3.13	38.46
実験	74	266	16	2
1-5	370		6.02	12.50
実験	78	332	16	11
1-6	390		4.82	68.75
実験	75	377	24	18
1-7	375		6.37	75.00

5.5 考 察

5.5.1 イベント順序逆転率

各実験の計測結果から、実験に用いたネットワーク対戦型ゲームの平均イベント順序逆転率を算出した。

平均イベント順序逆転率の算出に使用したのは、Ethernet/ISDN/Analog各1台ずつの構成で行った実

表5 実験2結果

Table 5 Results of Experiment 2

実験番号	遅延時間差 待機時間	イベント 総数	逆転回数 逆転率(%)	補正回数 補正率(%)
実験	79	332	16	11
2-1	395		4.82	68.75
実験	82	604	130	127
2-2	410		21.52	97.69

表6 実験3結果

Table 6 Results of Experiment 3

実験番号	遅延時間差 待機時間	イベント 総数	逆転回数 逆転率(%)	補正回数 補正率(%)
実験	-	554	48	0
3-1	5		8.66	0
実験	71	762	45	12
3-4	71		5.91	26.67
実験	71	482	37	17
3-5	142		7.68	45.95
実験	71	745	81	60
3-6	213		10.87	74.07
実験	-	438	55	43
3-2	300		12.56	78.18
実験	77	620	51	39
3-7	308		8.23	76.47
実験	75	377	24	18
3-8	375		6.37	75.00
実験	-	419	44	43
3-3	600		10.50	97.73

験(1-7,3-1～3-8)の結果である。

平均すると、イベント総数5328個に対して、イベント順序逆転が457回検出されており、イベント順序逆転率は8.58%となった。

この結果から、約12回に1回の割合でイベントの順序逆転が発生していることになる。すなわち、Ethernetと比較して、通信遅延時間が大きいISDNやAnalog回線の利用者は不利な状況に置かれていると考えられ、イベント順序補正がこのようなリアルタイムグループウェアに必要であることが裏付けられた。

5.5.2 イベントの発生頻度の順序補正能力への影響

実験2の結果から、クライアント計算機の台数が2倍になったときに、イベント総数は332個から604個へ約2倍になったのに対し、イベント順序逆転の発生頻度は16回から130回へ、率も4.82%から21.52%へと約8倍もの大幅増加となった。これは、クライアント計算機の台数が2倍になり、平均イベント発生時間間隔が約半分になったことで、通信遅延時間の差の影響が大きく現われたためであると考えられる(図4)。

イベント順序補正率はクライアント計算機の2から4台の変化に対して、68.75%から97.69%へと大幅に向

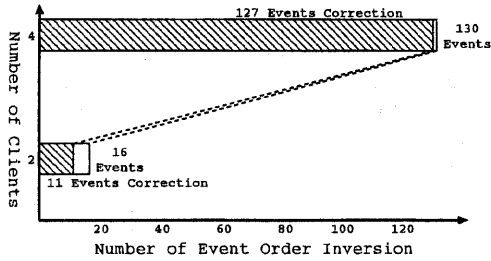


図4 イベント発生頻度と順序補正との関係(回数)

Fig. 4 Relation Between Event Rates and Correction Rates times

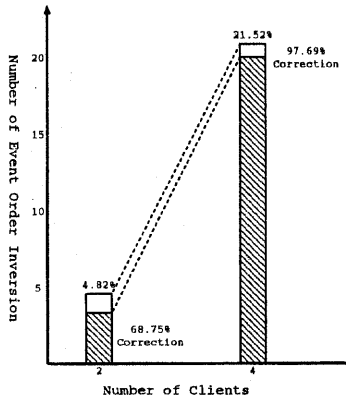


図5 イベント発生頻度と順序補正との関係(率)

Fig. 5 Relation between Event Rates and Correction Rates %

上している。これは、イベントの発生間隔が粗な状態だと、イベント待機時間内に到着するイベントの数が少なくなり、順序補正の効果が現われにくく、逆にイベント発生間隔が密な状態になると、イベント待機時間内に多くのイベントが到着し、順序補正の効果が顕著に現われたのだと考えられる(図5)。

以上より、通信遅延時間差の補完によるイベント順序補正は、ある程度以上のイベント発生頻度がないと効果が薄いと見える。逆に、イベント発生頻度が高い場合には、十分なイベント補正効果を発揮できることが分かった。

5.5.3 イベント待機時間による補正能力の変化

計測回毎にイベント順序逆転率にばらつきがあるが、順序補正率はイベント待機時間の増加に伴い、向上している(図6)。

しかし、十分な順序補正能力を確保するためには、数百ミリ秒のイベント待機時間が必要となっている。これは、アナログ回線のモデムが通信時にデータ圧縮を行うため、通信遅延時間のばらつきが大きいことが影響して

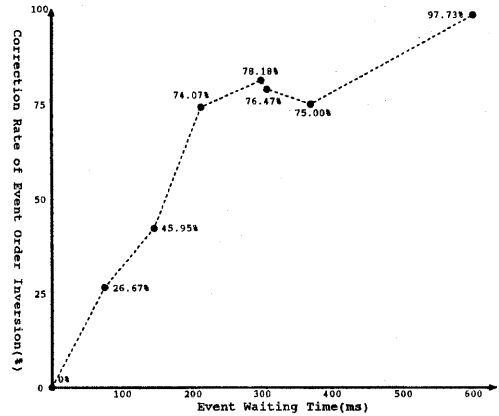


図6 イベント待機時間による補正能力の変化

Fig. 6 Relations between Correction Rates and Event Waiting Spans

いると考えられる。

6. 結 論

今回、実験したネットワーク対戦型ゲームは、イベント順序逆転の発生率が10%近く、モデム利用者が不利な状態で対戦していることが分かった。実際、キャラクタの動きも、Ethernetとモデムには大きな差があった。

RGソケットを使用し、イベントの順序補正を行うと、Ethernet接続の場合もほぼモデムと同等の速度となったが、この公平さはEthernet接続の操作性を犠牲にして確保されたものである。

今回の実験により、RGソケットはイベントの発生頻度が高い場合に有効であることが実証された。また、イベント待機時間を長くすることで、補正率を向上可能であるが、同時に応答性が低下するため、補正率と応答性のトレードオフを考慮して、イベント待機時間を設定する必要がある。

今後は、どの程度のイベント順序逆転が許容されるか、各種アプリケーションを調査し、ある程度の順序補正能力を維持しながら、応答性を確保する方法を検討する必要がある。

参 考 文 献

- 1) 石井裕: CSCWとグループウェア, オーム社(1994).
- 2) Mills, D. L.: Precision Synchronization of Computer Network Clocks, *SIGCOMM Computer Communication Review*, Vol. 24, No. 2, pp. 28-43 (1994).