

ボランティアクラウドにおける市場指向のQoS割当て

村上 陽平^{1,a)} 石田 亨^{1,b)} 宮田 直輝^{1,†1}

受付日 2013年5月14日, 採録日 2013年10月9日

概要: ボランティアクラウドの活性化には, ボランティアによるサービス提供のインセンティブが重要である. ボランティアで提供される無償サービスは, 提供者に利潤がないため, 利用者に対する選好に基づく効用によってのみサービス提供が動機付けられる. 一方, 利用者はコストの制約がないため, 必要以上のサービスのQoSを確保する傾向にあり, その結果, 利用されないサービスが増加し, 提供者の効用を減少させ, サービス提供のインセンティブの低下を招く可能性がある. そこで本研究では, ボランティアクラウド上でサービス提供者の効用が維持されるように, システムの資源とそれを用いて提供されるサービスのQoSを財として扱う, 市場モデルに基づくQoS割当て手法を提案する. さらに, シミュレーションによって, 提案手法が利用者に対して必要以上のサービスのQoSを確保しないように動機付けを行い, サービス提供のインセンティブが維持されることを確認した.

キーワード: サービスコンピューティング, QoS制御, ボランティアサービス, 市場モデル

Market-oriented QoS Allocation for Volunteer Cloud

YOHEI MURAKAMI^{1,a)} TORU ISHIDA^{1,b)} NAOKI MIYATA^{1,†1}

Received: May 14, 2013, Accepted: October 9, 2013

Abstract: Incentives to volunteer to provide services are significant to activate volunteer cloud. Those volunteer services are motivated by preferences of the service providers because they cannot earn profits from users. While, in using voluntary services, users tend to demand higher QoS (e.g., throughput of the services) than they actually need because there is no cost. As a result, many unused services decrease providers' utilities, which loses providers' incentive to sustain their services. Therefore, to maintain providers' utilities appropriately, we have proposed market-oriented QoS allocation mechanism where users and providers exchange system resources and QoS to improve their utilities. In order to verify whether it motivates users not to demand higher QoS than they need, we ran a series of simulations of our proposed mechanism. We successfully have checked the mechanism maintained providers' incentives.

Keywords: services computing, QoS control, voluntary services, market-oriented model

1. はじめに

コミュニティのメンバによるボランティアでの資源やサービス提供はイノベーションを生み出している [11]. ボランティアコンピューティングではメンバがデスクトップ

PCをボランティアで提供することで気象モデリングや創薬などの大規模計算が行われている [1]. さらに仮想化技術と統合することでボランティアクラウドへと発展させ, クラウド上のサービスはCPUだけでなくディスクもオンデマンドで利用できる環境を構築しつつある [6]. このような計算資源の提供だけでなく, 言語グリッド [4] では, メンバによってボランティアで提供された言語資源 (機械翻訳, 辞書, 構文解析ツールなど) を Web サービスとして共有し, これらの言語サービスを非営利目的に限定して無償で提供している. 本稿では, このように提供者がボランティアで資源を提供し, 利用者が無償で利用できるサービ

¹ 京都大学大学院情報学研究科社会情報学専攻
Department of Social Informatics, Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan

^{†1} 現在, NTT コミュニケーションズ
Presently with NTT Communications, Japan

a) yohei@i.kyoto-u.ac.jp

b) ishida@i.kyoto-u.ac.jp

スをボランティアサービスと定義し、このようなサービスを提供するシステム全体をボランティアクラウドと呼ぶ。ボランティアクラウドも通常のクラウドと同様にサービスとして提供される資源の種類に応じて多様な形態 (IaaS, PaaS, SaaS) がありうるが、本研究ではソフトウェア資源のサービス提供 (SaaS) に焦点を当てる。

ボランティアサービスの目的は、所属しているコミュニティに貢献することである。たとえば外国人の観光を手助けする NPO は、提供した観光用の用例対訳が外国人の観光の助けとなることを期待している。このようなボランティアサービスの提供者の目的を達成するには、利用者の選好だけでなく、提供者の利用者に関する選好も考慮して利用者に QoS を割り当てる必要がある。

しかしながら、ボランティアクラウドの場合、サービスの利用が無料のため、利用者は必要以上の需要 (疑似需要) を入力する可能性がある。疑似需要が入力されると、そのサービスを利用しようとする他の利用者の利用機会を減少させるだけでなく、サービスが利用されないため、そのサービス提供者の効用も低下させる。

そこで、本研究では、ボランティアクラウドにおける適切な QoS 制御を実現するために以下の課題に取り組む。

- ボランティアクラウドの QoS 割当てのモデル化
ボランティアサービスの提供者の効用は、利潤を最大化することではなく、できる限り選好の高い利用者に自分のサービスの QoS が多く消費されることである。そのため提供者の選好を考慮した QoS 割当てのモデル化が必要である。
- 疑似需要の抑制メカニズム
提供者、利用者双方の効用の低下を引き起こす疑似需要が発生しないように、利用者に真の需要を示すよう動機付けるメカニズムが必要である。

2. QoS 割当て

多数の資源提供者と利用者が存在するシステムにおいて、有限な資源を有効に活用する QoS 割当てを決定する手法が提案されている。Web サービスの分野では、Zeng ら [10] が QoS の視点から Web サービス合成の問題を定式化し、線形問題を用いて適切にサービスを選択する AgFlow を提案している。AgFlow は合成 Web サービス全体の品質を評価するためのサービス品質モデルを持つ。そして Web サービス合成のために、各タスクに基づいてサービスを選択する手法と、線形問題を用いてタスク全体を考慮してサービスを選択する手法を提案している。Menascé ら [7] は、サービス指向アーキテクチャにおいて、利用者の効用に基づいて QoS の割当てを行うシステム構成を提案している。利用者は効用関数とコスト制約、提供者はコスト関数を QoS プローカに登録する。利用者の要求に対して QoS プローカが選択可能な様々なサービスの QoS を分析的待

ち行列を用いて予測し割当てを決定する。

Buyya ら [2] は一般的なグリッドシステムへの市場モデルの導入の手法について提案している。グリッドシステムには様々な利用者と提供者が存在し、それぞれが独自の目的と戦略、需要供給パターンを持っている。提案手法では競争的市場モデルを導入することで利用者と提供者の効用を高める QoS 割当てを実現する。グリッドコンピューティングの分野では、そのほか、グリッドシステムの QoS 割当てに関する研究が行われている [3], [8]。我々が想定するボランティアクラウドではサービスの利用が無料である。そのため、単純に利用者の需要に基づいた QoS 割当てでは、利用者が疑似需要を記述して適切な QoS 割当てが妨げられる可能性がある。そこで、本研究では無料でサービスが利用できるボランティアクラウドにおいて適切な QoS 割当てを実現する手法を提案する。

3. ボランティアクラウドにおける QoS 割当て

3.1 ステークホルダ

図 1 のボランティアクラウドの概要が示すように、ボランティアクラウドにはクラウド管理者とサービス提供者、サービス利用者の 3 種類のステークホルダが存在する。クラウド管理者は、サービス提供者が自分のソフトウェア資源でボランティアサービスを提供できるようにシステム資源を供与しクラウド基盤を運用する。サービス提供者は、クラウド基盤のシステム資源上に自分のソフトウェア資源を配備しボランティアでサービスを提供する。提供者は利用者間に選好を持ち、より選好の高い利用者によって自分のサービスの QoS が消費されたいという効用を持つ。なお、各ソフトウェア資源がサービスとしてサービス利用者に提供する処理能力のことを本研究では QoS と呼ぶ。つまり、利用者からの要求に対して一定時間にどの程度のタスクを処理できるかを表している。たとえば翻訳サービスの場合、翻訳ソフトウェアが一定時間に翻訳できる文の数などに対応する。一方、利用者は複数の目的を保持し、クラウド上で共有されたサービスの中からそれぞれの目的を満たすサービスを選択し利用する。サービスには同じ機能を備えたものが複数あり、利用者はそれらの間の選好に基づいてサービスを選択する。利用者はできる限り自分の好みのサービスを用いて、できる限り優先度の高い自分の目

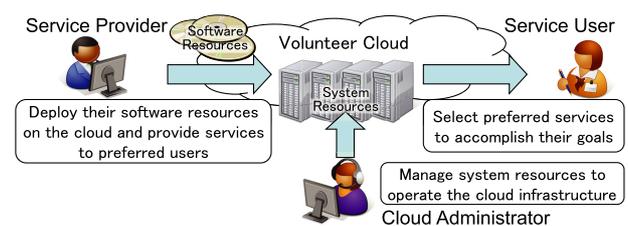


図 1 ボランティアクラウドの概要

Fig. 1 Stakeholders of Volunteer Cloud.

的をできる限り多く達成したいという効用を持つ。このようにボランティアクラウドは、クラウド管理者によって運用されるシステム資源と、サービス提供者によってボランティアで提供されるソフトウェア資源によってサービスを利用者に提供する。なお、クラウド上のシステム資源は有限であり共有されているため、あるサービスに多くのシステム資源を消費すると、他のサービスの QoS に影響を及ぼす。

3.2 QoS 割当てのモデル化

ボランティアクラウドにおける QoS 割当てメカニズムを構築するために、各サービスの提供可能な QoS や各ステークホルダの効用の定式化を行う。まず、クラウド上で共有されるサービスの集合を S とし、各サービス s ($s \in S, |S| = m$) がシステム資源 x を用いて提供できる QoS (q_s) は次式で求められるものとする。システム資源と QoS との関係には各ソフトウェアが持つテクノロジーによって様々な定義があり、規模による収穫逓減（入力の 1 単位量の増加に対する出力の上昇分が入力レベルの向上とともに減少する）の条件を満たす限り任意となるため、以下の議論では QoS の算出に次式を用いる。

$$q_s = Q_s - \frac{Q_s}{1 + \gamma_s x} \quad (1)$$

ここで Q_s は、サービス s が提供できる QoS の最大値であり、 γ_s はシステム資源の追加による生産量の増加度合いを表すパラメータである。

システム資源を多く投入するにつれて q_s は Q_s に収束する。また、利用者の集合を U とすると、 q_s は提供者が各利用者 u ($u \in U, |U| = n$) に割り当てたサービス s の QoS (q_s^u) の総和 ($\sum_{u \in U} q_s^u$) である。さらに利用者 u がサービス s で消費した QoS を y_u^s とすると、 $y_u^s \leq q_s^u$ が成り立つ。

次に利用者 と提供者の効用を定義する。利用者 u の効用 $utility_u$ は、できる限り自分の好みのサービスを用いて、できる限り優先度の高い自分の目的をできる限り満たすことである。利用者 u が消費したすべてのサービスの QoS を用いて次式で定まるとする。

$$utility_u = \sum_{d \in D_u} w_d \frac{\sum_{s \in S_d} v_u^s y_u^s}{r_d} \quad (2)$$

$$s.t. \quad y_u^s \leq q_s^u, \sum_{s \in S_d} y_u^s \leq r_d$$

ここで D_u は利用者 u の目的 d の集合であり、 w_d は目的 d に利用者が設定した重みである。また、 S_d は目的 d を満たすために利用したサービスの集合であり、 v_u^s ($0 \leq v_u^s \leq 1$) は各サービスに関して利用者が持つ選好の値、 r_d は利用者が目的 d を達成するのに必要な QoS である。

一方、提供者 s の効用 $utility_s$ は、利潤の最大化ではなく、できる限り選好の高い利用者 に自分のサービスの QoS

が多く消費されることである。各利用者によって消費されたサービス s の QoS を用いて次式で定まるとする。ただし、提供者が利用者 に設定する選好の値を v_s^u ($0 \leq v_s^u \leq 1$) とする。

$$utility_s = \sum_{u \in U} v_s^u y_u^s \quad s.t. \quad \sum_{u \in U} y_u^s \leq q_s \quad (3)$$

利用者 と提供者の目的は、各時刻の効用の総和を最大にすることである。ただし、ボランティアサービスは無償のため予算制約がなく疑似需要を引き起こす可能性がある。そこで、QoS 割当てのメカニズムは、疑似需要よりも真の需要を示す利用者の効用を高めることで、利用者 に真の需要を示すよう動機付ける必要がある。

4. 市場指向の QoS 割当て

ボランティアクラウドにおける QoS 割当てのモデルに対して、市場モデルを用いた QoS 割当てメカニズムを提案する。本研究では、市場モデルを用いて QoS 割当てを実現するために、Yamaki ら [9] が提案する消費者・生産者モデルと現在・未来モデルを用いた。

消費者・生産者モデルでは、有限なシステム資源を利用者に割り当てる。利用者は需要に応じて提供者に対するシステム資源の割当てを決定する。システム資源を割り当てられた提供者は提供可能な QoS を求め、利用者に対する QoS の割当てを決定する。

現在・未来モデルでは、市場で割り当てる財を現在のものと一定期間の未来のものとで区別する。利用者は自らの需要の変化に応じて、利用者間で現在と未来の財をやりとりする。このモデルによって、システム資源の効率的な活用を促進することができる。

4.1 消費者・生産者モデル

図 2 にボランティアクラウドの QoS 割当ての市場における消費者・生産者モデルを示す。中央の四角形は市場でやりとりされる財を表す。財にはシステム資源と QoS があり、これらの財はさらに時間軸方向で現在と未来の 2 つ

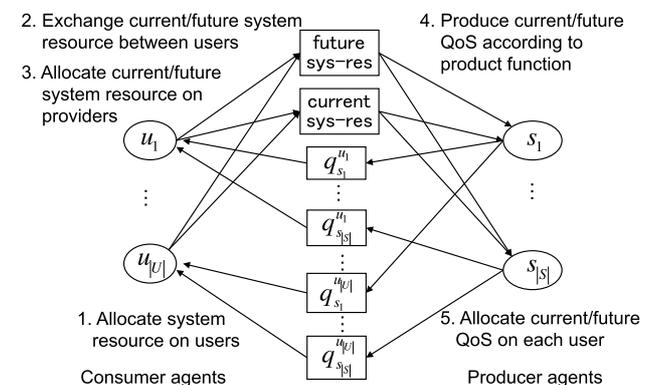


図 2 消費者・生産者モデル

Fig. 2 Consumer-producer model.

に分けられる。財としてのシステム資源はクラウド管理者によって利用者に割り当てられ、現在のシステム資源および未来のシステム資源がある。同様に、サービスによって利用者に割り当てられる QoS にも、現在の QoS と未来の QoS がある。図中で、サービス s によって利用者 u に割り当てられる QoS を q_s^u で表す。

図中の円はエージェントを表し、利用者の効用を代表する消費者エージェント (左) と、割り当てられたシステム資源から QoS への変換を代表する生産者エージェント (右) とがある。図中の矢印は財の流れを表し、利用者が持つシステム資源がサービスによって QoS へ変換され、利用者がそれを享受することを表している。また、現在と未来のシステム資源は、各利用者の需要に応じて、利用者間で交換される。

本市場における消費者エージェントはサービス利用者と 1 対 1 に対応する。利用者は、提供者によって割り当てられた QoS を評価する。他方の生産者エージェントはサービスと 1 対 1 に対応し、割り当てられたシステム資源を QoS へ変換する。このモデルにおいて、消費者エージェントの初期保有には QoS は含まれず、生産者エージェントによって割り当てられなければならない。生産者エージェントは、自身の効用が最大になるように、提供可能な QoS と利用者に対する選好、需要に基づいて、QoS を割り当てる。

4.2 現在・未来モデル

現在・未来モデルでは、時間を一定間隔に分割し、現時点からの 1 単位時間を現在、それ以降一定の時間 ($T-1$ 単位時間とする) を未来と定義する。システム全体で保有するシステム資源を β とすると、現在のシステム資源の初期保有量の総計は β 、未来のシステム資源の総和は $(T-1)\beta$ となる。

現在・未来モデルにおける財のやりとりを図 3 に示す。利用者の現在と未来のシステム資源の初期保有量をそれぞれ e_u^c , e_u^f とする。利用者間で現在と未来のシステム資源の交換を行い (図中 1)、現在と未来のシステム資源の割当

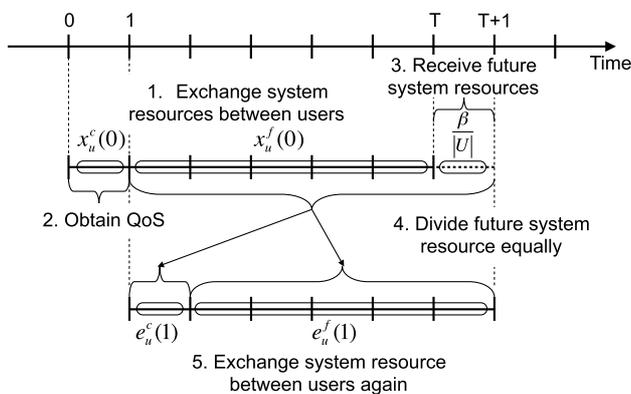


図 3 現在・未来モデル [9]

Fig. 3 Current-Future model [9].

て e_u^c , e_u^f を決定する。利用者は得られたシステム資源を用いて QoS を取得する (図中 2)。次に時刻を 1 単位時間進めて同じプロセスを繰り返すが、このときに前回の未来のシステム資源を次の割当ての際の初期割当てに反映させる。さらに前回の未来のシステム資源に加えて $\beta/|U|$ のシステム資源を与える (図中 3)。各消費者についてこれを $1:(T-1)$ に分割し、次回の現在および未来のシステム資源の初期保有量とする (図中 4)。つまり、利用者 u の時刻 t における現在と未来のシステム資源の初期保有量は次の式で定まる。

$$e_u^c(t) = \frac{1}{T} \left(x_u^f(t-1) + \frac{\beta}{|U|} \right) \quad (4)$$

$$e_u^f(t) = \frac{T-1}{T} \left(x_u^f(t-1) + \frac{\beta}{|U|} \right) \quad (5)$$

以上の初期保有量決定の後、各利用者の効用および提供者の効用を更新したうえで、再び利用者間のシステム資源の交換を実行する (図中 5)。

4.3 ヒューリスティクスを用いた QoS 割当て

市場モデルにおいて最適、またはパレート最適な QoS 割当てを決定するには多くの時間が必要なため、本研究では Kuwabara ら [5] が提案する手法を拡張し、ヒューリスティクスを用いた手法を提案する。提案手法では QoS を割り当てるために、以下の手順を財の割当てが収束するまで繰り返す。

ステップ 1: 利用者は現在と未来に得られる効用を評価し、現在と未来のシステム資源の譲渡量を決定する。譲渡されたシステム資源を市場に従って各利用者に再分配する。

ステップ 2: 利用者は目的の優先度を考慮し効用を効率的に高めるようサービスに対するシステム資源の割当てを決定する。

ステップ 3: 提供者は割り当てられたシステム資源を基に、利用者への選好を考慮して各利用者に割り当てる QoS を決定する。

ステップ 4: 利用者は割り当てられた QoS の消費量を決定し、利用者 と提供者の効用を決定する。

まずステップ 1 では、利用者は需要と供給の反復において、前回のシステム資源の割当てに応じてその譲渡量を調整する。利用者が譲渡するシステム資源の初期量は、利用者の現在と未来の目的の重みの比が、利用者の保有する現在と未来のシステム資源の比と等しくなるように定める。譲渡量の調整は、現在と未来のシステム資源の交換比率と、現在と未来で得られる効用を基に行われる。Algorithm 1 に譲渡量の調整アルゴリズムを示す。

現在のシステム資源 1 単位あたりに得られる効用が、現在のシステム資源 1 単位を他の利用者に譲渡することで得られる未来のシステム資源で取得できる未来の効用よりも

Algorithm 1 Release current and future system resources

1: α : sensitivity factor
 2: i : i -th iteration
 3: $p(i)$: exchange ratio of current system resource to future ones at i
 4: e_u^c, e_u^f : initial current/future system resources
 5: $g_u^c(i), g_u^f(i)$: released current/future system resources at i
 6: $utility_u^c(i), utility_u^f(i)$: current/future utility per unit system resource at i
 7: θ : threshold of released system resources
 8: **if** $p(i-1)utility_u^f(i-1) < utility_u^c(i-1)$ **then**
 9: $(g_u^c(i), g_u^f(i)) = \begin{cases} (0, g_u^f(i-1) + \alpha(e_u^f - g_u^f(i-1))) & (g_u^c(i-1) < \theta) \\ ((1-\alpha)g_u^c(i-1), 0) & (\text{otherwise}) \end{cases}$
 10: **else if** $p(i-1)utility_u^f(i-1) > utility_u^c(i-1)$ **then**
 11: $(g_u^c(i), g_u^f(i)) = \begin{cases} (g_u^c(i-1) + \alpha(e_u^c - g_u^c(i-1)), 0) & (g_u^f(i-1) < \theta) \\ (0, (1-\alpha)g_u^f(i-1)) & (\text{otherwise}) \end{cases}$
 12: **else**
 13: $(g_u^c(i), g_u^f(i)) = (g_u^c(i-1), g_u^f(i-1))$
 14: **end if**

大きい場合、利用者は未来のシステム資源を譲渡するか、現在のシステム資源の譲渡量を削減して現在の効用を高めようとする (8, 9 行目)。逆に現在のシステム資源 1 単位あたりに得られる効用の方が小さい場合、利用者は現在のシステム資源を譲渡するか、未来のシステム資源の譲渡量を削減して未来の効用を高めようとする (10, 11 行目)。

次に、各利用者から譲渡されたシステム資源量と市場の規則に従って、システム資源の再割当てを行う。市場モデルでは、利用者に再割当てされる未来のシステム資源量は、譲渡された現在のシステム資源の総量のうちその利用者が譲渡した割合で定まるとする。現在のシステム資源の再割当ても同様に、譲渡された未来のシステム資源の総量のうちその利用者が譲渡した割合で定まる。したがって、利用者 u に再割当てされる現在と未来のシステム資源 x_u^c, x_u^f は下記の式に基づいて決定する。

$$(x_u^c, x_u^f) = \left(\frac{g_u^f}{\sum_{u' \in U} g_{u'}^f} \sum_{u' \in U} g_{u'}^c, \frac{g_u^c}{\sum_{u' \in U} g_{u'}^c} \sum_{u' \in U} g_{u'}^f \right) \quad (6)$$

ステップ 2 では、利用者は自分の目的の優先度に基づいてサービスに対するシステム資源の割当てを決定する。Algorithm 2 に利用者が現在のシステム資源をサービスに対して割り当てるアルゴリズムを示す。利用者の各目的に対する割当ての初期値は、目的に設定した重みに基づいて決定する。その後は最も効率的に効用を高める目的により多くのシステム資源を割り当て、他の目的に割り当てるシステム資源を減少させる (12 行目)。

次に各目的に割り当てられたシステム資源を用いて、目的に対応するサービスへシステム資源を割り当てる。サービスへの初期割当量は、サービスの選好の値に基づいて決

Algorithm 2 System resource allocation to services

1: α : sensitivity factor
 2: i : i -th iteration
 3: $D_u^c \subseteq D_u$: set of currently active goals ($d \in D_u^c$)
 4: $S_d \subseteq S$: set of services that satisfy d
 5: $x_u^c(i)$: current system resources allocated to u at i
 6: $d_{best}(i) \in D_u^c$: goal bringing u the best utility per unit resource in D_u^c at i
 7: $s_{best}(i) \in S_d$: service providing the best utility per unit resource for d in S_d at i
 8: $rate_d(i)$: rate of system resources allocated to d at i
 9: $rate_s^d(i)$: rate of system resources allocated to s in S_d at i
 10: $x_{us}^c(i)$: system resources allocated to s by u at i
 11: **for all** $d \in D_u^c$ **do**
 12: $rate_d(i) = \begin{cases} rate_d(i-1) + \alpha(1 - rate_d(i-1)) & (d \text{ is } d_{best}) \\ (1-\alpha)rate_d(i-1) & (\text{otherwise}) \end{cases}$
 13: **for all** $s \in S_d$ **do**
 14: $rate_s^d(i) = \begin{cases} rate_s^d(i-1) + \alpha(1 - rate_s^d(i-1)) & (s \text{ is } s_{best}) \\ (1-\alpha)rate_s^d(i-1) & (\text{otherwise}) \end{cases}$
 15: $x_{us}^c(i) = rate_s^d(i)rate_d(i)x_u^c(i)$
 16: **end for**
 17: **end for**

Algorithm 3 QoS allocation to users

1: U_s : users that allocate system resources to s
 2: $q_s^u = 0$: QoS allocated to u ($u \in U_s$) by s
 3: $U_{left} = U_s$: users unsatisfied with QoS
 4: x_{us}^c : current system resources allocated to s by u
 5: v_s^u : preference to u by s
 6: r_d : QoS necessary for u to satisfy d
 7: q_{given} : QoS already allocated to users
 8: $q_{left} = q_s$: remaining QoS that s has
 9: **while** $U_{left} \neq \emptyset$ and $q_{left} > 0$ **do**
 10: $q_{given} = 0$
 11: **for all** $u \in U_{left}$ **do**
 12: $q = \min(q_{left}x_{us}^c v_s^u / \sum_{u \in U_{left}} x_{us}^c v_s^u, r_d - q_s^u)$
 13: $(q_{given}, q_s^u) = (q_{given} + q, q_s^u + q)$
 14: **if** $q_s^u == r_d$ **then**
 15: $U_{left} = U_{left} \setminus \{u\}$
 16: **end if**
 17: **end for**
 18: $q_{left} = q_{left} - q_{given}$
 19: **end while**

定する。目的へのシステム資源の割当てと同様に、目的を満たすサービスの中で最も効率的に効用を高めるサービスにより多くのシステム資源を割り当てる (14 行目)。上記と同様の手順で、未来のシステム資源の割当てを決定する。

ステップ 3 では、提供者が割り当てられたシステム資源を用いて算出した QoS を利用者に割り当てる。提供者による QoS の割当てアルゴリズムを Algorithm 3 に示す。提供者は利用者に対する選好の値と利用者から割り当てられたシステム資源に基づいて、QoS の割当てを決定する。求めた QoS は利用者の必要な QoS よりも小さい場合、その求めた QoS が利用者に割り当てられる。求めた QoS が利

ユーザーの必要な QoS よりも大きい場合、その必要量が利用者に割り当てられる。この手順を QoS の必要な利用者がいなくなるか、割り当てる QoS がなくなるまで繰り返す。

最後にステップ 4 では、利用者が割り当てられた QoS から自身の効用を最大化するように消費する QoS を選択し、利用者と提供者の効用が決定する。

5. シミュレーション

本章では、これまで述べてきた QoS 制御手法の有効性をシミュレーションによって検証する。

5.1 設定

このシミュレーションでは、乱数は一様分布に従うとする。利用者数 $|U|$ を 100、サービス数 $|S|$ を 100 とした。シミュレーションを実行する時間は 0 から 200 とし、利用者がこの期間内で持つ目的の数は 6 から 10 とした。各目的の有効期間は単位時間 10 から 30、目的の達成に必要な QoS 量 r_d を 10 から 20、目的の重み w_d は 0 から 1 とした。目的の達成に利用するサービスの数 $|S_d|$ は、現在運用中の言語グリッドの同一タイプのサービス数の平均値 5 を基準に ± 2 で 3 から 7 とし、サービスに対するユーザの選好 v_u^s は 0 から 1 の乱数を設定した。一方、サービス s が単位時間あたりに提供可能な QoS の最大値 Q_s は 10 から 20 とし、利用者に対する選好 v_u^s は 0 から 1 の乱数を設定した。また、現在・未来モデルで資源をやりとりする際に考慮する期間 T を 20 とした。利用者がシステム資源の割当てを調整する際の sensitive factor α を 0.01 とした。利用者に発行するシステム資源の総量は 1,000 とし、各利用者に均等に割り当てるものとした。利用者の数を 100 としているため、各利用者には単位時間ごとに 10 のシステム資源が割り当てられる。

5.2 結果

ある利用者に関して、目的の重み付け、消費したシステム資源量、効用のそれぞれの変化を図 4 に示す。各図の水平軸はシミュレーション上の時刻を表す。1 つ目のグラフの垂直軸は各時刻における目的の重み w_d の総和を表す。2 つ目のグラフの垂直軸は利用者が単位時間に QoS を取得するのに消費した現在のシステム資源量を表す。最後に 3 つ目のグラフの垂直軸は QoS 割当ての結果利用者が得た効用の大きさを表す。

この利用者は、時刻 17 から 34 まで重みの小さな（優先度の低い）目的を持ち、その後時刻 42 から 87、98 から 123、142 から 168、173 以降に比較的重みの大きな目的を持つ。時刻 16 までは、利用者は目的を持たないため、現在のシステム資源をすべて譲渡し、未来のシステム資源を取得する。そうして蓄えた未来のシステム資源を時刻 17 から使用し始める。このとき、考慮する期間を $T = 20$ とし

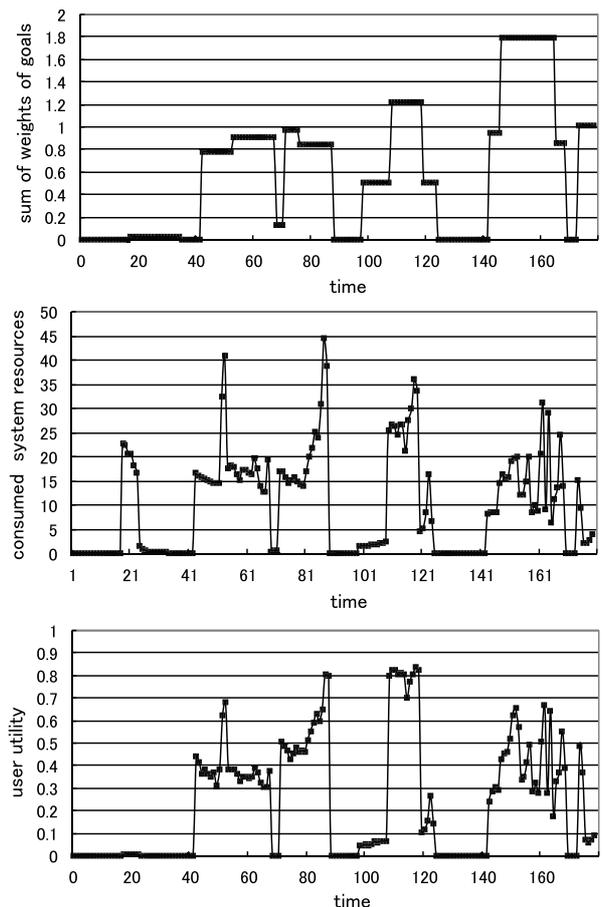


図 4 利用者の目的の重み、消費したシステム資源量、効用の推移
Fig. 4 Goal's weight, consumed system resources, and utility of a user.

ため、時刻 22 までは比較的多くのシステム資源を使用して QoS を取得する。時刻 23 からは時刻 42 から始まる比較的重みの大きい目的を考慮して、現在のシステム資源の消費を減らして残りを譲渡し、未来のシステム資源を取得する。時刻 42 からはそれまで取得してきたシステム資源を用いることで、QoS を取得して効用を高めることができる。同様に、時刻 122 では、時刻 142 から始まる比較的重みの大きい目的を考慮して、現在のシステム資源の消費を抑えている。以上から、提案手法によって、利用者は自分の目的の優先度に合わせて QoS の獲得に必要なシステム資源を取得し、効用を高めることを確認できる。

一方、提供者に関しては、利用者への選好と割り当てられたシステム資源に基づいて QoS を割り当てるため、より好ましい利用者からより多くのシステム資源が割り当てられることで、提供者は効用を高めることができる。

次に、疑似需要による効用への影響を検証する。疑似需要には必要以上の量の QoS を要求するものと、必要以上の期間で QoS を要求するものがある。どちらもサービスが無償であることで利用者が保険的意味合いで過剰な QoS を要求して生じる。たとえば、言語グリッドでは翻訳サービスを用いた多言語コミュニケーション実験で、実験の準

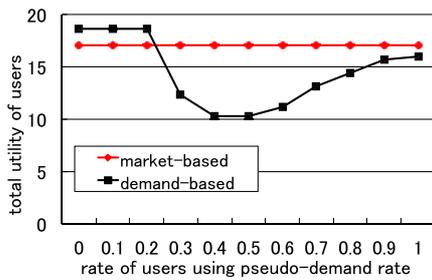


図 5 量的な疑似需要による利用者の効用への影響

Fig. 5 Effects of quantitative pseudo-demands on utility of users.

備や不測の事態による実験の延長などを考慮して必要以上に長い時間 QoS を確保する場合がある。また、翻訳サイトなどの一般公開時には、サービスが停止しないように、予想よりも多い QoS を確保する傾向がある。

まず、必要以上の QoS を指定する利用者（量的な疑似需要を用いる利用者）の割合を変化させ、利用者の効用の変化を比較した。量的な疑似需要を用いる利用者は、必要とする QoS の 2 倍を要求し、実際には必要とする QoS のみ消費するとする。図 5 に、量的な疑似需要を用いる利用者の割合に対する利用者の効用の総和の推移を示す。

疑似需要を用いる利用者の割合が少ないうちは、単純に需要に基づいて QoS を割り当てた方が効用が高まる。これはシステム資源を余らせることなく QoS に変換し利用者に提供できたためである。しかしながら、疑似需要を用いる利用者の割合が多くなるにつれて、必要としている利用者に QoS が割り当てられなくなり利用者全体の効用が下がり始める。最終的に、全利用者が必要以上の要求を行うため効用は元のレベルに近づく。一方、提案手法では、必要以上の QoS を要求したとしても、サービスに割り当てられたシステム資源量に応じて QoS の割当てが行われるため、利用者の効用への影響はない。これは、不必要な QoS を要求したとしても効用があがらないことを利用者は分かっているため、無駄なシステム資源を割り当てないからである。

次に、必要以上の期間で QoS を要求する利用者（時間的な疑似需要を用いる利用者）の割合を変化させ、利用者と提供者の効用の変化を比較した。時間的な疑似需要を用いる利用者は、目的の期間を 0 から 200 に指定し、実際には必要とする期間のみ QoS を消費するとする。図 6 に、時間的な疑似需要を用いる利用者の割合に対する利用者の効用の総和と、提供者の効用の総和の推移をそれぞれ示す。

疑似需要を用いる利用者が増加すると、利用者と提供者の効用はともに減少する。これは量的な疑似需要とは異なり時間的な疑似需要では、疑似需要を入力するために必ず無駄なシステム資源を割り当てる必要があり、これにより真の需要に割り当てられる QoS が減少しているためである。真の需要に対する QoS 割当てが減少することで、QoS を消費することで効用が高まる利用者と、QoS が消費され

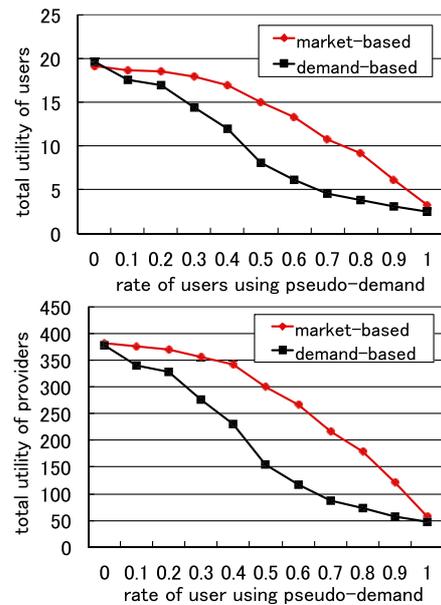


図 6 時間的な疑似需要による利用者および提供者の効用への影響

Fig. 6 Effects of temporal pseudo-demands on utility of users and providers.

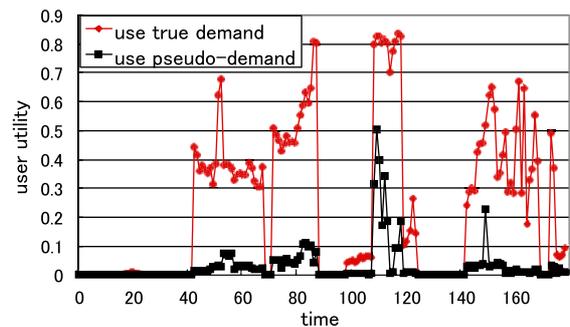


図 7 疑似需要と真の需要による利用者の効用の推移の比較

Fig. 7 Utility of a user using pseudo-demands and true demands.

ることで効用が高まる提供者の両者の効用が減少する。

このような疑似需要を用いる利用者による効用の減少は、QoS の需要の割合に応じて単純に割り当てるよりも提案手法の方が少なくすることができる。これは、真の需要を記述した利用者は、需要に応じてシステム資源を他の利用者とは交換できるため、疑似需要を用いた場合に比べて、消費できる現在のシステム資源が大きくなるからである。そのため単純な需要の割合に基づく割当て手法に比べて、疑似需要による影響を小さくすることができる。

時間的な疑似需要を用いる利用者が増加すると利用者も提供者の効用が低下するため、利用者にも真の需要を記述するよう動機付けを行う必要がある。ある利用者において真の需要を記述する場合と疑似需要を記述する場合とで得られる効用を比較した。このシミュレーションでは、注目している利用者以外の利用者は真の需要を入力するとする。

提案手法において真の需要のみを示した場合と、疑似需要を用いた場合とで利用者が得る効用の比較を図 7 に示

す。利用者が疑似需要を用いた場合、つねに QoS を取得しようとするため、本来必要とする時刻にシステム資源が少なくなる。そのため、真の需要のみを示した場合と比較して、疑似需要を用いた場合はつねに効用が小さくなる。したがって、提案手法は利用者に対してつねに真の需要のみを示すよう動機付けを行うことができる。

6. おわりに

本研究では、ボランティアクラウドにおける動的な QoS 割当て手法を提案した。本手法は利用者と提供者が互いに持つ選好に基づいて QoS を割り当てる。この QoS 制御を実現するための本研究の貢献は以下の 2 点である。

- ボランティアクラウドの QoS 割当てのモデル化
提供者の効用は利用者への選好と利用者による提供サービスの QoS の消費量との積の総和で定式化した。一方、利用者の効用は、ボランティアクラウド上に相互交換可能なサービスが複数あることを考慮し、利用者の目的への選好と利用したサービスへの選好のそれぞれに基づく 2 段階で定式化を行った。これにより、利用者と提供者の互いの選好に合わせて QoS の割当てが推移することをシミュレーションにより確認した。
- 疑似需要の抑制メカニズム
ボランティアクラウドにおいて適切な QoS 割当てを実現するために、現在・未来モデルと消費者・生産者モデルを用いた市場モデルを提案した。シミュレーションにより、提案手法が疑似需要よりも真の需要の方が利用者および提供者の効用を高めることを確認し、利用者に真の需要を示すよう動機付けることを示した。

謝辞 本研究は、日本学術振興会科学研究費基盤研究 (S) (24220002, 平成 24 年度~28 年度) の補助を受けた。

参考文献

- [1] Anderson, D.P. and Fedak, G.: The Computational and Storage Potential of Volunteer Computing, *Proc. 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID '06)*, pp.73-80 (2006).
- [2] Buyya, R., Abramson, D. and Venugopal, S.: The Grid Economy, *Proc. IEEE*, Vol.93, No.3, pp.698-714 (2005).
- [3] Galstyan, A., Czajkowski, K. and Lerman, K.: Resource Allocation in the Grid Using Reinforcement Learning, *Proc. 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '04)*, Washington, DC, USA, pp.1314-1315 (2004).
- [4] Ishida, T.: Language Grid: An Infrastructure for Inter-cultural Collaboration, *Proc. International Symposium on Applications on Internet (SAINT '06)*, pp.96-100 (2006).
- [5] Kuwabara, K., Ishida, T., Nishibe, Y. and Suda, T.: *An Equilibratory Market-based Approach for Distributed Resource Allocation and Its Applications to Communication Network Control*, pp.53-73, World Scientific Publishing Co., Inc. (1996).
- [6] Marosi, A., Kovács, J. and Kacsuk, P.: Towards a Volunteer Cloud System, *Future Generation Computer Systems*, Vol.29, No.6, pp.1442-1451 (2013).
- [7] Menascé, D.A. and Dubey, V.: Utility-based QoS Brokering in Service Oriented Architectures, *Proc. IEEE International Conference on Web Services (ICWS '07)*, pp.422-430 (2007).
- [8] Weng, C., Li, M., Lu, X. and Deng, Q.: An Economic-based Resource Management Framework in the Grid Context, *Proc. 5th IEEE International Symposium on Cluster Computing and the Grid (CCGrid '05)*, Vol.1, pp.542-549 (2005).
- [9] Yamaki, H., Wellman, M.P. and Ishida, T.: Controlling Application QoS Based on a Market Model, *The Transactions of the Institute of Electronics, Information and Communication Engineers*, Vol.81, No.5, pp.540-547 (1998).
- [10] Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J. and Chang, H.: QoS-Aware Middleware for Web Services Composition, *IEEE Trans. Softw. Eng.*, Vol.30, No.5, pp.311-327 (2004).
- [11] Zhang, L.-J.: TSC Cloud: Community-Driven Innovation Platform, *IEEE Trans. Services Computing*, Vol.2, No.1, pp.1-2 (2009).



村上 陽平 (正会員)

2006 年京都大学大学院情報学研究科社会情報学専攻博士課程修了。同年 (独) 情報通信研究機構入構。博士 (情報学)。現在、京都大学大学院情報学研究科社会情報学専攻特定研究員。電子情報通信学会サービスコンピューティング研究専門委員会委員長を務める。



石田 亨 (フェロー)

1976 年京都大学工学部情報工学科卒業、1978 年同大学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、工学博士。IEEE フェロー。デジタルシティ、言語グリッド、異文化コラボレーション等情報技術と社会をつなぐ研究プロジェクトを推進。



宮田 直輝

2008 年京都大学大学院情報学研究科社会情報学専攻修士課程修了。大規模マルチエージェントシステムに関心を持つ。