修 士 論 文 の 和 文 要 旨

研究科・専攻	大学院　情報理工学研究科　情報・ネットワーク工学　専攻　博士前期課程		
氏　　名	松枝友佳	学籍番号	1731148
論 文 題 目	クラウドソーシング型ピックアップアンドデリバリー問題のモデル化と経済分析		

要　旨

近年モバイルデバイスの普及と共に位置・空間情報を要する空間クラウドソーシングという新たな分類が生まれた．空間クラウドソーシングは，対象業務であるタスク，タスクを実行するワーカ，タスクとワーカを管理し，それらを結びつけるサーバの 3 つで構成される．物理的に人やモノが移動する業務にもこの空間クラウドソーシングは適用されている．

本研究の目的は，空間クラウドソーシングを利用したフードデリバリーサービスのモデル化及びワーカとタスクの情報がそれぞれ動的にサーバに到着するようなオンラインモデルに対して，サーバ全体のコストが最小となるようなタスク割当を行うオンラインアルゴリズムの提案である．特にワーカが自身にとって有益なタスク選択を行う場合に対して有用なサーバのタスク割当方法について提案を行い，計算機実験を通して提案アルゴリズムの性能評価及び経済分析を行う．

オフラインタスク割当問題をワーカとタスクそれぞれにタイムウィンドウ制約を有しかつ異なる前処理時間付き並列機械スケジューリング問題として，ワーカが拒否権を持つ場合持たない場合でそれぞれ定式化を行なった．動的にワーカにタスクを割当てるオンラインアルゴリズムとして，ワーカとタスクそれぞれが到着した順に割当対象となっているものからマッチングを行う Algorithm1 : FIFO と，ワーカのタスクに対する選好タイプによって既にサーバに到着しているタスクに対してワーカ毎に順位付けを行い，上位のものから割当を行う Algorithm2 : RANK の 2 つを提案した．

ワーカがタスクの選択権を有する場合，Algorithm1 : FIFO よりも Algorithm2 : RANK の方がタスクの割当率が高く，またワーカのタスク拒否率が低い結果となった．配送効率性を見ると，インスタンスによってはオフラインの場合でもその値が 0.5 を下回る結果となった．オンラインの場合，Algorithm1 : FIFO を適用した時の配送効率性が最大でも 0.3 程度であったのが，Algorithm2 : RANK を適用した時は最大で 0.4 を超える結果となった．

電気通信大学大学院情報理工学研究科
情報・ネットワーク工学専攻情報数理工学プログラム修士論文

クラウドソーシング型ピックアップアンドデリバリー問題のモデル化と経済分析

令和 2 年 1 月 27 日

情報数理工学プログラム

学籍番号 1731148

松枝友佳

指導教員　高橋里司
村松正和

目次

1　はじめに

1.1　研究背景

日本国内では 1997 年にサービスを開始した楽天が電子商取引 (e コマース) 業界最大手である．その楽天ではオークションサイトやオンライン証券取引，旅行代理業など様々なサービスを展開しており，2018 年の流通取引総額は 3.4 兆円に達した [1]．米国発の amazon も日本でのサービス提供を行っており，野村総合研究所の調査によると国内の B to C e コマース全体の市場規模は，2014 年度の 12.6 兆円から 2021 年度には倍増の 25.6 兆円に達する見込みだという [2]．経済産業省の調査によると，世界的に見ると日本の e コマース市場の規模は中国，アメリカ，イギリスに次いで 4 位となっている．ただしその成長率は他国と比較して鈍化傾向である [3]．

個人等が有する活用可能な資産等をインターネット上のマッチングプラットフォームを介して，他の個人等も利用可能とする経済活性化活動のことをシェアリング・エコノミーまたは共有経済社会と呼び，日本社会においても市場の拡大が続いている [4]．活用可能な資産等はスキルや時間などの無形のものも含まれる．代表的なサービスとして，使用していない物件を旅行者に貸し出す Airbnb[5] や配車サービスを行う Uber[6]，滴滴出行 [7]，個人間でモノの売買を行うことができるメルカリ [8]，クラウドファンディングサービスを提供する Readyfor[9] 等が挙げられる．

Uber eats など飲食店とは関係のない配達員がデリバリーを行うクラウドソーシング型フードデリバリーサービスは，配達員を飲食店間で共有するという点で共有経済社会の一部である．クラウドソーシングとは従来従業員などが行っていた業務を大規模な人々に公募の形で外部委託することである [15]．リモートで行えるような作業，例えば文章の作成，翻訳，プログラムのデバッグなどに利用されてきた．社員だけでは間に合わない作業がある会社が，会社とは関係はないが能力のある人材に対してその作業を委託する．作業を委託する会社と委託される人とを結びつけるプラットフォームが，世界規模では Amazon が運営を行う Amazon Mechanical Turk(MTurk)[11] や Upwork[12] など多数存在する．小さな子供がいるため在宅で隙間時間に勤務希望の母親向けプラットフォームや，コンペ形式でデザインを募集するプラットフォームなど日本に存在するクラウドソーシングプラットフォームの種類や形態も多岐にわたる．

近年ではモバイルインターネットの普及と共有経済社会の発達により，主にウェブ上で行われていたクラウドソーシングが，位置・時間情報を要する空間クラウドソーシングと

呼ばれる形にシフトしてきている [18]．この空間クラウドソーシングは 3 つの要素で構成されている．クラウドソーシングの対象であるタスク，そのタスクを行うワーカ，タスクとワーカを結びつけるプラットフォームもしくはサーバである．空間クラウドソーシングは各地の気象情報，レストランの評価，大きな建物の外観の写真などの収集に利用されてきた．最近では情報のやり取りだけではなく，物理的に人やモノが移動することにも利用され，配車サービスの Uber，滴滴出行，食料配達サービスの GrubHub[13]，美団 [14]，Uber eats などはまさにこの空間クラウドソーシングを利用している例である．また，実際に存在する空間クラウドソーシングを利用しているシステムで扱うデータはとても大きいことが知られている．例えば中国の配車サービスである滴滴出行では，2017 年のデータで 1 日に 2500 万ほどの配車要求を登録している 2100 万人のドライバーで応えなければならない．結果として 70TB を越える時空間データを毎日扱っている [18]．ワーカとタスクを結びつける高速で正確な手法は空間クラウドソーシングにおいてとても重要な事項である．

1.2 研究目的

本研究の目的は，タスクとワーカ両方が動的にそれぞれサーバに到着するような空間クラウドソーシングを利用したフードデリバリーサービスのモデル化である．更にサーバ全体で発生するコストが最小化となるようなタスク割当てのオンラインアルゴリズムの提案を行う．また，ワーカが自身にとって有益なタスク選択を行う場合に対してのサーバのタスク割当て方法についても提案を行う．計算機実験を通して，オンラインアルゴリズムの性能評価，適切な報酬の決定方法などの経済分析を行う．全体の評価方法としてはオフライン問題として解いた場合との競合比を見ることとする．

1.3 本論文の構成

本章では研究を行うにあたり社会的な背景と本研究の目的を述べた．2 章では本研究で取り扱うクラウドソーシング及び空間クラウドソーシングの定義を行い，モデル化と提案手法で用いるスケジューリング問題と二部グラフのマッチング問題に関して述べる．3 章ではクラウドソーシングを用いたフードデリバリーサービスに関する先行研究と二部グラフマッチング問題に対するオンラインアルゴリズムの紹介，4 章では本研究で取り扱うモデルの定義について述べる．5 章でそれぞれ動的にサーバに到着するワーカとタスクに対し，ワーカがタスクを選択する場合のサーバ側のコスト最小化を目指す提案オンライン

マッチングアルゴリズムについて述べる．6 章で計算機実験についてとその結果を記述，7 章で考察を行う．

2　準備

本章では本研究で扱うクラウドソーシングおよび空間クラウドソーシングの定義を行い，本研究のモデル化において関連するスケジューリング問題及び二部グラフのマッチング問題に関して述べる．

2.1　空間クラウドソーシング

クラウドとアウトソーシングの造語であるクラウドソーシングを Howe[15][16] が次のように定義した．

定義 1:クラウドソーシング (翻訳)[15][16]

指定されたエージェント (通常は従業員) によって行われていたこれまでの仕事を引き受け，それを不特定の一般に大規模な人々のグループに公募の形で外部委託する行為

このクラウドソーシングは従来文章の作成や翻訳，広告のデザインなど机上で行える作業に対して用いられ，作業の管理などはウェブ上で行われていた．近年スマートフォンなどのモバイルデバイスやモバイルネットワークの拡充によって，位置・時間情報を有する空間クラウドソーシングという新たな分類が生まれた [18]．空間クラウドソーシングでは位置や時間の情報が重要な各地の気象情報や，レストランの評価等の収集に用いられてきたが，物理的に人やモノが移動を要するタスクに対しても用いられるようになってきている．配車サービスやモノ・食料等の配達サービスなどである．空間クラウドソーシングを構成する要素は次の 3 つである．

- タスク:クラウドソーシングの対象とする仕事
- ワーカ:タスクを実際に実行する不特定多数の潜在的労働者
- サーバ:ワーカとタスクの情報を保持しそれらを結びつける管理者

空間クラウドソーシングに関する研究は主に次のような事柄に関して行われている．まずは本研究で扱う位置的制約，時間的制約を守りながら特定の目標を達成するようなワーカとタスクの組合せを考えるタスク割当問題である．その他には質の高いタスクの完成を目

指すための質のコントロールやワーカの勤務意欲を高めるための最適な報酬メカニズムや位置・時間情報を扱うため個人情報の保護方法に関して行われている．特にタスク割当問題については表 1 に示すように，タスクとワーカがサーバに到着するのが静的なのか動的なのか，またワーカとタスクのマッチングのみを考えるのか，タスクがどのような順番で行われるのかを考えるプランニングをするのかで 4 つに分類することができる．本研究ではタスクとワーカがそれぞれサーバに動的に到着する場合を想定し，タスクとワーカの動的なマッチングを考える．

表 1　空間クラウドソーシングにおけるタスク割当問題の分類

		到着過程	
		静的	動的
アルゴリズムのモデル	マッチング	**静的マッチング**	**動的マッチング**
	プランニング	**静的プランニング**	**動的プランニング**

2.2　スケジューリング問題

製造現場やサービス産業において一定時間与えられるジョブやタスクに対して，1 つもしくは複数の目的の最適化のために資源の配分を考える問題をスケジューリング問題という [19]．資源の数に比べて数が多いジョブやタスクに対して，限られた数しかない資源をどう割当てるのかを考える．例えば工場で製品を作るときに機械をどのように使うのか，イベントを行う際にスタッフをどのように配置するのかなどである．特に資源が 1 つしかないような場合を 1 機械スケジューリング問題，資源が複数あり互いに干渉せず同時に複数のタスクを並列して行うことができる場合を並列機械スケジューリング問題と呼ぶ．目的となる最適化の対象は実行しなければならない全タスクの完了時間最小化や，タスクによって報酬が異なる場合の報酬最大化などが挙げられる．

2.3　オンラインスケジューリング問題

タスク数やタスクが到着する時刻など将来の情報が未知な場合に，現在わかっているタスクに資源をどのように割当てるのかを考える問題をオンラインスケジューリング問題という．例として物のレンタルサービスを行うオーナを考える [25]．レンタル依頼が来たとき，依頼対象の在庫があれば貸し出すことはできるが，もしかしたらもう少し後により条

件の良い依頼が来るかもしれない．結果としてより多くの報酬を得るためには現在の依頼を受けた方が良いのか，それとも将来に来るかもしれないより条件の良い依頼を受けるために現在の依頼は断った方が良いのかを考える必要がある．

オンラインスケジューリング問題を解くためのオンラインアルゴリズムの性能評価を行う指標として，競合比 (Competitive Ratio，CR) というものがある [19]. オンラインスケジューリング問題のインスタンス L のオフラインにおける最適値を $\mathrm{OPT}(L)$，オンラインアルゴリズム A により得られる評価値を $v_A(L)$ とする．実数 $a \in \mathbb{R}$ に対して，最適化問題におけるオンラインアルゴリズム A の競合比が a であるとは任意のインスタンス L に対して常に

$$\frac{\mathrm{OPT}(L)}{v_A(L)} \leq a$$

が成り立つことを意味する．

2.4 二部グラフの最大マッチング問題

無向グラフ $G = (V, E)$ に対して，端点を共有しない辺の集合 $M \subseteq E$ をマッチングという．任意のマッチング $M' \subseteq E$ に対して常に $|M| \geq |M'|$ となるようなマッチング $M \subseteq E$ を求める問題を最大マッチング問題という [17]．ただし V は頂点集合，E は辺集合である．頂点集合 V を 2 つの集合 L，R に分割可能でかつ任意の辺 $(u, v) \in E$ が $u \in L$, $v \in R$ である時，グラフ $G = (V, E)$ は二部グラフであるという．

3 関連研究

本章ではクラウドソーシング型フードデリバリーサービスに関する先行研究とオンライン二部グラフマッチングアルゴリズムの先行研究に関して述べる．

3.1 クラウドソーシング型フードデリバリーサービス

デリバリータスクとワーカであるドライバーがそれぞれ動的にサーバに到着するクラウドソーシング型デリバリーサービスに関して Arslan ら [20] の研究がある．彼らはドライバー自身の移動途中，例えば通勤通学時にデリバリーサービスを行うことを想定し，サーバ全体のコストが最小となるようなデリバリータスクとドライバーのマッチング，各ドライバーのルート決定を行っている．タスクの完遂のためにサーバ側が支援車両を用意し，実行可能なドライバーが存在しない場合はその支援車両を用いる．ただしワーカに支払う

報酬よりも支援車両を用いたときにかかるコストが高くなるように設定されている．想定しているモデルはタスク，ワーカ双方が動的に到着するオンラインモデルであるが，新しいタスクもしくはドライバーが到着するたびに，その時点までにまだマッチングが行われていないタスクとドライバーを対象に，オフライン問題とみなしてタスクとドライバーのマッチング問題を解く Rolling horizon 方式を採用している．求解高速化の仮定として，各ドライバーが訪れることのできる地点数を制限することで，実行可能なルート数を削減している．タスクとドライバー数がそれぞれ 100，配達地域を 15km 四方の範囲に限定した場合，現実的な時間で実行可能解を得ることができるという結果を示した．ただしルート数削減の仮定ではドライバーが訪れることのできる地点数をかなり小さく，2 地点としているので，ワーカ 1 人につきこなせるタスク数が少なく，各ワーカが得ることのできる報酬は少ない．またタスクとワーカに関して到着や受託に関する確率的情報を用いていない．

3.2 オンライン二部グラフマッチングアルゴリズム

二部グラフの両側が動的に変化する両側オンライン二部グラフに対するマッチングアルゴリズムに関する研究が行われている [18]．両側オンライン二部グラフマッチングアルゴリズムは，臓器移植現場における提供者と移植希望者，就職市場における求職者と企業，モノの売買を行う市場における売り手と買い手，ライドシェアリングプラットフォームにおけるドライバーと乗車希望者などのマッチングを考えるのに応用されている．簡単な貪欲法を用いた場合の競合比は 1/2 である．

Huang ら [21] は，両側がオンラインである二部グラフの最大マッチング問題に対して，到着した頂点の値をランダムに決定し，隣接する頂点の中でその値が最小な頂点とマッチングさせるオンラインマッチングアルゴリズムを提案している．その競合比が 0.5541 から 0.5671 の間であることを示した．Huang ら [21] らのオンラインマッチングアルゴリズムにおいてマッチングを解く時間ステップは新たに頂点が到着する時か，ある頂点の期限が来た時の 2 種類を考えている．Huang ら [21] のオンラインマッチングアルゴリズムを Algorithm1 : 順位付けアルゴリズムとして示す．1 度マッチングが決定されるとそのやり直しは不可とする．また，頂点 v の順位付けを行うために与える値を y_v とし，頂点 v と辺を共有している頂点の集合を $N(v)$ とする．

また Wang ら [22] は，買い手と売り手双方が動的に到着するような経済市場で取引されるモノの供給を需要と同等にする市場清算を想定し，その一般化として，両側オンラインの二部グラフの b-マッチング問題に対して，オンラインアルゴリズムを提案している，

Algorithm 1 順位付けアルゴリズム [21]

(1) 頂点 v が到着: $y_v \in [0,1)$ をランダムに決定

(2) 頂点 v の期限:

if 頂点 v がまだマッチングしていない **then**

　if 頂点 v の近傍にまだマッチングしていない頂点がある **then**

　　y_v が最小な $u \in N(v)$ とマッチングさせる

　else 頂点 v はマッチング不可能

Wang ら [22] はその競合比が 0.526 であるということも示した.

Hassan ら [23] は，プラットフォームが割当てたタスクをワーカが拒否可能である時のオンラインの場合を多腕バンデット問題を用いてモデル化を行なった．二部グラフにおいて片側が拒否権を有する場合のオフラインにおける定式化は，

$$q_{uv} = \begin{cases} 0 & (\text{頂点 } u \text{ は頂点 } v \text{ とマッチング拒否}) \\ 1 & (\text{otherwise}) \end{cases}$$

を用いて，

$$\begin{aligned} \max \quad & \sum_{(u,v)\in E} q_{uv} x_{uv} \\ \text{s.t.} \quad & \\ & \sum_{u\in N(v)} x_{uv} \leq 1, \quad {}^{\forall} v \in V \\ & x_{uv} \geq 0, \qquad\quad {}^{\forall}(u,\, v) \in E, \end{aligned}$$

となる.

Tong ら [24] は特に配車サービスを想定した空間クラウドソーシングに対して，ワーカが事前にタスクを割当てられる確率を上げるために場所の移動を許可するような場合のモデル化を行い，競合比が 0.399 であるオンラインアルゴリズムを提案している.

4　提案モデル

本章では本研究の対象であるクラウドソーシング型フードデリバリーサービスの数理モデルについて述べる．

4.1　クラウドソーシング型フードデリバリーサービス

タスクとワーカの情報がそれぞれが動的に到着するオンラインの空間クラウドソーシングサーバを考える．サーバはそれぞれ動的に到着するタスクとワーカの実行可能な組合せに対し，サーバ全体で発生するコストが最小となるような組合せを選択する．ワーカはサーバから割当てられたタスクを実行することで報酬を得ることができる．ワーカはサーバから割当てられたタスクに対して実行するか否かの選択権を所有しており，意思に沿わないタスクが割当てられた場合には拒否することもできる．

4.2　サーバのオフラインタスク割当問題

タスクとワーカが動的に到着するオンラインモデルを考える前に，あらかじめタスクとワーカの状況がわかっているオフラインタスク割当問題を考える．タスクの有限集合 T とワーカの有限集合 D を考える．ワーカ $d \in D$ は出発地 (初期位置)o_d，勤務可能時間 $[e_d, \ell_d]$，速度 s_d を情報として持つ．タスク $t \in T$ はタスクの実行報酬 f_t，配達希望時間帯 $[e_t, \ell_t]$，店の場所 l_{o_t}，配達先 l_{d_t} を情報として持つ．配達希望時間はワーカが配達先に到着する時間である事に注意する．タスクの実行報酬 f_t は，ワーカがタスク t の店に到着し料理を受け取る時に発生する受け取り報酬，ワーカが料理をタスク t の配達先に配達した時に発生する受け渡し報酬の 2 つの固定報酬と，店から配達先までの距離に比例する移動報酬の和である．タスクの実行報酬はタスク依存であることに注意する．またあるワーカ d がタスク $\tilde{t}$ の直後にタスク $t \neq \tilde{t}$ を行うためにかかる準備時間を $r_{d\tilde{t}t}$ とする．

$$r_{d\tilde{t}t} = \begin{cases} \frac{\|o_d - l_{o_t}\|}{s_d} & (\tilde{t} = 0) \\ \frac{\|l_{d_{\tilde{t}}} - l_{o_t}\|}{s_d} & (\tilde{t} \neq 0). \end{cases}$$

図 1 に連続して行われるタスク $\tilde{t}$ と t の時間関係を示す．配達員の現在地から店までの移動にかかる準備時間 $r_{d\tilde{t}t}$ を前処理時間，店から配達先までの移動にかかる時間 $p_t = \frac{\|l_{d_t} - l_{o_t}\|}{s_d}$ を処理時間とし，オフラインタスク割当問題 (OTAP) をタスクとワーカそれぞれにタイムウィンドウ制約を有しかつ異なる前処理時間付き並列機械スケジューリン

グ問題として定式化を行う．ワーカにタスクを割当てることを表す 0-1 変数 $x^d_{\tilde{t}t}$ を以下のように定義する．

$$x^d_{\tilde{t}t} = \begin{cases} 1 & \text{ワーカ } d \text{ にタスク } t \text{ をタスク} \tilde{t} \text{の直後に割当てる} \\ 0 & \text{otherwise.} \end{cases}$$

$x^d_{0t} = 1$ である時は，タスク t がワーカ d にとって最初のタスクであることを表し，$x^d_{\tilde{t}0} = 1$ である時は，タスク $\tilde{t}$ がワーカ d にとって最後のタスクであることを表す．タスク t の処理が開始される時刻，つまりあるワーカがタスク t の店 l_{o_t} を出発する時刻を表す時間変数は a_t とする．目的関数は，ワーカへの報酬とタスクが実行されなかった時に発生する配達失敗コスト γ の総和である．したがって，次のように目的関数を定義する．

$$\sum_{t\in T}\{f_t \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T} x^d_{\tilde{t}t} + \gamma(1 - \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T} x^d_{\tilde{t}t})\}.$$

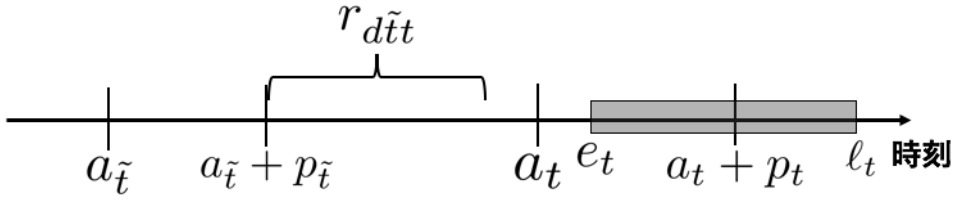


図 1　タスクに関する時間関係

式 (1) から (11) にオフラインタスク割当問題 (OTAP) を示す．

(OTAP)

$$\min \quad \sum_{t\in T}\{f_t \sum_{d\in D}\sum_{\tilde{t}\in T} x^d_{\tilde{t}t} + \gamma(1 - \sum_{d\in D}\sum_{\tilde{t}\in T} x^d_{\tilde{t}t})\} \tag{1}$$

s.t.

$$\sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T\setminus\{t\}} x^d_{\tilde{t}t} \le 1, \qquad {}^\forall t \in T \tag{2}$$

$$\sum_{t\in T} x^d_{0t} \le 1, \qquad {}^\forall d \in D \tag{3}$$

$$\sum_{\tilde{t}\in\{0\}\cup T\setminus\{t\}} x^d_{\tilde{t}t} - \sum_{\tilde{t}\in\{0\}\cup T\setminus\{t\}} x^d_{t\tilde{t}} = 0, \qquad {}^\forall d \in D^\forall,\ t \in T \tag{4}$$

$$a_{\tilde{t}} + p_{\tilde{t}} + r_{d\tilde{t}t} - a_t \le M_1(1 - x^d_{\tilde{t}t}), \qquad {}^\forall d \in D,\ {}^\forall t, \tilde{t} \in T, \tilde{t} \ne t \tag{5}$$

$$a_t - e_d - r_{d0t} \ge M_2(x^d_{0t} - 1), \qquad {}^\forall d \in D,\ {}^\forall t \in T \tag{6}$$

$$a_t - \ell_d + p_t \le M_3(1 - x^d_{t0}), \qquad {}^\forall d \in D,\ {}^\forall t \in T \tag{7}$$

$$a_t \ge \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T}(e_t - p_t)x^d_{\tilde{t}t}, \qquad {}^\forall t \in T \tag{8}$$

$$a_t \le \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T}(\ell_t - p_t)x^d_{\tilde{t}t}, \qquad {}^\forall t \in T \tag{9}$$

$$x^d_{\tilde{t}t} \in \{0,\ 1\}, \qquad {}^\forall d \in D,\ {}^\forall t \in T,\ {}^\forall \tilde{t} \in \{0\} \cup T\setminus\{t\} \tag{10}$$

$$a_t \in \mathbb{R}_+, \qquad {}^\forall t \in T. \tag{11}$$

M_1，M_2，M_3 は十分大きい定数である．式 (1) はタスク割当てで発生する総コストである．タスク t がいずれかのワーカにより実行される場合タスクの実行報酬が発生，どのワーカにも実行されない場合は配達失敗コスト γ が発生する．式 (2) は各タスクが行われるのは高々 1 つであるという制約，式 (3) は各ワーカが最初に行うタスクは高々 1 つであるという制約である．式 (4) はフロー制約であり，あるタスクが実行されるならばその前後に必ず実行されるタスクが存在するという制約である．また，ワーカ d がタスク t の次にタスク $\tilde{t}$ を実行する場合，図 1 からもわかるようにそれぞれの実行時間 $a_t, a_{\tilde{t}}$ は $a_t + p_t + r_{\tilde{t}td} \leq a_{\tilde{t}}$ という関係が成り立っていなければならない．これがワーカ d がタスク t の次に $\tilde{t}$ を実行する場合のみ成り立っていれば良いので式 (5) は連続するタスクの両立性を表している．ここで $a_{\tilde{t}} + p_{\tilde{t}} + r_{\tilde{t}td} - a_t \leq a_{\tilde{t}} + p_{\tilde{t}} + r_{\tilde{t}td} \leq \ell_{\tilde{t}} + p_{\tilde{t}} + r_{\tilde{t}td}$ より $M_1 = \ell_{\tilde{t}} + p_{\tilde{t}} + r_{\tilde{t}td}$ でよい．各ワーカに対して，ワーカにとって最初に実行するタスクの前処理開始時刻はワーカのタイムウィンドウ開始時刻以降，つまりワーカ d がタスク t を最初に実行するタスクとする場合 $a_t - r_{d0t} \geq e_d$ という関係が成り立っていなければならない．また，ワーカにとって最後に実行するタスクの処理終了時刻はワーカのタイムウィンドウ終了時刻以前，つまりワーカ d がタスク t を最後に実行するタスクとする場合 $a_t + p_t \leq \ell_d$ という関係が成り立っていなければならない．ワーカ d がタスク t を最初もしくは最後に実行するタスクとする場合のみそれぞれ成立していれば良いので，式 (6)(7) は各ワーカのタイムウィンドウ制約である．ただし $-a_t + e_d + r_{d0t} \leq e_d + r_{d0t} \leq \ell_d - r_{d0t}$ より $M_2 = \ell_d + r_{d0t}$ でよい．同様に $a_t - \ell_d + p_t \leq a_t + p_t \leq \ell_t + p_t$ より $M_3 = \ell_t + p_t$ でよい．各タスクに対してタスクの処理終了時刻，つまり配達される時刻がタイムウィンドウ内でなければならないので，タスク t に対して $e_t \leq a_t + p_t \leq \ell_t$ という関係が成り立っていなければならない．いずれかのワーカによってタスクが実行される時のみ成立していれば良いので，式 (8)(9) は各タスクのタイムウィンドウ制約である．

4.3 ワーカが拒否権を有するオフラインタスク割当問題

現実の状況では，ワーカは報酬の最大化やタスク処理の効率性を考えるため，全てのタスクを無条件に受理する訳ではない．したがって，ワーカがタスクに対して選好タイプを持ち，割当てられるタスクに対して拒否権を有している場合のオフラインタスク割当問題 (ROTAP) を考える．ワーカがタスクを受理するかどうかを表す定数として 0-1 定数 $q^d_{\tilde{t}t}$ を以下に定義する．

$$q^d_{\tilde{t}t} = \begin{cases} 1 & \text{ワーカ } d \text{ が} \tilde{t} \text{の直後に割当てられたタスク } t \text{ を受け入れる} \\ 0 & \text{otherwise.} \end{cases}$$

よって目的関数は

$$\sum_{t\in T}\{f_t \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T} q^d_{\tilde{t}t}x^d_{\tilde{t}t} + \gamma(1 - \sum_{d\in D}\sum_{\tilde{t}\in\{0\}\cup T} q^d_{\tilde{t}t}x^d_{\tilde{t}t})\} \quad (12)$$

となる．他の制約は (OTAP) と同様である．よってワーカが拒否権を有する場合のオフラインタスク割当問題 (ROTAP) は目的関数を式 (12)，制約式として式 (1) から式 (11) をもつ問題となる．

4.4 ワーカのタイプ分け

ワーカ d がタスク t を割当てられた時，その割当てを受けるかどうかのタイプ分けを行う．本研究では次の 3 種類のタイプを考える．

タイプ 1 $\frac{\text{前処理時間}}{\text{処理時間}} = \frac{r_{d\tilde{t}t}}{p_t} \leq 1 \Rightarrow q^d_{\tilde{t}t} = 1$：労働力に対する報酬が相対的に高い方が良い

タイプ 2 報酬 $f_t \geq F \Rightarrow q^d_{\tilde{t}t} = 1$：報酬が比較的高いものだけやりたい ($F$ = 全タスクの平均報酬)

タイプ 3 前処理時間 $r_{d\tilde{t}t} \leq R \Rightarrow q^d_{\tilde{t}t} = 1$：無報酬なことはあまりしたくない ($R$ = 全前処理時間の平均)

オフラインの場合 $q^d_{\tilde{t}t}$ は入力として与える 0-1 定数である．しかし現実の状況に近いオンラインの場合，サーバはワーカのタイプは取得できるが $q^d_{\tilde{t}t}$ は未知である．つまりサーバは F や R などのタスク側が決定する閾値を知らないものとする．

4.5　サーバのオンラインタスク割当問題

タスク t がサーバに到着する時刻を $\alpha_t(< e_t)$，ワーカ d がサーバに到着する時刻を $\alpha_d(< e_d)$ とする．タスク t が遅くとも割当てされなければならない時刻を $\ell'_t = \ell_t - p_t - \max\{r_{d\tilde{t}t}|^{\forall}d \in AD, {}^{\forall}\tilde{t} \in T\}$ とする．ただし $AD = \emptyset$ の時，$r_{d\tilde{t}t}$ は 0 とする．この時刻までなら，タスク t をタスク t が実行可能などのワーカにも割当て可能である．また直前にタスク $\tilde{t} \in \{0\} \cup T$ を行ったワーカ d に最後のタスクを割当て可能な時刻を $\ell'_d = \min\{\ell_d - p_t - r_{d\tilde{t}t}|^{\forall}t \in AT\}$ とする．この時刻までならばワーカ d は実行可能などのタスクも割当て可能である．

サーバはそれぞれ動的に到着するタスクとワーカに対して二部グラフの両側オンラインマッチング問題を解く．既にサーバに到着していて，ワーカが未決定のマッチング対象のタスクの集合を $AT(\subseteq T)$，タスクが未決定のマッチング対象のワーカの集合を $AD(\subseteq D)$ とする．時間ステップは表 2 の 4 つに分類する．オフラインの場合の報酬コストと配達失敗コストに加えて，ワーカがサーバに割当てられたタスクを拒否したときコスト tc が発生する．割当てたタスクをワーカが拒否した場合、このタスクはマッチング対象のまま再び別のワーカに割当てされるのを待つ．サーバ全体のコストを最小化するためにこのコスト tc を発生させないようなタスクの割当て，つまりワーカにタスクを拒否されない割当てを考えることが重要になる．

表 2　時間ステップ

α_t	タスク t がサーバに到着する時刻
α_d	ワーカ d がサーバに到着する時刻
ℓ'_d	ワーカ d に最後のタスクを割当てられる時刻
ℓ'_t	タスク t が遅くとも割当てされなければならない時刻

5　提案手法

本章ではワーカタイプを考慮したオンラインタスク割当問題に対するオンラインアルゴリズムを提案する．

5.1　オンラインタスク割当アルゴリズム

ワーカ d が自身の現在地を出発可能な時刻は，サーバに到着する時刻 α_d が既にワーカ d のタイムウィンドウ内であれば α_d，そうでなければ e_d 以降である．よってワーカ d が現在地を出発可能な時刻は $\max\{\alpha_d, e_d\}$ であり，ワーカ d がタスク t を実行し終える時刻は $\max\{\alpha_d, e_d\} + r_{d\tilde{t}t} + p_t$ 以降である．ワーカ d がタスク t を実行する時，ワーカ d がタスク t を実行し終える時刻がタスク t のタイムウィンドウ内に入ってなければならないので $e_t \leq \max\{\alpha_d, e_d\} + r_{d\tilde{t}t} + p_t \leq \ell_t$ でなければならない．また，ワーカ d がタスク t を実行し終える時刻は，ワーカ d 自身のタイムウィンドウ内にも入ってなければならないので $\max\{\alpha_d, e_d\} + r_{d\tilde{t}t} + p_t \leq \ell_d$ でなければならない．

よって，あるタスク t をあるワーカ d が実行可能であるかどうかの判定条件として

$$e_t \leq \max\{\alpha_d, e_d\} + r_{d\tilde{t}t} + p_t \leq \ell_t$$

かつ

$$\max\{\alpha_d, e_d\} + r_{d\tilde{t}t} + p_t \leq \ell_d$$

を満たすかどうかをみる．

サーバにそれぞれ動的に到着するワーカとタスクをマッチングさせるオンラインアルゴリズムとして Algorithm2 : FIFO と Algorithm3 : ワーカの好みに基づく順位付けアルゴリズムの 2 つを次に示す．

Algorithm 2 FIFO

(1) 現在時刻 $= \alpha_d$:

if 割当て可能な t が存在 **then** d に t を割当て，d の情報を更新

else $AD = AD \cup \{d\}$

(2) 現在時刻 $= \alpha_t$:

if 割当て可能な d が存在 **then** t を d に割当て，d の情報を更新

else $AT = AT \cup \{t\}$

(3) 現在時刻 $= \ell'_d$:

$AD = AD \backslash \{d\}$

(4) 現在時刻 $= \ell'_t$:

$AT = AT \backslash \{t\}$

実行可能な割当てが複数存在する場合は先に到着しているものを優先することとする．ワーカ d の情報更新では，d が再びサーバに到着する時刻を，割当てられたタスク t を実行し終える時刻 $a_t + p_t$ とし，d の新たな出発地点 o_d を，タスク t の配達先 l_{d_t} とする．図 2 にワーカ d がタスク t を行う際の，ワーカの位置と時刻との関係及びワーカ d の情報更新について示す．横軸は時刻を表し，ワーカ d が自身の初期位置からタスク t の店，店から配達先までの移動を矢印で表している．配達先に到着する時刻がワーカ d のタイムウィンドウの終了時刻以前の場合，ワーカ d は新たなタスクを割当てられるために再び割当て対象となる．サーバに再び到着する時刻はタスク t の配達先に到着した時刻となり，ワーカ d の初期位置はタスク t の配達先となる．

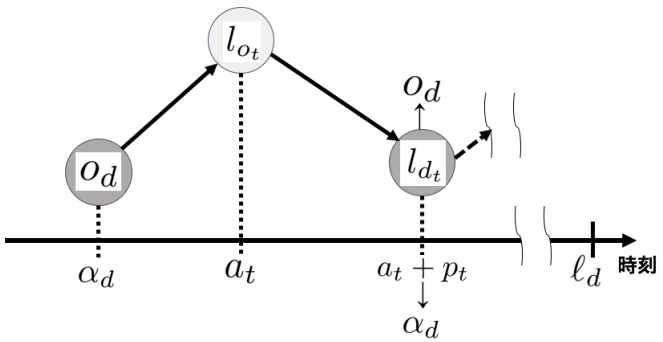


図 2　ワーカ d がタスク t を行う際の位置と時刻の関係

次に，サーバがワーカのタイプを取得可能な場合を考える．但しワーカの各閾値についてはサーバは未知である．$AT_d(\subseteq AT)$ を AT 内のワーカ d が実行可能なタスクの集合とする．

Algorithm 3 ワーカの好みに基づく順位付けアルゴリズム

(1) 現在時刻 $= \alpha_d$:

AT_d 内のタスクをワーカ d のタイプで順位付け

$AD = AD \cup \{d\}$

(2) 現在時刻 $= \alpha_t$:

if t を含む AT_d が存在 **then** t を含む AT_d 内のタスクの順位付けを変更

$AT = AT \cup \{t\}$

(3) 現在時刻 $= \ell'_d$:

if $AT_d \neq \emptyset$ **then** 最上位の t を割当て

$AD = AD \backslash \{d\}$

(4) 現在時刻 $= \ell'_t$:

if t を含む AT_d が存在しない **then** $AT = AT \backslash \{t\}$

else t を最上位にしている d に割当て

$AD = AD \backslash \{d\}$，$AT = AT \backslash \{t\}$，d の情報を更新

ワーカ d の情報更新はアルゴリズム 2:FIFO と同様である．タスク割当てを行うのが時刻 ℓ'_d, ℓ'_t であるのは，各ワーカに対してできるだけ多くの実行可能なタスクの順位付けを行い，ワーカに拒否されないタスクの割当てを行うことを目的としているためである．

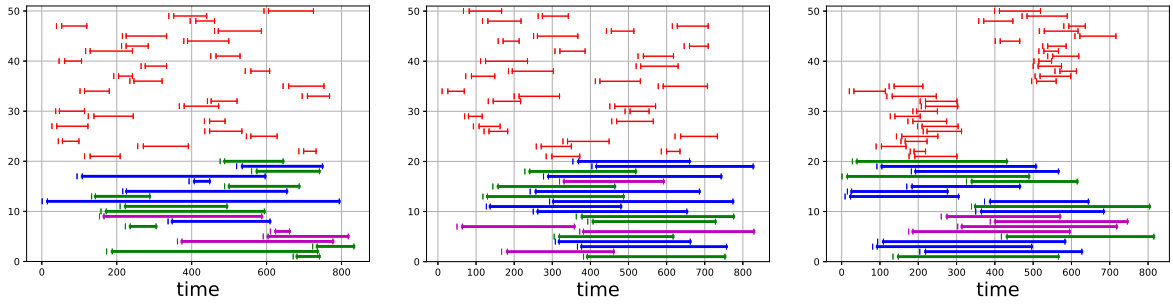
6　計算機実験

本章では提案したオンラインアルゴリズム Algorithm2 : FIFO と Algorithm3 : ワーカの好みに基づく順位付けアルゴリズムに対して行なった計算機実験について及びその実験結果について述べる．

6.1　インスタンスの生成

10×10 マス，1 マス 2000m の範囲の格子上にワーカの出発地点，タスクの店の場所及び配達先を配置する．ワーカの移動速度は分速 250m とした．ワーカとタスクのタイムウィンドウはそれぞれ 9 時から 23 時の間に収まるようにし，次の 4 種類のインスタンスの生成を行う．タスクの店の場所及び配達先はインスタンスタイプ 1,2,3 については一様ランダムに配置する．

- インスタンスタイプ 1:完全ランダム
  - ワーカのタイムウィンドウ:2 つの時刻をランダムに生成，その大小関係から決定
  - タスクのタイムウィンドウ:開始時刻と時間間隔をそれぞれランダムに生成，タイムウィンドウの終了時刻は開始時刻から時間間隔を足したもの
- インスタンスタイプ 2:ワーカが比較的長い時間働く
  - ワーカのタイムウィンドウ:ワーカのタイムウィンドウ幅が 4 時間から 8 時間となるように設定
  - タスクのタイムウィンドウ:インスタンスタイプ 1 と同様
- インスタンスタイプ 3:タスクの発生時間にピークとオフピークがある
  - ワーカのタイムウィンドウ:インスタンスタイプ 2 と同様
  - タスクのタイムウィンドウ:昼と夜の時間に多くタスクが発生するように設定
- インスタンスタイプ 4:タスクの発生位置に偏りを持たせる
  - ワーカのタイムウィンドウ:インスタンスタイプ 2 と同様
  - タスクのタイムウィンドウ:インスタンスタイプ 3 と同様
  - タスクの店の位置:中心 (5,5) 付近で多く発生するように生成
  - タスクの配達先の位置:一様ランダムに生成

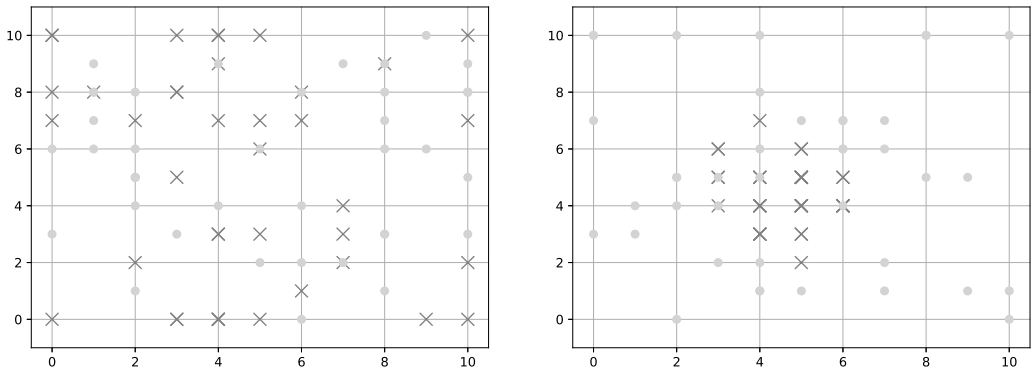


[1] インスタンスタイプ 1　　[2] インスタンスタイプ 2　　[3] インスタンスタイプ 3

図 3　$|D| = 20, |T| = 30$ のインスタンス例

図 3 にインスタンス例としてワーカ数 20，タスク数 30 のインスタンスタイプ 1,2,3 におけるワーカとタスクのそれぞれのタイムウィンドウを示す．下側にワーカ，上側にタスクのそれぞれのタイムウィンドウをプロットしている．

図 4 に一様ランダムにタスクの店と配達先の位置を生成した場合の例と，中心 (5,5) 付近に店の位置を多く設定した場合例を示す．タスク数は 40 である．× を店，○ を配達先としてプロットした．タスクによって位置が同一の場合は同一プロットになっている為，タスク数に比べてプロット数が少なくなっている．



[1] 一様ランダムに生成　　[2] 中心に店の位置を多く設定

図 4　店の配置例

実験環境は表 3 の通りである．配達失敗コスト γ は各インスタンスにおいてタスクの実行報酬の最大値に 100 を足したものとする．

表 3　実験環境

CPLEX	12.9.0.0
プロセッサ	3.5GHz Intel Core i7
メモリ	16GB 1600MHz DDR3
使用言語	C++(clang-902.0.39.2)

6.2　実験結果

インスタンスタイプ 1，2，3，4 それぞれに対するワーカが拒否権を持たない場合 (OTAP) と持つ場合 (ROTAP) のオフラインタスク割当問題の実験結果をそれぞれ表 4，5，6，7 に示す．ワーカの数とタスクの数を変化させてそれぞれ異なる 10 個のインスタンスに対する計算時間及び割当率の平均をとっている．計算時間は 3600 秒で打ち切りとしている．目的関数値増加率は OTAP の目的関数値に対する ROTAP の目的関数値の増加率である．また，インスタンスタイプ 1，2，3，4 それぞれに対して，ワーカが拒否権を持たない場合に対して，Algorithm2 : FIFO と Algorithm3 : ワーカの好みに基づく順位付けアルゴリズム (RANK) を適用した場合のそれぞれの結果を表 9，10，11，12 に示す．ワーカが拒否権を持つ場合の結果を表 13，14，15，16 に示す．ワーカが拒否権を有する場合の割当率及び拒否率の平均を図 5，6，7，8，9，10 に示す．

表 4　インスタンスタイプ 1 に対するオフラインタスク割当問題

		OTAP		ROTAP		
$\|D\|$	$\|T\|$	計算時間	割当率	計算時間	割当率	目的関数値増加率
10	10	0.06(sec)	92.0%	0.06(sec)	85.0%	16.7%
	20	0.26(sec)	87.0%	0.24(sec)	82.0%	7.8%
	30	1.56(sec)	77.0%	1.01(sec)	71.3%	9.6%
20	10	0.12(sec)	96.0%	0.11(sec)	90.0%	16.5%
	20	0.65(sec)	93.5%	0.64(sec)	91.0%	7.3%
	30	2.53(sec)	91.0%	2.91(sec)	88.7%	4.2%
	40	7.90(sec)	87.0%	7.49(sec)	83.3%	9.1%
30	10	0.18(sec)	94.0%	0.18(sec)	91.0%	8.6%
	20	1.09(sec)	92.0%	1.14(sec)	91.5%	4.1%
	30	5.31(sec)	90.7%	5.16(sec)	90.3%	5.3%
	40	15.02(sec)	91.3%	16.08(sec)	90.0%	2.6%

表 5　インスタンスタイプ 2 に対するオフラインタスク割当問題

		OTAP		ROTAP		
$\|D\|$	$\|T\|$	計算時間	割当率	計算時間	割当率	目的関数値増加率
10	10	0.06(sec)	87.0%	0.05(sec)	84.0%	12.3%
	20	0.23(sec)	82.5%	0.22(sec)	79.5%	4.8%
	30	1.05(sec)	82.0%	0.78(sec)	78.7%	5.6%
20	10	0.10(sec)	90.0%	0.10(sec)	88.0%	5.1%
	20	0.61(sec)	93.5%	0.59(sec)	89.5%	8.8%
	30	2.35(sec)	86.7%	2.34(sec)	84.0%	5.5%
	40	29.30(sec)	88.8%	9.88(sec)	86.3%	6.7%
30	10	0.19(sec)	91.0%	0.18(sec)	86.0%	6.0%
	20	1.12(sec)	95.5%	1.15(sec)	94.0%	3.6%
	30	5.87(sec)	93.3%	70.34(sec)	90.4%	6.1%
	40	18.31(sec)	93.0%	14.89(sec)	91.5%	3.3%

表 6　インスタンスタイプ 3 に対するオフラインタスク割当問題

		OTAP		ROTAP		
$\|D\|$	$\|T\|$	計算時間	割当率	計算時間	割当率	目的関数値増加率
10	10	0.05(sec)	72.0%	0.05(sec)	59.0%	22.8%
	20	0.23(sec)	69.5%	0.21(sec)	65.5%	5.3%
	30	6.05(sec)	74.3%	3.20(sec)	71.7%	4.3%
20	10	0.10(sec)	92.0%	0.10(sec)	87.0%	8.0%
	20	0.66(sec)	93.5%	0.66(sec)	92.0%	3.7%
	30	2.77(sec)	92.3%	2.50(sec)	90.0%	4.5%
	40	48.90(sec)	86.5%	8.25(sec)	84.5%	4.2%
30	10	0.20(sec)	96.0%	0.20(sec)	93.0%	2.7%
	20	1.16(sec)	93.0%	1.25(sec)	90.0%	5.2%
	30	5.52(sec)	91.3%	5.34(sec)	90.3%	2.7%
	40	58.12(sec)	90.8%	318.78(sec)	88.0%	5.1%

表 7　インスタンスタイプ 4 に対するオフラインタスク割当問題

		OTAP		ROTAP		
$\|D\|$	$\|T\|$	計算時間	割当率	計算時間	割当率	目的関数値増加率
10	10	0.06(sec)	92.0%	0.06(sec)	86.0%	13.3%
	20	2.18(sec)	86.5%	3.75(sec)	82.5%	12.8%
	30	49.28(sec)	82.0%	12.50(sec)	79.0%	6.9%
20	10	0.12(sec)	97.0%	0.12(sec)	95.0%	4.1%
	20	0.78(sec)	97.5%	1.36(sec)	96.0%	3.5%
	30	22.48(sec)	91.7%	23.97(sec)	90.0%	5.6%
	40	-(sec)	-%	-(sec)	-%	-%
30	10	0.22(sec)	97.0%	0.19(sec)	96.0%	4.1%
	20	1.88(sec)	97.5%	2.93(sec)	96.0%	2.0%
	30	-(sec)	-%	-(sec)	-%	-%
	40	-(sec)	-%	-(sec)	-%	-%

表 4,5,6,7 より，インスタンスタイプ 1 から 4 までワーカ数 $|D|$, タスク数 $|T|$ に関わらず OTAP に比べて ROTAP の方が計算時間は短く割当率が低い結果となっている．これは OTAP に比べて ROTAP の方が実行可能なワーカとタスクの組み合わせが少ないのが理由として挙げられる．また $|D|, |T|$ の増加とともに計算時間は長くなり割当率は低くなる傾向もある．

インスタンスタイプ 4 は $|D| = 20, |T| = 40$，$|D| = 30, |T| = 30$，$|D| = 30, |T| = 40$ のそれぞれの場合に時間制限 3600 秒以内に 10 個のインスタンスの内，最適解を求めることができなかったものがあるため，表 7 において結果を省略している．最適解が求まったインスタンスのみで平均を取った結果を表 8 に示す．このことから，駅周辺などの繁華街に集中している店からその周りに位置する配達先に配達を行うような，想像し得る現実的なケースは最適なワーカとタスクの組み合わせを求めるのは，そうでないケースより難しいことがわかる．

表 8　インスタンスタイプ 4 に対する最適解が求まったインスタンスのみの結果

$\|D\|$	$\|T\|$	OTAP			ROTAP		
		最適解が求まったインスタンス数	計算時間	割当率	最適解が求まったインスタンス数	計算時間	割当率
20	40	7	227.81(sec)	95.4%	7	302.36(sec)	93.9%
30	30	6	11.65(sec)	95.0%	9	32.63(sec)	93.3%
	40	4	34.30(sec)	97.5%	4	107.39(sec)	93.8%

次に，ワーカが拒否権を持たない場合にインスタンスタイプ 1 から 4 に対して提案オンラインアルゴリズム Algorithm2 : FIFO と Algorithm3 : RANK を適用させた時の結果を示す．

表 9　インスタンスタイプ 1 に対するオンラインアルゴリズムの比較

		OTAP	FIFO		RANK	
$\|D\|$	$\|T\|$	割当率	競合比	割当率	競合比	割当率
10	10	92.0%	0.65	54.0%	0.59	46.0%
	20	87.0%	0.68	53.5%	0.61	43.5%
	30	77.0%	0.75	51.7%	0.64	36.7%
20	10	96.0%	0.80	78.0%	0.72	68.0%
	20	93.5%	0.73	71.0%	0.66	64.5%
	30	91.0%	0.79	73.7%	0.68	63.0%
	40	87.0%	0.78	67.3%	0.67	55.5%
30	10	94.0%	0.81	77.0%	0.81	77.0%
	20	92.0%	0.77	71.0%	0.74	70.0%
	30	90.7%	0.84	77.7%	0.76	70.0%
	40	91.3%	0.80	74.3%	0.71	66.3%

表 10　インスタンスタイプ 2 に対するオンラインアルゴリズムの比較

		OTAP	FIFO		RANK	
$\|D\|$	$\|T\|$	割当率	競合比	割当率	競合比	割当率
10	10	87.0%	0.72	54.0%	0.69	53.0%
	20	82.5%	0.79	62.5%	0.69	51.0%
	30	82.0%	0.80	63.0%	0.67	49.0%
20	10	90.0%	0.81	70.0%	0.79	68.0%
	20	93.5%	0.79	74.5%	0.75	71.5%
	30	86.7%	0.88	76.0%	0.78	67.7%
	40	88.8%	0.85	76.3%	0.72	64.0%
30	10	91.0%	0.88	77.0%	0.86	75.0%
	20	95.5%	0.82	81.0%	0.78	78.0%
	30	93.3%	0.86	82.0%	0.77	74.0%
	40	93.0%	0.86	81.8%	0.76	73.5%

表 11　インスタンスタイプ 3 に対するオンラインアルゴリズムの比較

		OTAP	FIFO		RANK	
$\|D\|$	$\|T\|$	割当率	競合比	割当率	競合比	割当率
10	10	72.0%	0.83	53.0%	0.78	51.0%
	20	69.5%	0.83	52.0%	0.73	42.0%
	30	74.3%	0.79	53.0%	0.64	35.0%
20	10	92.0%	0.76	66.0%	0.76	65.0%
	20	93.5%	0.82	77.5%	0.72	70.0%
	30	92.3%	0.81	76.0%	0.65	62.7%
	40	86.5%	0.82	70.5%	0.68	55.5%
30	10	96.0%	0.89	86.0%	0.86	85.0%
	20	93.0%	0.92	85.0%	0.85	80.0%
	30	91.3%	0.89	81.7%	0.78	74.3%
	40	90.8%	0.86	79.3%	0.73	69.0%

表 12　インスタンスタイプ 4 に対するオンラインアルゴリズムの比較

		OTAP	FIFO		RANK	
$\|D\|$	$\|T\|$	割当率	競合比	割当率	競合比	割当率
10	10	92.0%	0.73	63.0%	0.62	49.0%
	20	86.5%	0.69	55.0%	0.60	45.0%
	30	82.0%	0.71	54.0%	0.63	43.3%
20	10	97.0%	0.83	80.0%	0.77	74.0%
	20	97.5%	0.81	79.5%	0.73	74.0%
	30	91.7%	0.82	76.0%	0.73	67.3%
	40	-%	-	76.0%	-	62.8%
30	10	97.0%	0.95	92.0%	0.92	90.0%
	20	97.5%	0.91	89.5%	0.82	83.5%
	30	-%	-	87.0%	-	77.7%
	40	-%	-	81.8%	-	69.5%

表 9,10,11,12 より，ワーカが拒否権を持たない場合，Algorithm2 : FIFO の方が Algorithm3 : RANK よりも割当率が同等もしくは高い結果となっている．これはアルゴリズムの性質上，Algorithm2 : FIFO の方が Algorithm3 : RANK よりもワーカ 1 人に対してタスク割当を行う回数が多いためであると考えられる．競合比を見てみると，Algorithm2 : FIFO はインスタンスタイプ 4 において $|D| = 30, |T| = 10$ の時に最大の 0.95，インスタンスタイプ 1 において $|D| = 10, |T| = 10$ の時に最小の 0.65 という結果になっている．Algorithm3 : RANK でもインスタンスタイプ 4 において $|D| = 30, |T| = 10$ の時に最大の 0.92，インスタンスタイプ 1 において $|D| = 10, |T| = 10$ の時に最小の 0.59 という結果になっている．

次にワーカが拒否権を有する場合，インスタンスタイプ 1 から 4 に対して提案オンラインアルゴリズム Algorithm2 : FIFO と Algorithm3 : RANK を適用させた時の結果を示す．拒否率とはサーバがワーカにタスク割当を行なった回数に対して拒否された回数の比である．

表 13　ワーカが拒否権を有する場合のインスタンスタイプ 1 に対するオンラインアルゴリズムの比較

		OTAP	ROTAP	FIFO			RANK		
$\|D\|$	$\|T\|$	割当率	割当率	競合比	割当率	拒否率	競合比	割当率	拒否率
10	10	92.0%	85.0%	0.48	21.0%	58.7%	0.47	23.0%	57.4%
	20	87.0%	82.0%	0.47	17.0%	72.4%	0.55	33.0%	38.3%
	30	77.0%	71.7%	0.49	10.7%	80.2%	0.57	24.3%	35.6%
20	10	96.0%	89.0%	0.43	28.0%	72.2%	0.54	49.0%	42.3%
	20	93.5%	90.5%	0.40	21.0%	74.3%	0.48	37.5%	44.2%
	30	91.0%	89.0%	0.43	24.3%	72.5%	0.54	41.7%	35.4%
	40	87.0%	81.8%	0.47	23.3%	70.5%	0.53	34.3%	42.8%
30	10	94.0%	89.0%	0.44	32.0%	69.2%	0.57	51.0%	44.2%
	20	92.0%	90.5%	0.45	29.5%	69.3%	0.55	46.0%	39.9%
	30	90.7%	88.7%	0.44	27.3%	72.2%	0.56	45.0%	36.7%
	40	91.3%	89.8%	0.44	25.5%	72.1%	0.56	45.5%	37.0%

表 14　ワーカが拒否権を有する場合のインスタンスタイプ 2 に対するオンラインアルゴリズムの比較

		OTAP	ROTAP	FIFO			RANK		
$\|D\|$	$\|T\|$	割当率	割当率	競合比	割当率	拒否率	競合比	割当率	拒否率
10	10	87.0%	84.0%	0.48	18.0%	78.0%	0.52	27.0%	52.4%
	20	82.5%	79.5%	0.47	14.5%	78.0%	0.57	32.0%	41.5%
	30	82.0%	78.7%	0.49	17.3%	74.1%	0.56	31.3%	42.0%
20	10	90.0%	88.0%	0.51	31.0%	64.0%	0.56	40.0%	44.3%
	20	93.5%	89.5%	0.43	24.0%	73.7%	0.55	45.5%	38.8%
	30	86.7%	84.0%	0.47	24.0%	72.5%	0.58	39.7%	38.4%
	40	88.8%	86.3%	0.46	26.5%	73.1%	0.59	47.8%	32.8%
30	10	91.0%	86.0%	0.53	36.0%	61.8%	0.59	46.0%	45.6%
	20	95.5%	94.0%	0.41	29.0%	74.2%	0.55	52.0%	38.0%
	30	88.7%	92.0%	0.48	26.7%	72.1%	0.60	47.7%	39.1%
	40	93.0%	91.5%	0.43	29.5%	73.2%	0.55	47.8%	35.8%

表 15　ワーカが拒否権を有する場合のインスタンスタイプ 3 に対するオンラインアルゴリズムの比較

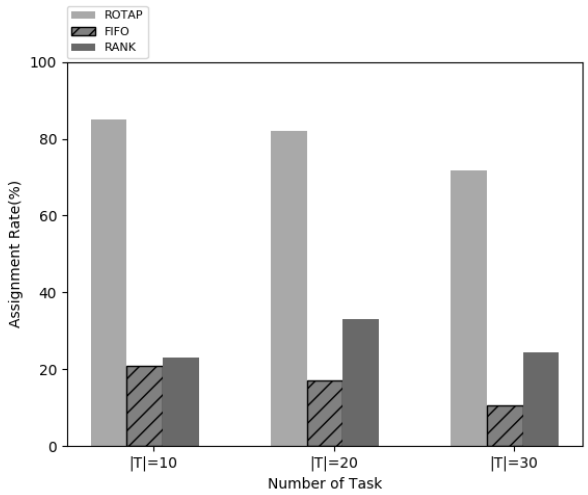
		OTAP	ROTAP	FIFO			RANK		
$\|D\|$	$\|T\|$	割当率	割当率	競合比	割当率	拒否率	競合比	割当率	拒否率
10	10	72.0%	59.0%	0.54	16.0%	81.0%	0.61	34.0%	49.2%
	20	69.5%	65.5%	0.58	17.0%	71.1%	0.65	31.0%	40.0%
	30	74.3%	71.7%	0.51	11.3%	81.7%	0.60	27.3%	36.1%
20	10	92.0%	87.0%	0.47	25.0%	72.2%	0.56	44.0%	47.9%
	20	93.5%	92.0%	0.40	20.5%	78.7%	0.53	46.0%	38.8%
	30	92.3%	90.0%	0.41	21.3%	75.1%	0.53	44.0%	35.2%
	40	86.5%	84.5%	0.45	21.8%	76.5%	0.55	38.5%	40.6%
30	10	96.0%	93.0%	0.50	44.0%	59.8%	0.62	61.0%	39.0%
	20	93.0%	90.0%	0.44	22.5%	77.5%	0.58	50.0%	42.8%
	30	93.3%	90.4%	0.43	27.0%	72.8%	0.55	49.3%	38.5%
	40	90.8%	88.0%	0.42	25.5%	75.4%	0.56	50.5%	34.6%

表 16　ワーカが拒否権を有する場合のインスタンスタイプ 4 に対するオンラインアルゴリズムの比較

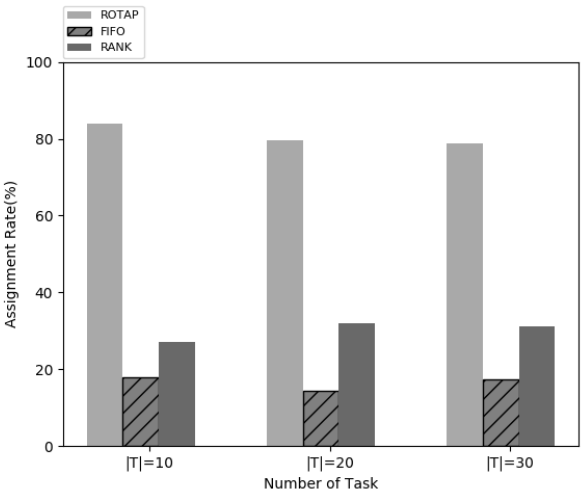
		OTAP	ROTAP	FIFO			RANK		
$\|D\|$	$\|T\|$	割当率	割当率	競合比	割当率	拒否率	競合比	割当率	拒否率
10	10	92.0%	86.0%	0.44	12.0%	80.0%	0.49	26.0%	53.6%
	20	86.5%	82.5%	0.44	15.5%	74.8%	0.51	31.0%	45.0%
	30	82.0%	79.0%	0.44	13.0%	80.6%	0.49	24.0%	58.2%
20	10	97.0%	95.0%	0.41	22.0%	76.3%	0.51	45.0%	48.5%
	20	97.5%	96.0%	0.39	25.0%	73.7%	0.50	45.0%	45.3%
	30	91.7%	90.0%	0.40	20.7%	78.0%	0.51	40.0%	44.9%
	40	-%	-%	-	25.5%	74.0%	-	42.0%	41.1%
30	10	97.0%	96.0%	0.43	37.0%	70.7%	0.58	60.0%	41.8%
	20	97.5%	96.0%	0.40	28.5%	74.9%	0.53	51.0%	43.6%
	30	-%	-%	-	25.0%	79.3%	-	45.7%	48.5%
	40	-%	-%	-	28.0%	71.6%	-	44.8%	41.2%

表 13,14,15,16 より，Algorithm2 : FIFO より Algorithm3 : RANK の方が割当率の平均が高く，拒否率の平均が低い結果となっている．Algorithm2 : FIFO の割当率の平均はインスタンスタイプ 3 に対して $|D| = 10, |T| = 30$ の時に最少となっており、その値は 5% である．インスタンスタイプ 4 に対して $|D| = 30, |T| = 10$ の時に割当率の平均が最大となっているがその値は 30% となっている．一方で Algorithm3 : RANK はインスタンスタイプ 3 に対して $|D| = 30, |T| = 30$ の時に割当率の平均が最少となっているが，その値は Algorithm2 : FIFO の最大値を超えて 32.0% である．競合比を見てみると，Algorithm2 : FIFO ではインスタンスタイプ 3 において $|D| = 10, |T| = 10$ の時に最大の 0.62，インスタンスタイプ 4 において $|D| = 20, |T| = 20$ の時に最小の 0.35 という結果になっている．Algorithm3 : RANK では，インスタンスタイプ 3 において $|D| = 10, |T| = 10$ の時に最大の 0.72，インスタンスタイプ 4 において $|D| = 20, |T| = 20$ の時に最小の 0.48 という結果になっている．

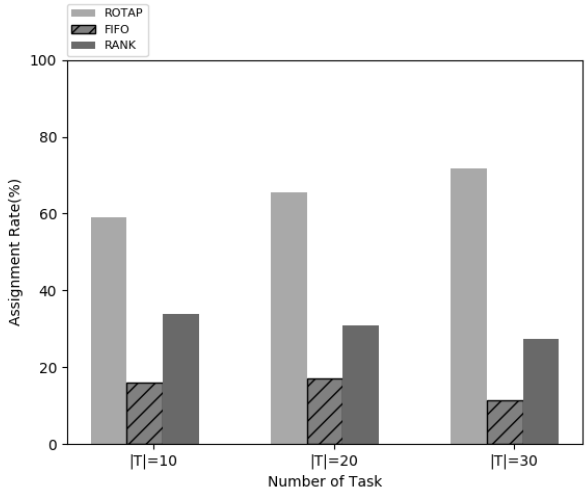
図 5,6,7 にワーカ数毎の割当率の平均を，図 8，9，10 にワーカ数毎の拒否率の平均を棒グラフで表す．



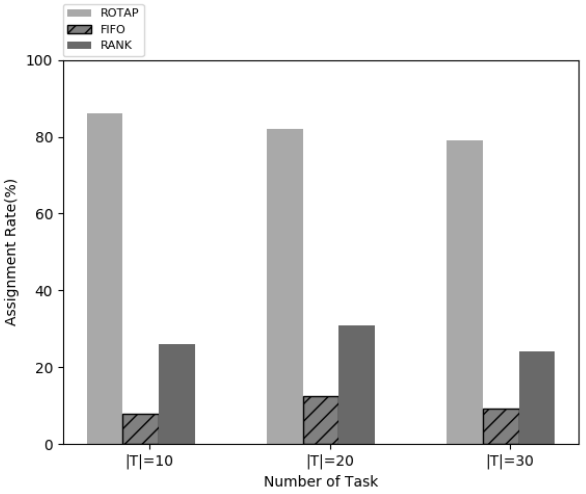
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2

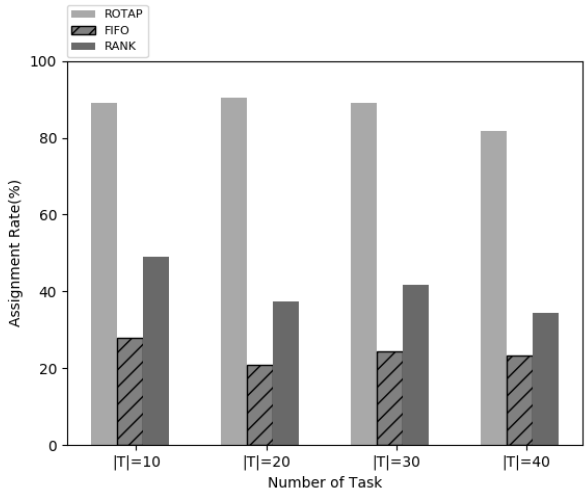


[3] インスタンスタイプ 3

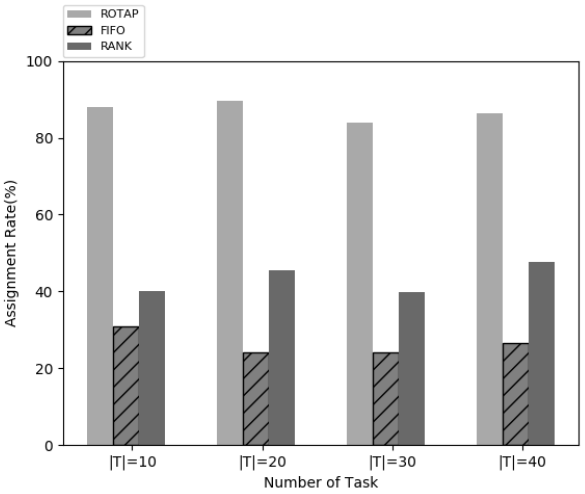


[4] インスタンスタイプ 4

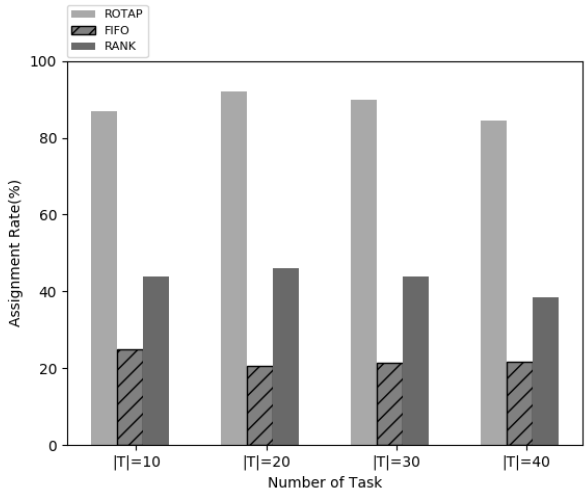
図 5　$|D| = 10$ の時の各インスタンスタイプに対する割当率



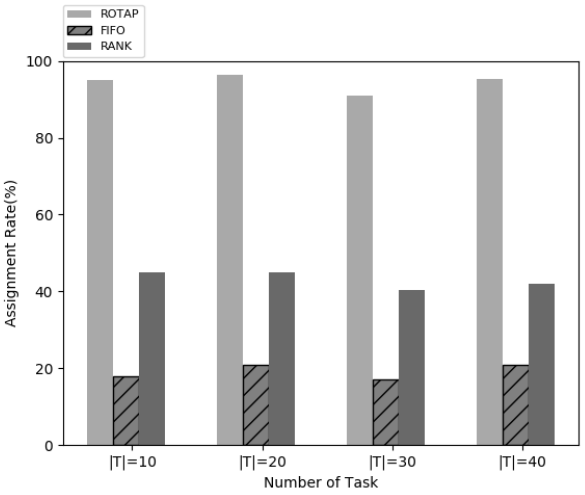
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2

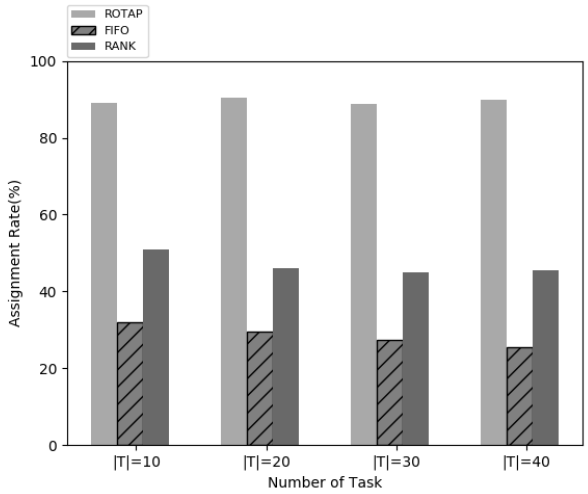


[3] インスタンスタイプ 3

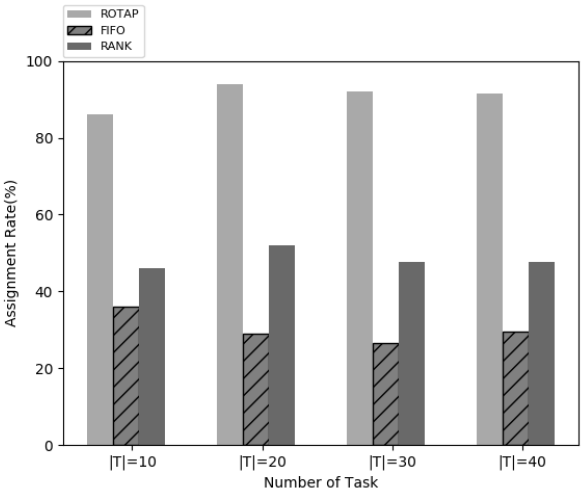


[4] インスタンスタイプ 4

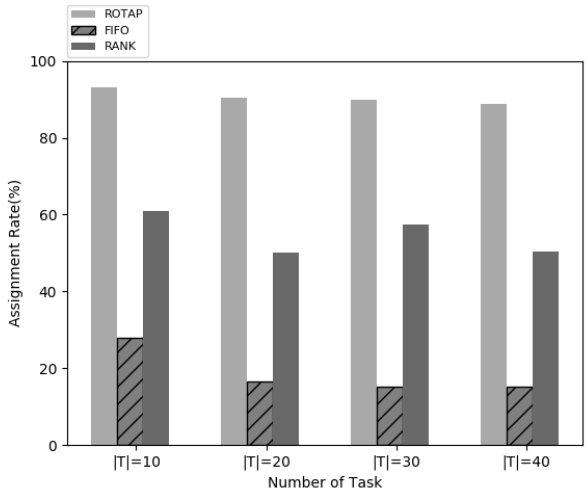
図 6　$|D| = 20$ の時の各インスタンスタイプに対する割当率



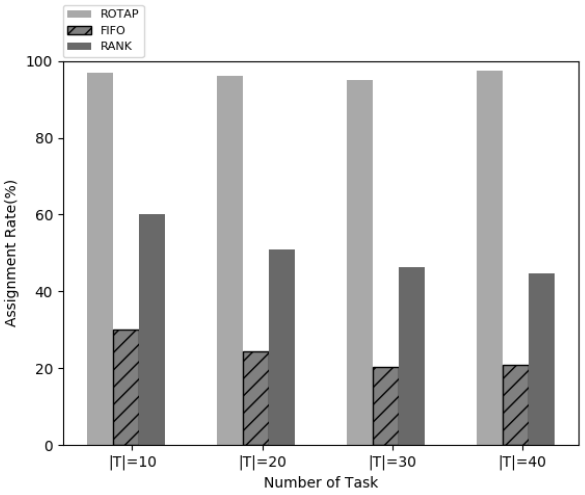
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2

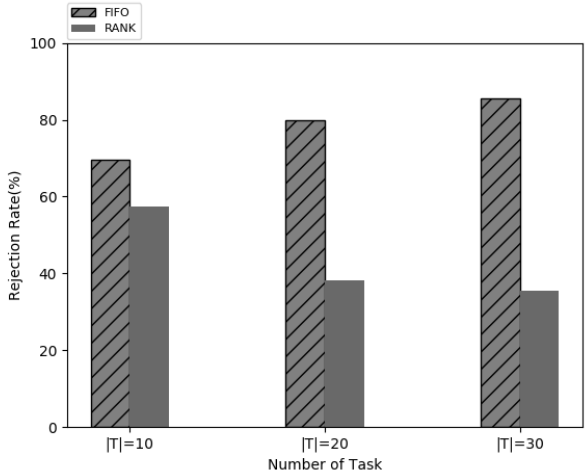


[3] インスタンスタイプ 3

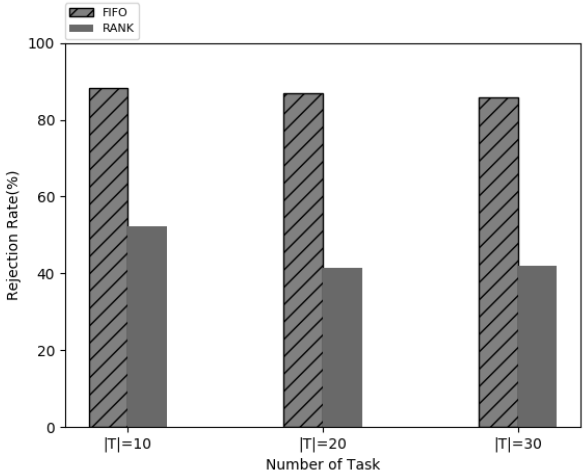


[4] インスタンスタイプ 4

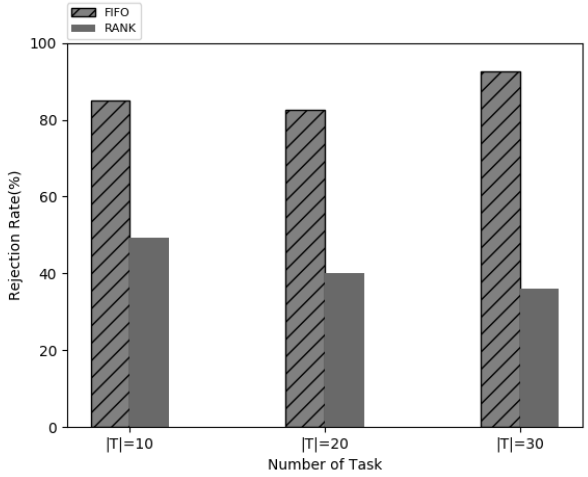
図 7　$|D| = 30$ の時の各インスタンスタイプに対する割当率



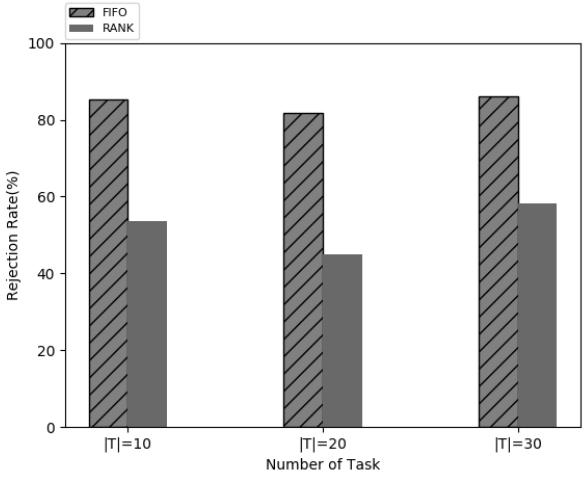
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2

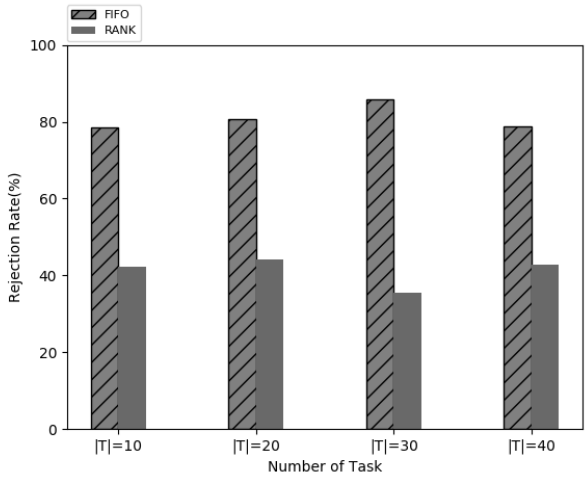


[3] インスタンスタイプ 3

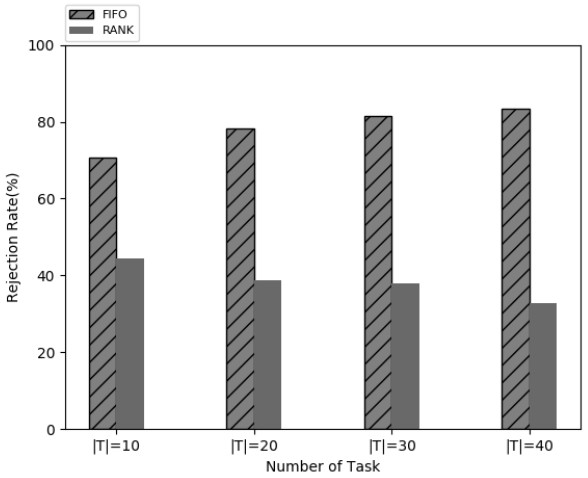


[4] インスタンスタイプ 4

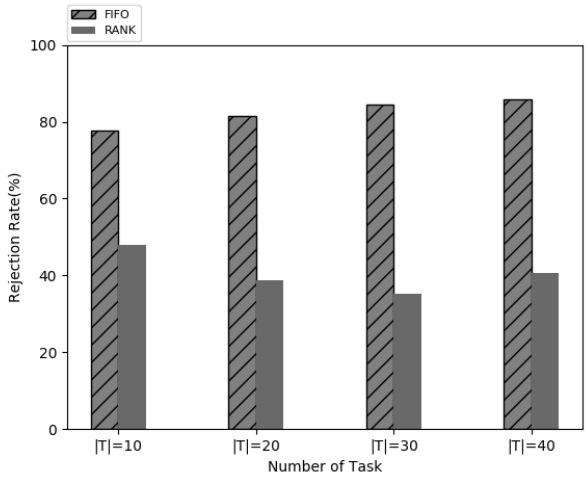
図 8　$|D| = 10$ の時の各インスタンスタイプに対する拒否率



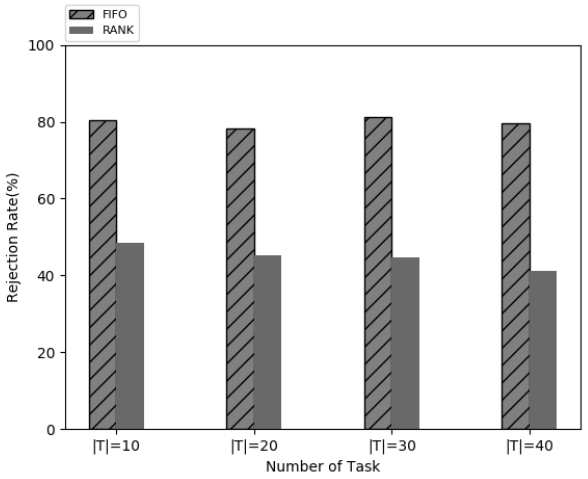
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2

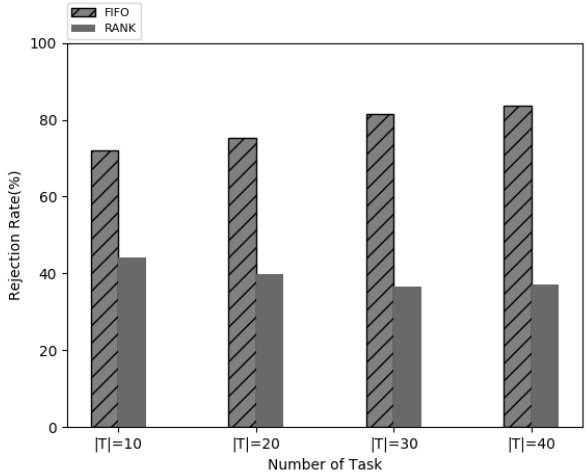


[3] インスタンスタイプ 3

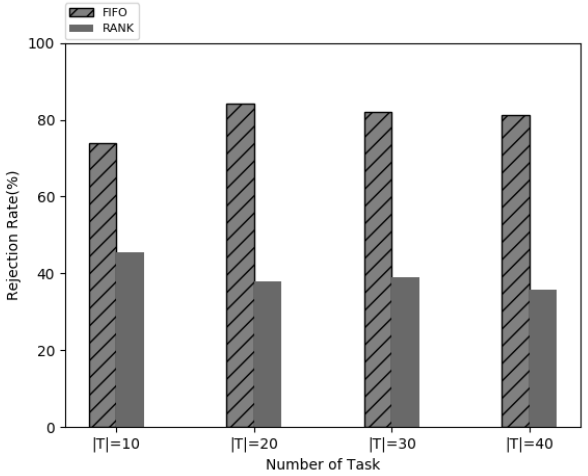


[4] インスタンスタイプ 4

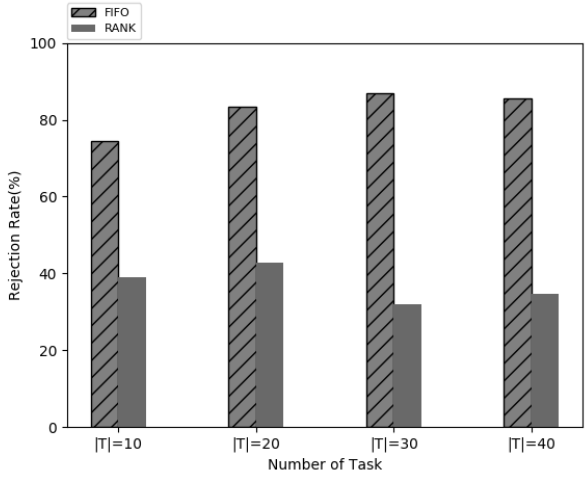
図 9　$|D| = 20$ の時の各インスタンスタイプに対する拒否率



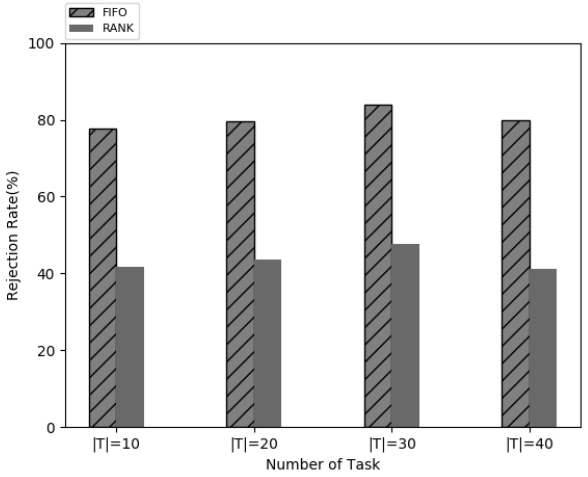
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2



[3] インスタンスタイプ 3



[4] インスタンスタイプ 4

図 10　$|D| = 30$ の時の各インスタンスタイプに対する拒否率

7　経済分析

本章では，6 章で行なった計算機実験の結果を踏まえた経済分析を行う．

7.1　社会的余剰

本研究ではユーザの支払い及びサーバの収益を 0 としている．よって，本研究におけるオフラインの場合の市場全体において，サーバの利益はワーカに支払う報酬と配達失敗コスト和を足し合わせた値にマイナスをかけたもの，ワーカの収入はサーバより支払われる報酬，ユーザの利得は配達失敗コストの和となる．これらの総和より，社会的余剰は-2×配送失敗コストとなる．サーバにおいて生じる全コストの値とワーカの収入の値が等しくなることがサーバにとっての理想ではあるが，全タスクを行えなかったり，ワーカがタスクに対して選好タイプを持ち，サーバが割当てたタスクをワーカが拒否したりすることで，実際にはサーバにおいて生じるコストの値の方が大きくなる．本研究においてサーバの収益が 0 であることに留意して，サーバにおいて生じる全コストからワーカに対して支払う報酬を差し引いた値にマイナスをかけた値がサーバにおける余剰と見ることができる．このサーバにおける余剰が大きいほど，サーバ全体の利益は良いということになる．よって，ワーカがタスクに対して選好タイプを持つ場合とオンラインモデルの場合に余剰がどの程度なのかを見ることにする．

7.2　配送効率性

ワーカの有限集合が D，タスクの有限集合が T のあるインスタンスに対して，OTAP における余剰を $S_{OPT} = -\gamma(1 - \sum_{d \in D} \sum_{\tilde{t} \in \{0\} \cup T} x_{\tilde{t}t}^{d})$ とする．同様のインスタンスに対して，ROTAP における余剰を $S_{ROTAP} = -\gamma(1 - \sum_{d \in D} \sum_{\tilde{t} \in \{0\} \cup T} q_{\tilde{t}t}^{d} x_{\tilde{t}t}^{d})$ とする．同様にオンラインアルゴリズム Algorithm2 : FIFO と Algorithm3 : RANK より得られるサーバ全体のコストから，ワーカへの報酬を引いた値にマイナスをかけた値をそれぞれのオンラインアルゴリズムより得られる余剰 S_F, S_R とする．提案したオンラインアルゴリズム Algorithm2 : FIFO，Algorithm3 : RANK の性能指標として配送効率性，S_{OPT} に対する S_{ROTAP} と S_F，S_R のそれぞれの比を表 17，18，19，20 に示す．

ROTAP より得られる配送効率性はインスタンスタイプ 1 において $|D| = 30, |T| = 20$ のとき最大の 0.917，インスタンスタイプ 4 において $|D| = 10, |T| = 10$ の時には最小の 0.488 となっている．ワーカがタスクに対して選好タイプを持ち，サーバから割当てられ

たタスクに対して選択を行う場合，そうでない場合に比べて配送効率性は明らかに悪くなることがわかる．ここで配送効率性が例え 1.000 であっても，目的関数値が等しいわけではないということに注意する．なぜならば，余剰は実行できなかったタスク数にのみ依存するが，目的関数値は各タスクの報酬にも依存するからである．Algorithm3 : RANK の方が Algorithm2 : FIFO よりも全体的に配送効率性は良いが，その値は ROTAP より得られる値と比べるとかなり低い．Algorithm2 : FIFO の配送効率性はインスタンスタイプ 3 において $|D| = 10, |T| = 20$ の時に最大の 0.315，インスタンスタイプ 4 において $|D| = 20, |T| = 20$ の時に最小の 0.023 となっている．Algorithm3 : RANK の配送効率性はインスタンスタイプ 3 において $|D| = 10, |T| = 20$ の時に最大の 0.407，インスタンスタイプ 4 において $|D| = 20, |T| = 10$ の時に最小の 0.037 となっている．Algorithm3 : RANK より得られる配送効率性の最大値が ROTAP において得られる配送効率性の最小値よりも小さい．

表 17　インスタンスタイプ 1 における配送効率性

$\|D\|$	$\|T\|$	ROTAP(S_{OPT}/S_{ROTAP})	FIFO(S_{OPT}/S_F)	RANK(S_{OPT}/S_R)
10	10	0.710	0.093	0.091
	20	0.678	0.133	0.177
	30	0.799	0.225	0.290
20	10	0.600	0.047	0.067
	20	0.650	0.070	0.092
	30	0.838	0.090	0.140
	40	0.713	0.135	0.178
30	10	0.750	0.067	0.105
	20	0.917	0.086	0.131
	30	0.805	0.096	0.156
	40	0.838	0.090	0.144

表 18　インスタンスタイプ 2 における配送効率性

$\|D\|$	$\|T\|$	ROTAP(S_{OPT}/S_{ROTAP})	FIFO(S_{OPT}/S_F)	RANK(S_{OPT}/S_R)
10	10	0.825	0.128	0.145
	20	0.835	0.172	0.235
	30	0.833	0.179	0.233
20	10	0.900	0.129	0.152
	20	0.715	0.066	0.108
	30	0.802	0.135	0.205
	40	0.827	0.113	0.198
30	10	0.733	0.115	0.139
	20	0.767	0.049	0.079
	30	0.613	0.075	0.120
	40	0.843	0.071	0.123

表 19　インスタンスタイプ 3 における配送効率性

$\|D\|$	$\|T\|$	ROTAP(S_{OPT}/S_{ROTAP})	FIFO(S_{OPT}/S_F)	RANK(S_{OPT}/S_R)
10	10	0.697	0.289	0.364
	20	0.900	0.315	0.407
	30	0.886	0.250	0.332
20	10	0.833	0.092	0.147
	20	0.867	0.066	0.102
	30	0.700	0.077	0.127
	40	0.841	0.131	0.195
30	10	0.750	0.058	0.085
	20	0.825	0.072	0.124
	30	0.858	0.084	0.174
	40	0.858	0.090	0.163

表 20　インスタンスタイプ 4 における配送効率性

$\|D\|$	$\|T\|$	ROTAP(S_{OPT}/S_{ROTAP})	FIFO(S_{OPT}/S_F)	RANK(S_{OPT}/S_R)
10	10	0.488	0.073	0.080
	20	0.699	0.131	0.176
	30	0.819	0.162	0.204
20	10	0.850	0.027	0.037
	20	0.767	0.023	0.040
	30	0.904	0.078	0.115
	40	-	-	-
30	10	0.900	0.035	0.039
	20	0.850	0.024	0.044
	30	-	-	-
	40	-	-	-

7.3 報酬のスケーリング

次に配達失敗コスト γ の値を変更せずに，γ の値を超えない範囲で報酬の値を 1.3 倍，1.5 倍にした時の割当率, 拒否率の結果を表 21, 22, 23, 24, 25, 26, 27, 28 にそれぞれインスタンスタイプとアルゴリズム毎に示す．コスト比とは 1.0 倍の時に Algorithm2 : FIFO と Algorithm3 : RANK によりそれぞれ得られるコストに対する比である．Algorithm2 : FIFO と Algorithm3 : RANK 共に報酬の値を大きくするにつれて割当率は上がり，拒否率は下がる結果となったがその差はわずかなものであった．表 25 の $|D| = 20, |T| = 20$ の時などコスト比が 1 より小さくなる時が見られることから，報酬を上げて実行するタスク数を多くすることができればサーバ全体で発生するコストの削減が期待できる．$|D| = 20, |T| = 40$ の時の各インスタンスタイプに対する Algorithm2 : FIFO と Algorithm3 : RANK のそれぞれの割当率，拒否率を図 11 に示す．

表 21　インスタンスタイプ 1:FIFO

		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	21.0%	58.7%	1.030	21.0%	58.7%	1.037	21.0%	58.7%
	20	17.0%	72.4%	1.005	18.5%	69.9%	1.003	19.5%	67.7%
	30	10.7%	80.2%	1.002	12.0%	78.2%	1.005	12.0%	78.2%
20	10	28.0%	72.2%	1.012	31.0%	69.9%	1.022	31.0%	69.9%
	20	21.0%	74.3%	1.017	22.0%	73.6%	1.019	22.5%	73.2%
	30	24.3%	72.5%	1.004	26.3%	70.9%	1.012	26.3%	70.9%
	40	23.3%	70.5%	1.004	25.0%	68.0%	1.011	24.8%	67.9%
30	10	32.0%	69.2%	1.036	33.0%	68.3%	1.026	35.0%	65.0%
	20	29.5%	69.3%	1.016	32.0%	66.2%	1.007	33.5%	64.1%
	30	27.3%	72.2%	1.005	29.7%	69.4%	1.011	30.0%	69.0%
	40	25.5%	72.1%	1.013	27.5%	70.5%	1.015	28.5%	69.9%

表 22　インスタンスタイプ 1:RANK

		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	23.0%	57.4%	1.003	27.0%	49.8%	1.005	28.0%	49.3%
	20	33.0%	38.3%	1.040	33.5%	35.4%	1.047	34.0%	33.3%
	30	24.3%	35.6%	1.011	26.7%	30.2%	1.022	26.7%	30.2%
20	10	49.0%	42.3%	1.057	51.0%	39.0%	1.083	51.0%	39.0%
	20	37.5%	44.2%	1.022	40.5%	39.5%	1.016	43.0%	36.2%
	30	41.7%	35.4%	1.046	43.0%	33.2%	1.064	43.3%	33.0%
	40	34.3%	42.8%	1.003	38.0%	34.8%	0.992	40.8%	30.0%
30	10	51.0%	44.2%	1.060	53.0%	39.7%	1.043	57.0%	35.9%
	20	46.0%	39.9%	1.038	49.5%	34.6%	1.049	50.5%	33.3%
	30	45.0%	36.7%	1.033	48.7%	31.9%	1.044	49.3%	30.5%
	40	45.5%	37.0%	1.049	47.0%	33.3%	1.071	47.0%	33.3%

表 23　インスタンスタイプ 2:FIFO

		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	18.0%	78.0%	1.016	19.0%	77.1%	1.019	19.0%	77.1%
	20	14.5%	78.0%	0.999	16.5%	74.4%	1.003	16.5%	74.4%
	30	17.3%	74.1%	1.010	18.0%	72.6%	1.014	18.0%	72.6%
20	10	31.0%	64.0%	1.017	34.0%	60.3%	1.018	36.0%	57.7%
	20	24.0%	73.7%	1.015	25.5%	71.8%	1.018	26.0%	71.3%
	30	24.0%	72.5%	1.010	26.3%	70.0%	1.019	26.3%	70.0%
	40	26.5%	73.1%	1.002	29.8%	69.7%	1.007	30.3%	69.3%
30	10	36.0%	61.8%	1.036	39.0%	58.7%	1.055	39.0%	58.7%
	20	29.0%	74.2%	1.013	31.5%	71.8%	1.018	31.5%	71.3%
	30	25.7%	73.2%	0.997	28.7%	69.4%	1.004	29.0%	69.1%
	40	29.5%	73.2%	1.027	29.8%	72.8%	1.018	31.5%	70.9%

表 24　インスタンスタイプ 2:RANK

		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	27.0%	52.4%	1.009	31.0%	46.8%	1.018	31.0%	46.8%
	20	32.0%	41.5%	1.005	36.0%	34.3%	1.019	36.0%	34.3%
	30	31.3%	42.0%	1.022	33.3%	37.8%	1.029	34.0%	36.5%
20	10	40.0%	44.3%	1.040	44.0%	41.0%	1.057	44.0%	41.0%
	20	45.5%	38.8%	1.032	50.5%	31.9%	1.042	52.0%	30.5%
	30	39.7%	38.4%	1.026	44.0%	34.0%	1.046	44.0%	34.0%
	40	47.8%	32.8%	1.047	50.3%	29.9%	1.061	51.0%	29.0%
30	10	46.0%	45.6%	1.058	49.0%	41.8%	1.084	49.0%	41.8%
	20	52.0%	38.0%	1.050	55.0%	33.3%	1.074	55.0%	33.3%
	30	48.3%	37.6%	1.027	52.3%	32.2%	1.044	53.0%	31.3%
	40	47.8%	35.8%	1.042	51.0%	31.6%	1.050	52.5%	29.6%

表 25　インスタンスタイプ 3:FIFO

		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	16.0%	81.0%	1.016	16.0%	80.8%	0.990	20.0%	74.0%
	20	17.0%	71.1%	0.994	18.5%	66.5%	0.999	18.5%	66.5%
	30	11.3%	81.7%	0.995	13.7%	78.8%	0.994	14.0%	78.2%
20	10	25.0%	72.2%	1.006	28.0%	66.9%	1.015	28.0%	66.9%
	20	20.5%	78.7%	0.977	25.5%	73.4%	0.984	25.5%	73.4%
	30	21.3%	75.1%	1.013	21.7%	73.9%	1.010	23.0%	72.6%
	40	21.8%	76.5%	0.994	24.8%	72.9%	0.998	24.8%	72.6%
30	10	44.0%	59.8%	1.039	48.0%	55.9%	1.047	49.0%	55.2%
	20	22.5%	77.5%	0.998	26.5%	73.2%	1.007	26.5%	73.2%
	30	22.3%	79.8%	0.984	26.0%	75.9%	0.990	26.0%	75.6%
	40	25.5%	75.4%	1.003	27.0%	73.1%	1.000	27.8%	72.1%

表 26　インスタンスタイプ 3:RANK

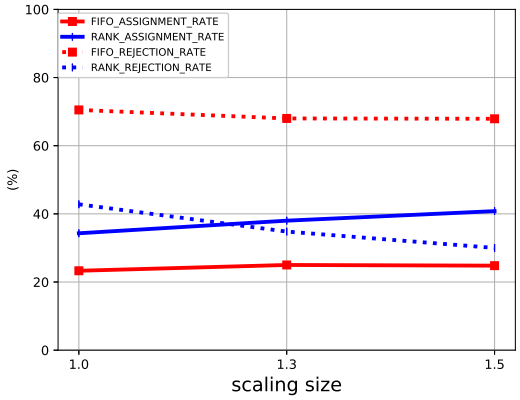
		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	34.0%	49.2%	0.986	39.0%	36.5%	0.953	43.0%	26.5%
	20	31.0%	40.0%	1.023	33.0%	34.0%	1.025	34.0%	29.7%
	30	27.3%	36.1%	1.029	28.3%	30.8%	1.041	28.3%	30.8%
20	10	44.0%	47.9%	1.016	48.0%	37.9%	1.022	49.0%	36.9%
	20	46.0%	38.8%	1.017	51.0%	32.2%	1.036	51.0%	32.2%
	30	44.0%	35.2%	1.028	47.3%	29.4%	1.039	48.7%	27.8%
	40	38.5%	40.6%	1.032	40.3%	35.5%	1.047	40.5%	34.9%
30	10	61.0%	39.0%	1.122	61.0%	39.0%	1.152	61.0%	39.0%
	20	50.0%	42.8%	1.037	55.0%	35.8%	1.057	55.5%	35.2%
	30	57.3%	32.0%	1.043	60.3%	25.2%	1.065	60.7%	24.9%
	40	50.5%	34.6%	1.059	52.0%	30.6%	1.078	52.7%	29.7%

表 27　インスタンスタイプ 4:FIFO

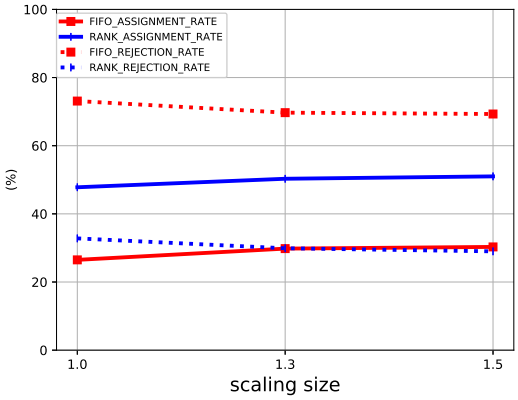
		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	12.0%	80.0%	0.996	14.0%	73.5%	1.000	14.0%	73.5%
	20	15.5%	74.8%	1.007	16.0%	74.0%	0.997	17.0%	70.7%
	30	13.0%	80.6%	0.998	14.7%	77.7%	0.996	15.3%	76.8%
20	10	22.0%	76.3%	1.011	25.0%	73.0%	1.021	25.0%	73.0%
	20	25.0%	73.7%	1.019	25.5%	73.1%	1.027	25.5%	73.1%
	30	20.7%	78.0%	0.991	24.0%	74.4%	0.990	24.3%	73.5%
	40	25.5%	74.0%	0.995	28.0%	70.6%	0.997	28.5%	70.0%
30	10	37.0%	70.7%	1.004	40.0%	67.4%	1.020	40.0%	67.4%
	20	28.5%	74.9%	1.023	29.5%	73.6%	1.034	29.5%	73.6%
	30	25.0%	79.3%	1.003	28.0%	76.6%	1.008	28.7%	76.0%
	40	28.0%	71.6%	1.016	29.0%	70.4%	1.012	30.5%	68.6%

表 28　インスタンスタイプ 4:RANK

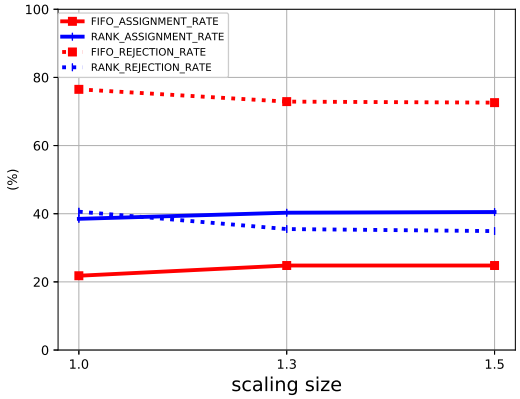
		1.0 倍		1.3 倍			1.5 倍		
$\|D\|$	$\|T\|$	割当率	拒否率	コスト比	割当率	拒否率	コスト比	割当率	拒否率
10	10	26.0%	53.6%	1.005	29.0%	43.8%	0.991	32.0%	39.9%
	20	31.0%	45.0%	1.030	32.5%	42.9%	1.037	33.0%	41.3%
	30	24.0%	58.2%	1.005	26.3%	51.1%	1.010	26.7%	49.2%
20	10	45.0%	48.5%	1.050	48.0%	44.9%	1.066	49.0%	43.9%
	20	45.0%	45.3%	1.031	48.5%	39.7%	1.050	49.0%	39.1%
	30	40.0%	44.9%	1.030	43.3%	41.4%	1.037	44.0%	39.8%
	40	42.0%	41.1%	1.042	44.0%	38.7%	1.053	44.8%	37.4%
30	10	60.0%	41.8%	1.056	64.0%	37.0%	1.081	65.0%	36.1%
	20	51.0%	43.6%	1.035	55.5%	38.4%	1.064	55.5%	38.4%
	30	45.7%	48.5%	0.986	53.0%	39.1%	1.011	53.0%	39.1%
	40	44.8%	41.2%	1.037	47.8%	37.6%	1.058	47.8%	37.4%



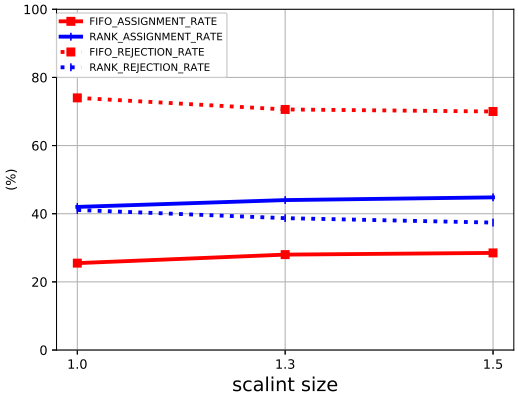
[1] インスタンスタイプ 1



[2] インスタンスタイプ 2



[3] インスタンスタイプ 3



[4] インスタンスタイプ 4

図 11　$|D| = 20, |T| = 40$ の時の各インスタンスタイプの割当率，拒否率

7.4 より過密なインスタンス

昼間や夕方のピーク時を想定し，より過密にタスクが発生した場合の考察を行う．時間範囲を昼間の 11 時から 13 時の 2 時間に変更したインスタンスに対して $|D|, |T|$ を変化させてそれぞれ 10 個ずつインスタンスを生成，Algorithm2 : FIFO と Algorithm3 : RANK をそれぞれ適用した時の割当率と拒否率の平均の結果を表 29 に示す．ワーカのタイムウィンドウは 30 分から 2 時間とし，店の配達先の位置は中心に多く設定，対象範囲を 1 マス 1km の 5×5 マスとし，タスクのタイムウィンドウは 10 分から 30 分とした．コスト比は Algorithm2 : FIFO で得られるコストに対する Algorithm3 : RANK により得られるコストの比である．表 29 より $|D|, |T|$ が増加するにつれて Algorithm2 : FIFO の割当率が上がり拒否率が下がっているが，Algorithm3 : RANK より得られる割当率は減少する結果となった．コストに関しては全体を通して，Algorithm3 : RANK より得られるコストの方が小さい結果となっているが，これは Algorithm2 : FIFO は Algorithm3 : RANK よりも各ワーカ，各タスクあたりに対する割当回数が多いので割当失敗コストが多く発生していることが原因だと考えられる．

表 29　ピーク時を想定した場合

		FIFO		RANK		
$\|D\|$	$\|T\|$	割当率	拒否率	割当率	拒否率	コスト比
50	100	33.9%	77.5%	40.1%	34.1%	0.563
100	150	35.1%	53.1%	32.3%	23.4%	0.546
100	200	37.7%	38.6%	29.0%	15.4%	0.577
200	300	38.8%	26.2%	22.3%	9.8%	0.562

次に同一のタスクに対して $|D|$ を 10 ずつ増加させた時の Algorithm2 : FIFO と Algorithm3 : RANK より得られる割当率と拒否率を見てみる．コスト比は Algorithm2 : FIFO で得られるコストに対する Algorithm3 : RANK により得られるコストの比である．$|T|$ が 100 の時の結果を表 30，150 の時の結果を表 31，200 の時の結果を表 32 にそれぞれ示す．また $|D|$ の増加による Algorithm2 : FIFO と Algorithm3 : RANK による割当率と拒否率の推移を各タスク数毎に図 12，13，14 に示す．全体を通して Algorithm2 : FIFO で得られる拒否率よりも Algorithm3 : RANK で得られる拒否率の方が低い結果となっている．また Algorithm2 : FIFO で得られる拒否率は $|T|$, $|D|$ に依らず常に 80% 前後となっている．表 30 より $|T| = 100$ の時は $|D|$ をタスク数と同等まで増加させても Algorithm2 : FIFO よりも Algorithm3 : RANK で得られる割当率の方が高い結果となった．表 31，32 より $|D|$ を増加させていくと，あるところから Algorithm2 : FIFO で得られる割当率の方が Algorithm3 : RANK で得られる割当率よりも高い結果となっている．コストに関しては全体を通して Algorithm2 : FIFO よりも Algorithm3 : RANK の方がサーバ全体で生じるコストが小さい結果となっている．

表 30　$|T| = 100$

	FIFO		RANK		
$\|D\|$	割当率	拒否率	割当率	拒否率	コスト比
10	10.9%	85.5%	15.9%	52.0%	0.761
20	19.6%	81.9%	28.6%	46.9%	0.676
30	28.7%	78.0%	35.4%	41.1%	0.629
40	30.9%	78.3%	36.9%	39.6%	0.604
50	36.8%	76.1%	38.8%	36.4%	0.588
60	36.7%	77.3%	40.1%	34.9%	0.558
70	36.7%	78.6%	42.6%	31.9%	0.514
80	35.4%	79.5%	37.3%	39.2%	0.549
90	37.8%	79.4%	42.9%	31.0%	0.489
100	39.4%	79.3%	45.5%	30.2%	0.474

表 31　$|T| = 150$

	FIFO		RANK		
$\|D\|$	割当率	拒否率	割当率	拒否率	コスト比
10	8.7%	89.7%	12.0%	58.4%	0.732
20	15.4%	85.4%	20.1%	49.1%	0.680
30	23.5%	80.7%	26.1%	43.4%	0.655
40	31.2%	77.3%	30.2%	37.9%	0.631
50	34.1%	76.5%	28.7%	40.7%	0.644
60	38.2%	75.8%	30.5%	38.1%	0.619
70	34.5%	78.7%	31.9%	35.5%	0.573
80	36.8%	77.5%	31.3%	36.6%	0.585
90	36.5%	78.5%	32.7%	34.3%	0.561
100	36.3%	79.6%	32.3%	35.8%	0.546
110	36.9%	79.3%	33.7%	32.3%	0.530
120	38.3%	79.4%	33.9%	32.3%	0.522
130	38.4%	79.8%	34.7%	31.4%	0.509
140	36.4%	81.3%	35.3%	30.6%	0.488
150	39.1%	80.0%	35.9%	28.6%	0.491

表 32　$|T| = 200$

$\|D\|$	FIFO		RANK		コスト比
	割当率	拒否率	割当率	拒否率	
10	6.9%	89.3%	9.0%	64.8%	0.771
20	12.5%	86.6%	15.0%	53.9%	0.708
30	16.9%	85.0%	18.9%	49.7%	0.670
40	22.4%	82.3%	22.8%	42.8%	0.641
50	26.6%	80.4%	22.6%	43.9%	0.646
60	30.5%	78.9%	24.4%	39.5%	0.628
70	32.4%	78.8%	23.4%	43.6%	0.635
80	35.4%	77.8%	24.4%	40.1%	0.620
90	36.1%	78.1%	23.9%	41.9%	0.616
100	35.6%	79.0%	24.1%	42.5%	0.603
110	34.2%	79.8%	25.1%	39.8%	0.584
120	37.1%	78.7%	26.4%	37.8%	0.578
130	37.7%	79.0%	27.3%	35.0%	0.560
140	34.5%	81.1%	25.7%	38.7%	0.552
150	36.9%	80.4%	25.4%	39.6%	0.556
160	38.4%	79.9%	26.6%	36.2%	0.544
170	38.0%	80.5%	26.2%	37.1%	0.536
180	38.5%	80.6%	26.2%	37.4%	0.533
190	37.6%	81.3%	27.5%	35.1%	0.512
200	34.6%	83.0%	25.4%	39.8%	0.516

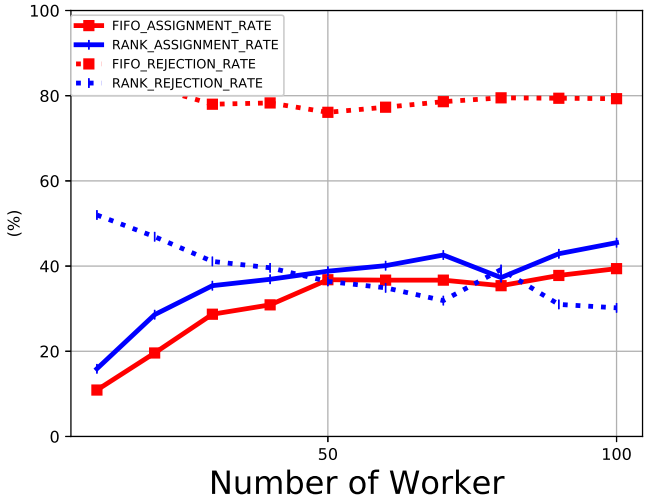


図 12　$|T| = 100$ の時のワーカ数増加による割当率と拒否率の推移

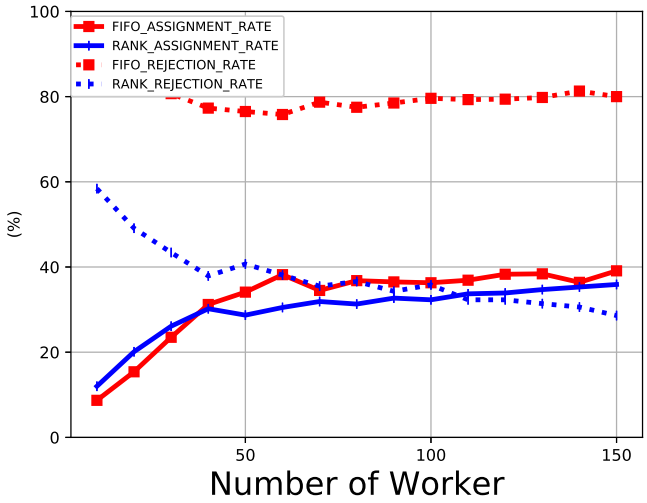


図 13　$|T| = 150$ の時のワーカ数増加による割当率と拒否率の推移

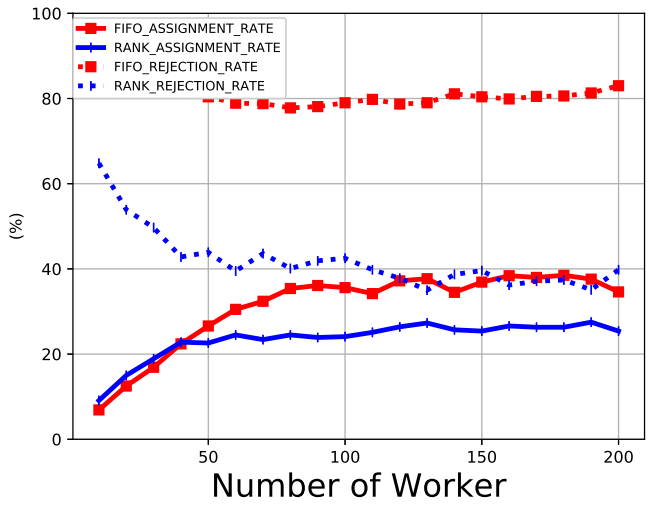


図 14　$|T| = 200$ の時のワーカ数増加による割当率と拒否率の推移

8　おわりに

本研究では，空間クラウドソーシングを適用したクラウドソーシング型ピックアップアンドデリバリー問題のモデル化，及びワーカとタスクの情報がそれぞれ動的にサーバに到着するオンラインモデルに対して，サーバ全体で生じるコスト最小化を目的とした動的にワーカにタスクを割当てるオンラインアルゴリズムの提案を行なった．特に，ワーカがタスクに対して選好タイプを持ち，サーバが割当てたタスクをワーカが実行するかどうか選択可能な場合に対して，サーバ側がワーカの選好タイプを取得可能な場合を想定し，ワーカにタスクを拒否されない割当てを行うようなオンラインアルゴリズムの性能を実験によって評価した．また，配送効率性を見ることで経済分析を行なった．

今後の課題として，配送効率性向上のためのワーカのインセンティブ設計として，配達失敗コストと各タスクの報酬との関係を分析することが挙げられる．特に，タイムウィンドウの終了時刻が迫っているタスクに対する報酬増加などのインセンティブ設計は，配送効率性向上に寄与すると考えられる．インセンティブの与え方を理論及び実験的に分析することが今後の課題である．

謝辞

本論文の執筆にあたり指導教官の高橋里司先生からは多大な助言，指導を賜りました．深く感謝致します．また，最適化輪読や最適化ゼミを通して指導してくださいました村松正和先生にも感謝の意を表します．同一の学生部屋で過ごした高橋研究室，村松研究室，保木研究室の学生の皆様にも謝意を示します．

参考文献

[1] 楽天株式会社,2018 年度通期および第 4 四半期決算説明会 (2019 年 2 月 12 日)

[2] 株式会社野村総合研究所,2021 年度までの ICT・メディア市場の規模とトレンドを展望 (2015 年)

[3] 経済産業省報告書, 平成 29 年度我が国におけるデータ駆動型社会に係る基盤整備 (電子商取引に関する市場調査) 平成 30 年 4 月

[4] 総務省, シェアリングエコノミー: http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h29/html/nc112220.html(最終閲覧日:2019/12/23)

[5] Airbnb:https://www.airbnb.jp(最終閲覧日:2019/12/23)

[6] Uber:https://www.uber.com/jp/ja/(最終閲覧日:2019/12/23)

[7] 滴滴出行:https://www.didiglobal.com(最終閲覧日:2019/12/23)

[8] メルカリ:https://www.mercari.com/jp/(最終閲覧日:2020/1/15)

[9] Readyfor:https://readyfor.jp(最終閲覧日:2020/1/15)

[10] UberEats:https://www.ubereats.com/ja-JP/tokyo/(最終閲覧日:2019/12/23)

[11] MTurk:https://www.mturk.com(最終閲覧日:2019/12/24)

[12] Upwork:https://www.upwork.com(最終閲覧日:2019/12/24)

[13] GrubHub:https://investors.grubhub.com/investors/overview/default.aspx(最終閲覧日:2019/12/24)

[14] 美団:https://www.meituan.com(最終閲覧日:2019/12/24)

[15] Jeff Howe:Crowdsourcing:A definition,http://crowdsourcing.typepad.com/cs/2006/06/crowdsourcing_a.htm(2006)

[16] Jeff Howe:The rise of crowdsourcing,Wired 14(6):14(2006)

[17] A. Schrijver: Combinatorial Optimization - Polyhedra and Efficiency.(Springer, 2003)

[18] Yongxin Tong, Zimu Zhou, Yuxiang Zeng, Lei Chen, Cyrus Shahabi:Spatial crowdsourcing:a survey.Springer-Verlag GmbH Germany, part of Springer Nature(2019)

[19] Michael L. Pinedo: Scheduling Theory, Algorithms, and Systems Fifth Edition.(Springer, 2016)

[20] Alp M. Arslan, Niels Agatz, Leo Kroon, Rob Zuidwijk:Crowdsourced Delivery: A Dynamic Pickup and Delivery Problem with Ad-hoc drivers,Transportation

Science(Vol.52,Issue 1 January-Feburary 2019)

[21] Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, Xue Zhu.:How to Match when All Vertices Arrive Online. In:Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of computing, pp.17-29(2018)

[22] Yajun Wang, Sam Chiu-wai Wong.:Two-sided Online Bipartite Matching and Vertex Cover: Beating the Greedy Algorithm. In:42nd International Colloquium on Automata, Languages, and Programming, pp.1070-1081(2015)

[23] Umair ul Hassan, Edward Curry.:A Multi-armed Bandit Approach to Online Spatial Task Assignment. In: 2014 IEEE 11th International Conference on Ubiquitous Intelligence and Computing and 2014 IEEE 11th Internaitonal Conference on Autonomic and Trusted Computing and 2014 IEEE 14th International Conference on Scalable Computing and Communications and Its Associated Workshops, pp.212-219(2014).

[24] Yongxin Tong, Libin Wang, Zimu Zhou, Bolin Ding, Lei Chen, Jieping Ye, Ke Xu.:Flexible online task assignment in real-time spatial data. PVLDB 10(11),1334-1345(2017)

[25] Shashank Goyal, Diwakar Gupta.: The Online Reservation Problem.a thesis submitted to the faculty of the university of minnesota(2018)