分散市場モデルの移動エージェントによる実装

田中慎司[†,☆] 八槇博史[†] 石田亨[†]

インターネットに代表されるネットワーク環境において資源の効率的な割当てを行うための一手法として，計算機上の仮想的市場を用いることにより利用者の選好を基にしたネットワークQoSの制御を行う手法を提案してきた．このような市場に基づくアプローチは広く提案されており，多数の利用者の様々な要求を効率的な制御につなげる手法として有効である．そこでの重要な課題として，価格調整過程にともなう通信量の問題があげられる．本稿では，移動エージェントを利用することによって計算速度の問題を解決した，ネットワークQoS制御システム *QoS Market* について報告する．QoS Marketは，様々なユーザの各アプリケーションへの選好を反映した効率的な資源割当て目標とし，移動エージェントの適用によって，1）ユーザの個人情報の保護，2）効率的な市場計算を実現している．実環境への適用実験の結果，移動エージェントを利用した実装方式が有効であること，このシステムがイントラネット規模のネットワークにおいて実用上十分な性能を確保できていることを示した．

An Implementation with Mobile-agents for Distributed Market Computing

Shinji Tanaka,[†,☆] Hirofumi Yamaki[†] and Toru Ishida[†]

We have proposed an approach to control QoS in multimedia applications in the Internet by applying computational markets to derive efficient resource allocation based on users' private preferences. Market-based approaches have been applied where efficient resource control based on various requests from users is required. One of the critical issues in the previous works has been the communication cost in price adjustment process. In this paper, we introduce *QoS Market*, which aims to derive efficient resource allocation based on various user preference. By applying mobile agents. it successfully 1) protects users' private information, and 2) achieves fast computation of the market-based resource allocation. The result of experiment shows that mobile agent approach is effective, and that the system satisfies the performance required to be used to control networks with the size of intranets.

1. はじめに

インターネットに代表される今日のネットワーク環境では，多くのユーザがユーザがネットワーク資源を共有している．限られた資源の中でより良いサービスを実現するために，通信ごとに使用されるネットワーク資源を制御する技術の開発が行われている．実際にネットワークQoS制御については様々な研究があり，効率についても優れたものがある[4),9)]．これらがアプリケーションの多様な性質から最適な資源利用を求めるのを目的としているのに対し，本研究の目標は多数の利用者がそれぞれに複雑な選好を持つ場合に，どの利用者がどのアプリケーションにどれだけ資源を用いるのが望ましいのかというポリシーを与えることにある．その過程において，計算機上の仮想的市場を用いることにより利用者の選好を基にしてネットワークQoSの制御を行う手法を提案してきた[15)]．

この手法では，資源割当て問題を計算的市場の形でモデル化し，ユーザの選好とアプリケーションの性質を消費者エージェントと生産者エージェントとして代表させる．それぞれのエージェントは入札を価格に基づいて更新し，市場の価格調整機構によって需要と供給が均衡する．この操作の結果パレート効率的な資源割当ての配分が得られるという性質が，一般均衡理論の成果として知られている．上述のうちFulpらによるQoS制御[4)]においても，ほぼ同様の方式が採用されており，良好な結果が得られている．

† 京都大学大学院情報学研究科
Graduate School of Informatics, Kyoto University
☆ 現在，NTTネットワークサービスシステム研究所
Presently with NTT Network Service Systems Laboratories

このような一連の研究の中でつねに問題となるのは，計算に必要なメッセージ交換の増大によるパフォーマンス低下である．市場モデルによるネットワーク資源割当ての計算は，各ユーザがネットワーク上に分散しており，また価格調整のための計算において，ユーザと市場の間で大量の通信が発生する．特に，ネットワークへの要求が時々刻々と変化していく中にあっては，割当て計算の遅れは割当てそのもの品質低下へと直結する．

本稿では，市場機構による割当て方式の実装にあたって移動エージェントを用いることにより，これらの通信を軽減させる方式について述べる．移動エージェントを利用することにより，特定のアプリケーションに依存せず，ユーザのアプリケーションへの選好という個人情報を漏洩することなしに，市場モデルによる効率的な資源割当てを実用的な時間で計算するシステムを提案する．

価格調整機構に基づく計算は，一般的に，財の需給の均衡点を導くために多数の繰返しを含む．そのため，環境が与えられてから，資源割当ての計算が終了するまでに，ある程度の遅延が生じる．しかし，多数の利用者が利用しているネットワーク環境は，アプリケーションの使われ方によって，環境が動的に変化するため，この遅延を最小限にする必要が生じる．利用者が選好を変更してから，実際に資源割当てが変更されるまでのタイムラグは，均衡点計算の正確さと計算時間のトレードオフ[16]となる．この研究では，市場モデルを 3D 仮想空間上のマルチメディア会議システム FreeWalk[7]に組み込み，その画像転送を制御した．また，市場モデルによる資源割当て計算の実用的な計算時間の目安を 250 msec としており，これを達成するために，システム全体が 1 つのプロセスで実装することにより，ネットワークを介するメッセージ交換を少なくし，計算時間を短縮していた．しかし，この実装では，一般のアプリケーションに対して資源割当てを適用できず，また，利用者の選好情報が，需要に従って価格を更新する機能を持つ競売人に漏洩するなどの問題があった．

本稿では，この問題を解決するために移動エージェントを利用したネットワーク QoS 制御システム *QoS Market* について報告する．

2. 資源割当てのための市場計算

本章では，市場モデルによるネットワーク資源割当ての枠組みについて述べ，この手法を実際に適用する場合に問題となる性能について議論する．

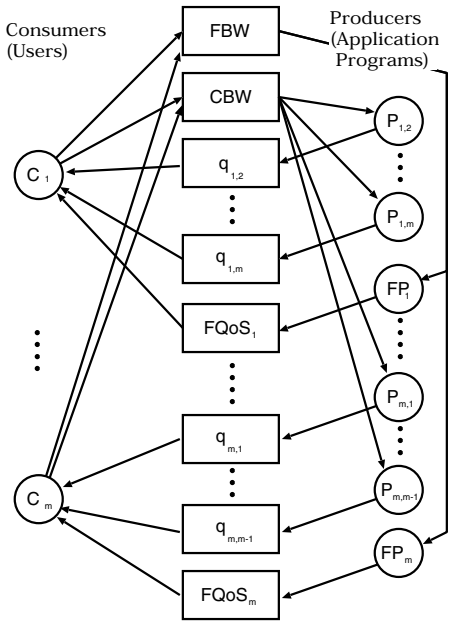

図 1 市場モデル
Fig. 1 Market model.

2.1 QoS 市場モデル

本研究で採用している個々の利用者の様々な選好に基づく効率的な資源割当て機構は，計算的市場に基づくものである．この計算的市場によるアプリケーション QoS 制御は，以下の 2 つの基本的な考え方に基づいている．

(1) 利用者はネットワーク資源の性能指標（帯域幅など）そのものではなく，アプリケーション QoS の方を評価する．アプリケーション QoS は用いた資源の量によってある程度知ることはできるが，ここでは QoS は独立なパラメータとして定義され，任意の生産関数によって消費する資源量から計算される．

(2) ネットワークに「現在」と「未来」との区別を設けることで，比較的活動的でない利用者が，ネットワーク資源を活動的な利用者群に対して譲渡する動機付けを行う．同一の価格システムの中で現在と未来について別々に財を定義することによって，利用者の要求の動的変化に自然に対処することが可能となる．

アプリケーション QoS 割当てのための市場モデルを**図 1** に示す．中央の四角形は市場中でやりとりされる財（good）を表す．財には帯域幅（bandwidth）とサービス品質（QoS）とがあり，これらの財はさらに時間軸方向で現在（current）と未来（future）の 2 つに分けられる．財としての帯域幅は各利用者が保持し

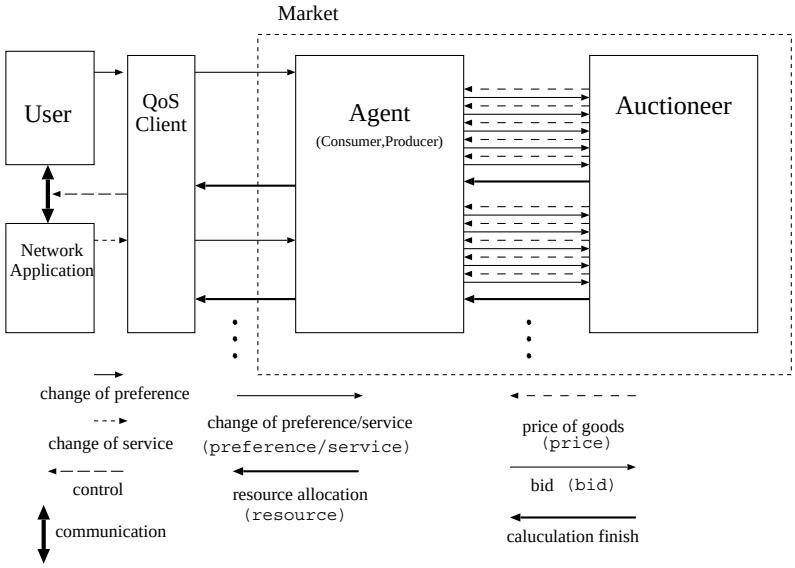

図 2 QoS Market 内の情報の流れ
Fig. 2 Information flow in QoS Market.

ている帯域の使用権を代表し，現在の帯域幅（CBW）および未来の帯域幅（FBW）とがある．利用者 j から利用者 i へ向かう通信の現在のサービス品質は q_{ij} で表され，$FQoS_i$ は利用者 i が受ける未来のサービス品質である．未来のサービス品質についてはモデルの簡単化のため，通信ごとには定義せずに各利用者 1 つずつの財としている．

図中の円はエージェントを表し，これには各アプリケーションにおける利用者の効用を代表する消費者エージェント（左列）と，使用するネットワーク資源からアプリケーション QoS へのハードウェアおよびソフトウェアによる変換操作を代表する生産者エージェント（右列）とがある．図中の矢印はモデル中での財の流れを表し，利用者が持つ帯域の使用権がアプリケーションによりサービス品質へと変換され，利用者がそれを享受することを表している．

2.2 市場計算アルゴリズム

この市場モデルには，エージェント，競売人の 2 つの要素がある．IPv6 における RSVP [2] のようなネットワーク資源制御機構を前提とした場合の，これら要素間の情報の流れを**図 2** に示す．

消費者エージェントは，利用者と 1 対 1 に対応し，利用者の選好を代表する．利用者の選好は，消費する財の量を引数とし，より高い値が好ましい効用値を返す効用関数として定義される．利用者が自らの選好に関する情報を消費者エージェントに伝える（`preference` メッセージ）．消費者エージェントは，効用関数を最大化するように，財への入札を変化させる．

生産者エージェントは，アプリケーションのサービスと生産者エージェントは 1 対 1 に対応し，各アプリケーションのサービスの特性を代表する．したがって，アプリケーションと生産者エージェントは 1 対 1 対応をしない場合もある．アプリケーションの特性は，消費するネットワーク資源を引数とし，出力するアプリケーション QoS を返す生産関数として定義される．新しいアプリケーションが立ち上がる，または終了する場合，利用者は新たに生産者エージェントを生成，または消去する（`service` メッセージ）．

競売人は，各財と 1 対 1 に対応する．競売人は，エージェントから送られてきた情報を基に，需要と供給が均衡するように価格を調整する．そして，調整された価格は，エージェントに送られ，ふたたび入札が行われる．

実際の市場計算は以下に示すアルゴリズムで行われる（**図 3**）．

(1) 競売人はエージェントに各財の価格を伝える（`price` メッセージ）（3 行目）．ループの初回では，利用者の財の初期保有量も同時に伝えられる．各利用者の財の初期保有量は，前回の市場計算結果における未来の財の保有量を反映する．
(2) エージェントは，与えられた価格情報を基に，利用者の効用を最大化する財の需給量を計算し（4 行目），それを市場に伝える（`bid` メッセージ）．
(3) 競売人は，各財の需給量を合計し，需給が均衡

するように財の価格を調整する（6〜13 行目）．

(4) 需要と供給の差が利用者が決定する閾値より小さい場合は，計算を終了する．このときの価格が正確にネットワーク資源の割当てを反映している（15 行目）．それ以外の場合 (1) に戻り，計算を繰り返す．

(5) 市場計算が終了すると，生産者エージェントは最終価格からネットワーク資源の割当てを計算し，対応するアプリケーションプログラムの通信を制御するために QoS Client に通知する（`resource` メッセージ）．

市場計算は，環境変化のたびに繰り返される．そのため，市場計算のオーバヘッドが大きいと，環境変化への追従が遅れ，さらに，市場計算自体に資源が占有され，効率的な資源割当てが実現されるとはいいがたい．

市場から利用者への市場計算の計算結果の `resource` メッセージは，市場計算の打ち切り時に発生する．また，利用者が消費者エージェントに選好の変化を通知する `preference` メッセージや利用者が生産者エージェントに状況の変化を通知する `service` メッセージは，利用者の入力に反応して発生する．そのため，通信頻度・通信量ともにそれ以外のメッセージに対して無視できる程度に小さい．他方，市場がエージェントに各財の価格を通知するための `price` メッセージと，エージェントが競売人に各財の需給量を通知するための `bid` メッセージは，価格の調整のたびに発生し，全体で発生する通信の大部分を占めている．

以下，本稿ではこれらの通信メッセージの削減による性能の向上に関して論じる．

```
 1: repeat
 2:   begin
 3:     Actioneer reports price to agents;
 4:     Agent calculates supply or demand;
 5:     Auctioneer aggregates supply and demand;
 6:     if(supply over demand) then
 7:       begin
 8:         Auctioneer raises price;
 9:       end
10:     if(supply under demand) then
11:       begin
12:         Auctioneer lowers price;
13:       end
14:   end
15: until(| supply - demand |> threshold);
16:
17: Agent calculates its share of bandwidth;
18: Agent reports it to QoS Client;
```

図 3 市場計算のアルゴリズム
Fig. 3 Algorithm of market calculation.

3. 実装方式の比較

本章では，移動エージェントを利用した実装について議論を行う．

3.1 移動エージェント方式

本研究における市場モデルでは，エージェントは利用者のネットワークアプリケーションの選好や使い方から導き出される入札を競売人に送る．利用者は，ネットワーク上に分散しているため，エージェントの実装には，分散実装と集中実装という 2 つの典型的な方針がある．

静的エージェントによる分散実装（**図 4** (a)，図中のメッセージは図 2 と同じ）は，エージェントを利用者のローカルにおく．この場合，エージェントの計算は並列に行うことができるが，前章で述べたように `price` メッセージと `bid` メッセージは，ネットワークを介した通信となり，市場計算において，大きなオーバヘッドの原因となる．1 回の市場計算に必要な時間は，次式のようになる．

$$T_d = n_r(T^d_{price} + T_{agent} + T^d_{bid} + T_{auctioneer}) + T_{result} \quad (1)$$

ここで，クライアントの数を n_c，需給均衡に達するまでの計算の繰返し回数を n_r とし，財の価格情報

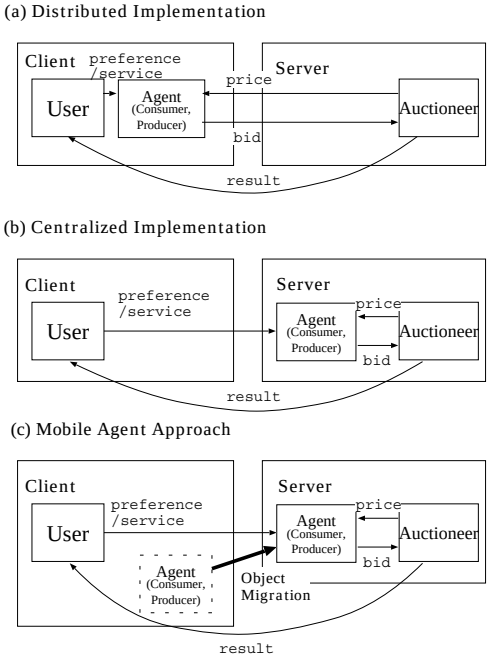

図 4 実装方式
Fig. 4 Implementation pattern.

の通信時間を T^d_{price}，エージェントの財の需給量の計算時間を T_{agent}，財の需給量の通信時間を T^d_{bid}，競売人の価格調整時間を $T_{auctioneer}$，計算結果を利用者に通知する時間を T_{result}，1回の市場計算の時間を T_d とする．

それに対し，集中実装（図4(b)）では，エージェントを競売人の位置するホスト（以下，市場サーバとする）に実装する．この場合，エージェントの計算は逐次行う必要があるため，分散実装より長くかかるが，通信コストは最小化される．そのため，1回の市場計算に必要な時間が次式のようになる．

$$T_c = n_r(T^c_{price} + n_c T_{agent} + T^c_{bid} + T_{auctioneer}) + T_{result} \quad (2)$$

ここで，財の価格情報の通信時間を T^c_{price}，財の需給量の通信時間を T^c_{bid}，1回の市場計算の時間を T_c とする．

分散実装を行った場合と，集中実装を行った場合では，1回の市場計算に必要な時間の差は，以下のようになる．

$$T_d - T_c = n_r(T^d_{price} + T^d_{bid} - T^c_{price} - T^c_{bid} - (n_c - 1)T_{agent}) \quad (3)$$

`bid` メッセージと `price` メッセージの通信時間と，エージェントの効用最大化問題に基づく入札の計算時間の差によって，どちらの実装方式が適しているかが決定される．現在のネットワーク環境では，前者が後者を上回っていることが明らかであり，分散実装より集中実装のほうが適している．

集中実装では，効用関数と生産関数が競売人と同じプロセスに静的に実装される．この方式は，比較的単純であるが，効用関数および生産関数として利用可能な関数が制限される．また，利用者の個人情報である効用関数を競売人が知ることができるため，個人情報の保護の点で好ましくない．

本稿では，両者の長所を生かすための方法として，移動エージェントによる実装方式（図4(c)）を採用した．移動エージェントは，効用関数と生産関数を持ち，市場サーバに移動する．利用者は，選好情報をエージェントに送る．

この方式では，柔軟性と個人情報の保護の両方を達成している．移動エージェントのセキュリティに関する問題はすでに先行研究があり，広範な議論がなされている[13]．

移動エージェントは，実行途中に，コンテクストを維持しながらネットワーク上の任意の場所に移動し，実行を再開することができるプロセスである．移動エージェントは，移動先のホストやほかの移動エージェントと通信できる．いくつかの環境では，サーバが悪意のあるエージェントから自らを保護するために，エージェントを認証し，また，エージェントもサーバを認証する，双方向の認証機構を提供する．

これらの機能は，市場モデルによる資源割当て機構が，利用者の選好情報が保護されなければならないというセキュリティの問題を解決し，市場計算の効率性と柔軟性を達成することを可能にする．

4. 移動エージェント方式の実装

これまでに示した均衡計算の問題を解決するために移動エージェント方式を用いた，QoS Market というシステムを実装した．本章では，QoS Market について説明する．

以下，QoS Market の構成と動作の概要について説明する．

4.1 QoS Market

システム構成図とメッセージの流れを**図5**に示す．図中のメッセージは図2と同じである．システムは，大きく QoS Client，移動エージェント，競売人の3つの要素に分けられる．矢印は，これらの間のエージェントの移動と，入札や価格，資源配分といった情報の流れを示す．すべての要素は，Linux 上の CORBA（Common Object Request Broker Architecture）[12] のオブジェクトとして実装されている．CORBA では，LAN や Internet などのネットワーク上に分散されているオブジェクト間の通信をサポートしている．

(1) QoS Client
QoS Client は各利用者のホストに位置し，利用者とのインタフェースを提供する．
はじめに，QoS Client は移動エージェントを市場サーバに送り，QoS Market にログインする．その後，QoS Client は，利用者の選好とネットワークアプリケーションの使われ方を移動エージェントに通知する．さらに，QoS Client は，市場計算の計算結果を受け取り，ホスト上の各プロセスによる通信を制御する．

(2) 移動エージェント
利用者の選好を代表する消費者エージェントと，各アプリケーションを代表する生産者エージェントを内部に持ち，現在の価格情報を基に入札を実行することで，市場計算に参加する．
利用者が QoS Client を起動したときに，1回だけ QoS Client によって市場サーバに送り込まれる．効用関数の更新は，QoS Client から

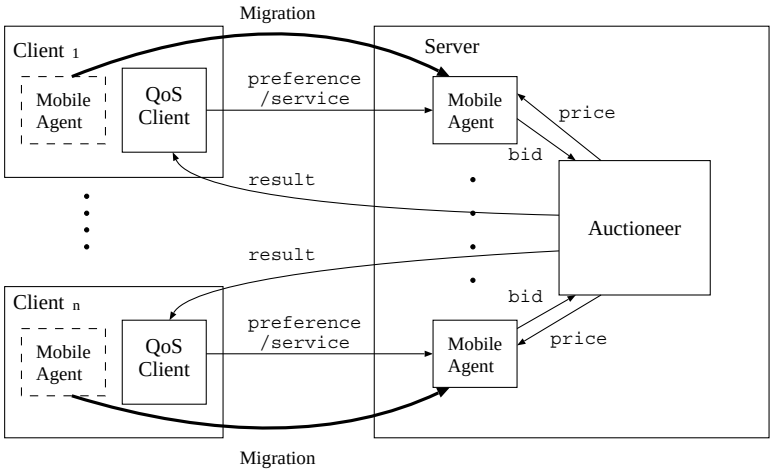

図 5 システム構成
Fig. 5 System configuration.

関数のパラメータを受け取ることによって行われる．移動エージェントは，競売人から求められたときに財の需給量を計算し，競売人に通知する．

(3) 競売人
競売人は，市場サーバに位置し，財の価格の調整を行う．現在のシステムでは，競売人は 2 つのスレッドで実装されている．1 つはすべての財を操作し，QoS Client から送り込まれてくる移動エージェントともに，市場計算を実行する．市場計算の終了し，資源割当てが決定したときは，QoS Client にその割当てを通知する．もう 1 つのスレッドは QoS Client からの要求を受け取る．

4.2 移動エージェントの実装

移動エージェントのシステムは，Aglets，Odyssey，Voyager [5),8)]など，様々なものが開発されているが，多くは Java や Tcl などをもととしており，計算のオーバヘッドが致命的となるような応用には適していない．本研究では，CORBA の C++による実装の 1 つである omniORB2 [1)]を利用した．移動エージェントは CORBA のオブジェクトとして実装され，QoS Client から，市場サーバにアップロードされ，実行される．QoS Market の移動エージェントに求められる機能は，利用者のログイン時のサーバへのアップロード機能および，市場計算時のメッセージ交換機能である．現時点で利用可能なシステムでは前者がサポートされていないため，独自に実装することとした．

はじめに，移動エージェントは実行可能ファイルとしてそれぞれのクライアントに存在する．利用者が QoS Client を起動し，QoS Market にログインすると，QoS Client は移動エージェントをエンコードして市場サーバに送る．市場サーバでは送られてきた移動エージェントをデコードし実行する．移動エージェントは QoS Client と通信し，利用者の選好情報などを受け取り，生産者エージェントと消費者エージェントを生成，市場計算に参加する．

CORBA のオブジェクトはコンテクストを持たずに移動するため，今回の実装は必ずしも本来の意味での移動エージェントとはいえず，また情報の隠蔽についても完全ではない．しかし，この実装は少なくとも移動エージェント方式の評価には十分な機能を持っていると考えている．将来的に，より高速に動作する安全な移動エージェントプラットフォームが実現されれば QoS Market の移動エージェントをそのシステムと置き換えることを検討している．

5. 移動エージェント方式の性能評価

移動エージェント方式が効率的なネットワーク資源割当てにおいて有効であることを実証するために，QoS Market が市場計算と QoS Client への計算結果の通信時間を計測した．

5.1 評価方法

評価環境は 3 台の PC で構成され，これらは 100 Mbps Ethernet で接続されている．それぞれの PC の CPU は PentiumII 450 Mhz，OS は Linux を採用している．また，CORBA は omniORB2 [1)]を採用した．それぞれのホストで QoS Client が実行し，そのうちの 1 つのホストで市場サーバおよび ORB を実行した．移動エージェントは，利用者がログインし

表 1 市場計算の各段階における所要時間
Table 1 Time consumed in each stage of market calculation.

	移動エージェント方式（msec）	分散実装（msec）
移動エージェントの移動時間	3300	0
価格計算	3.9	4.6
価格通知	1.4	1.4
需給量取得	2.8	3.9
価格調整	0.2	0.2
結果通知	5.6	5.6
割当て計算時間	18.4	24.3

たときに，はじめて市場サーバに移動する．移動エージェントのサイズは 612 KByte である．

利用者は，QoS Market に同時にログインし，それぞれの利用者は 30 秒間，ランダムに通信を発生，終了させ，また，アプリケーションへの選好を変化させた．このときの市場計算の各段階における計算，または通信時間を計測し，平均値をとった．

5.2 評価結果

表 1 に市場計算の各部の所要時間を示す．移動エージェントの移動時間は，QoS Client が市場サーバに移動エージェントを送り，それが起動されるまでの時間を測定したものである．価格計算は市場計算において 1 回価格更新を行う時間であり，財とエージェントの数に依存する．価格通知に要する時間は市場サーバが移動エージェントに現在の各財の価格を通知する時間を測定したもので，式 (1) から (3) における T_{price} に相当する．需給量取得に要する時間は，市場サーバがそれぞれの移動エージェントの各財の需給量を取得する時間を測定したもので，$T_{agent} + T_{bid}$ に相当する．価格調整に要する時間は，各財の需給量を基に価格更新を行う時間を測定したもので，$T_{auctioneer}$ に相当する．結果通知については，市場計算の結果を QoS Client に通知する時間を測定したもので，T_{result} に相当する．最後に，割当て計算時間としては，初期財の配分から割当て結果が得られるまでの時間を測定した．

移動エージェントによる実装と分散実装の差は，移動エージェントの移動と需給量取得に要した時間に現れている．分散実装の場合，需給量の取得時にネットワーク経由の通信が必要となるため，所要時間が約 40%増加している．これにより市場計算全体では，所要時間が約 30%増加している．移動エージェントの移動は毎回ではなく，利用者が QoS Market に最初にログインしたときと，移動エージェントでは対応できない効用関数を利用するときのみであり，それ以外には移動エージェントを新たに送る必要はない．そのため，移動エージェントの移動の必要な時間は，システム全体での挙動に対して影響はほとんどないといえる．需給量取得は，3.1 節で論じたように，クライアント数の増加にともない，分散実装と，集中実装と同時間で計算可能な移動エージェント方式との差が増大する．

移動エージェント方式では，市場計算 1 回に要する時間は，18.4 msec となっており，十分実用的な負荷でネットワーク QoS の配分に成功しているといえる．

6. おわりに

本稿では，計算的市場に基づくネットワーク資源割当てシステム QoS Market に移動エージェントを適用する手法について述べた．

計算的市場に基づくアプローチにおいては，入札と価格調整の繰返しがアルゴリズムの中心となるが，分散環境において実現するには，そのための通信コストの削減が大きな課題となってきた．

QoS Market システムでは，一群のエージェントを動的に適切な位置に移動する移動エージェントとして実装している．これにより市場計算時に大きなオーバヘッドとなる，競売人とエージェント間の通信コストを，エージェント群を市場サーバに移動させることにより最小化する．また，エージェント群は移動エージェントとして実装されるため，利用者の個人情報をその内部に隠蔽することが可能となる．

このシステムを Linux 上で実装し，性能評価を行い，実環境における実験結果から，移動エージェント方式による実装では，約 20 msec で市場計算を完遂することができており，要求が動的に変化していく中でのネットワーク資源制御として実用的な性能を示すことができた．

参考文献

1) AT&T Laboratories Cambridge: omniORB2 (1999). http://www.uk.research.att.com/omniORB/omniORB.html
2) Braden, B., Zhang, L., Berson, B., Herzog, S. and Jamin, S.: Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification, Internet Request for Comments, RFC2205 (1997).
3) Clearwater, S.H. (Ed.): *Market-Based Control: A Paradigm for Distributed Resource Allocation*, World Scientific (1996).
4) Fulp, E.W., Ott, M., Reininger, D. and Reeves, D.S.: Paying for QoS: An Optimal Distributed Algorithm for Pricing Network Re-

sources, *Proc. IWQoS 1998*, pp.75–84 (1998).
5) Lange, D.B. and Chang, D.T.: IBM Aglets Workbench – Programming Mobile Agents in Java, *IBM Corporation White Paper* (1996).
6) Nahrstedt, K. and Smith, J.M.: The QoS Broker, *IEEE MultiMedia*, Vol.2, No.1, pp.53–67 (1995).
7) Nakanishi, H., Yoshida, C., Nishimura, T. and Ishida, T.: FreeWalk: Supporting Casual Meeting in a Network, *Proc. 5th CSCW*, pp.308–314 (1996).
8) Object space, Inc.: ObjectSpace Voyager Technical Overview (1997). `http://www.objectspace.com/products/voyager`
9) Shahar, I.B., Orda, A. and Shimkin, N.: Best-Effort Resource Sharing by Users with QoS, *Proc. INFOCOM 1999*, Vol.2, pp.883–890 (1999).
10) Shoven, J.B. and Whalley, J.: *Applying General Equilibrium*, Cambridge University Press (1992).
11) Tanaka, S., Yamaki, H. and Ishida, T.: MobileAgents for Distributed Market Computing, *Proc. 28th ICPP-99*, pp.472–479 (1999).
12) The Object Management Group: The Object Management Group WWW Page (2000). `http://www.omg.org/`
13) Vigna, G. (Ed.): *Mobile Agents and Security*, LNCS, Vol.1419, Springer-Verlag (1998).
14) Wellman, M.P.: A Market-Oriented Programming Environment and Its Application to Distributed Multicommodity Flow Problems, *J. Artificial Intelligence Research*, Vol.1, pp.1–22 (1993).
15) 八槇博史，マイケル P. ウェルマン，石田 亨：市場モデルに基づくアプリケーション QoS の制御，電子情報通信学会論文誌，Vol.J81-D-I, No.5, pp.540–547 (1998).
16) 八槇博史，山内 裕，石田 亨：市場モデルによるアプリケーション QoS の制御：実装上のトレードオフ，情報処理学会論文誌，Vol.40, No.1, pp.142–149 (1999).

(平成 12 年 5 月 22 日受付)
(平成 12 年 12 月 1 日採録)

田中 慎司

昭和 49 年生．1998 年京都大学工学部情報工学科卒業．2000 年同大学院修士課程修了．同年日本電信電話（株）入社．分散オブジェクトモデルの次世代ネットワークへの適用に関する研究に従事．

八槇 博史（正会員）

1995 年京都大学工学部情報工学科卒業．1996 年同大学院修士課程修了．1999 年同大学院博士後期課程修了．現在，京都大学大学院情報学研究科社会情報学専攻助手，博士（情報学）．マルチエージェント，市場モデル，モバイルコンピューティング等社会情報システムに関する研究に従事．電子情報通信学会，人工知能学会，ACM, IEEE 各会員．

石田 亨（正会員）

1976 年京都大学工学部情報工学科卒業．1978 年同大学院修士課程修了．同年日本電信電話公社電気通信研究所入所．米国コロンビア大学計算機科学科客員研究員，ミュンヘン工科大学客員教授等．現在，京都大学大学院情報学研究科社会情報学専攻教授．工学博士．人工知能，コミュニケーション，社会情報システムに興味を持つ．IEEE PAMI, Autonomous Agents and Multiagent Systems 等数誌の編集に従事．編著書に，「分散人工知能」（コロナ社），「Real-time Search for Learning Autonomous Agents」（Kluwer Academic Publishers），「Community Computing: Collaboration over Global Information Networks」（John Wiley and Sons）等．