短時間再割り当てを考慮した組み合わせオークション勝者決定の高速近似手法

福田 直樹 伊藤 孝行

我々は，組み合わせオークションをアプリケーションシステムに適用する場合に，勝者決定問題に対して解の最適性を厳密に保証することが計算資源的に難しいのなら，現実的な計算時間で質の高い近似解を求めたうえで，その解に対してある一定の好ましい性質を持たせられるようにするアプローチもありえると考え，組み合わせオークションの勝者決定に関する新しい近似アルゴリズムと，それが持つべき好ましい性質についての解析および考察を行ってきた．しかしながら，これまでの我々のアプローチ，および関連研究のアプローチでは，いずれも，個々のオークションを独立した問題としてとらえ，オークション同士の関係を利用した勝者決定の近似の効率化の可能性については検討してこなかった．本論文では，ユビキタス環境下での資源配分問題のように，短時間で状況が変化し，それに伴って資源の再割り当ての必要が頻繁に生じるような状況下のように，組み合わせオークションが何度も繰り返され，そのオークション間の入札の差分が小さい場合に，直前のオークションにおける近似勝者を効果的に再利用する手法について述べる．

In this paper, we propose enhanced approximation algorithms of combinatorial auction that are suitable for the purpose of periodical reallocation of items. Our algorithms are designed to effectively reuse the last solutions to speeding up initial approximation performance. We show experimental results that show our proposed algorithms outperform existing algorithms in some aspects when the existing bids are not deleted. Also, we propose an enhanced algorithm that effectively avoids undesirable reuse of last solutions in the algorithm. This is especially effective when some existing bids are deleted from the last cycle.

1 はじめに

組み合わせオークションとは，入札の対象として単一の財に対してでなく，複数の財の組み合わせに対して入札が可能なオークションである [1]．組み合わせオークションには，すでに電子商取引で広く普及している単一財を対象としたオークションなどに置き換わる新たなオークションメカニズムとして，広く普及する可能性がある．Sandholm らによる例 [16] の他，米国 FCC による周波数帯域割り当てへの適用が検討された例も報告されている [2]．同時に，他の多くの複雑な組み合わせ最適化問題の近似にも適用できることが明らかにされつつある．たとえば，チリにおける給食の配分効率化問題への適用事例が報告されている [3]．

組み合わせオークションの勝者決定問題に対しては，最適解を高速に求めるアルゴリズムについての研究が進みつつある [14][15][4][1] が，この問題は NP-hard であることが知られており [1]，特に入札数の増加に対して爆発的に計算が複雑になる．たとえば，CASS アルゴリズム [4] では，我々が試した限りでは，1つのオークションに対する入札数がおおよそ 3000 を超えると，最適解を求めることが非常に困難になる．一方で，組み合わせオークションにおける入札数は，代替可能財を扱おうとすると，爆発的に増加してしまう．さらに，エージェント（プログラム）による入札の自動生成を考えると，入札者の持つ選好関数などからそれを満たす（人手では到底不可能な）大量の入札を作り出すことができるため，エージェントが参加可

Towards Approximated Short Cycle Reallocation on Combinatorial Auctions.
Naoki Fukuta, 静岡大学情報学部, Faculty of Informatics, Shizuoka University.
Takayuki Ito, 名古屋工業大学大学院産業戦略専攻, Master Cource of Techno-Business Administration, Nagoya Institute of Technology.

能な組み合わせオークションでは，非常に大量の入札を前提とした高速な勝者決定が求められる．

組み合わせオークションにおける高速な最適解の探索手法と，それによる組み合わせオークションの様々な有益な性質を保証することが非常に重要であることは明らかである．しかし，我々は，組み合わせオークションをアプリケーションシステムに適用する場合に，勝者決定問題に対して解の最適性を厳密に保証することが計算資源的に難しいのなら，現実的な計算時間で質の高い近似解を求めたうえで，その解に対してある一定の好ましい性質を持たせられるようにするアプローチもありえると考えている．そこで，我々はこれまでに，組み合わせオークションの勝者決定に関する新しい近似アルゴリズム[7]と，それが持つべき好ましい性質[9]についての解析[6][5]および考察[8]を行ってきた．

しかしながら，これまでの我々のアプローチ，および関連研究のアプローチでは，いずれも，個々のオークションを独立した問題としてとらえ，オークション同士の関係を利用した勝者決定の近似の効率化の可能性については検討してこなかった．本論文では，ユビキタス環境下での資源配分問題のように，短時間で状況が変化し，それに伴って資源の再割り当ての必要が頻繁に生じるような状況下のように，組み合わせオークションが何度も繰り返され，そのオークション間の入札の差分が小さい場合に，直前のオークションにおける近似勝者を効果的に再利用する手法について述べる．

2 組み合わせオークションと勝者決定問題

2.1 Lehmann アルゴリズム

組み合わせオークションの勝者決定問題は計算論的に複雑度の高い問題であるため，その近似解法についての研究もなされている．勝者決定問題の近似解法としてよく知られたものに，Lehmann らによって提案されたアルゴリズム[12]がある．

Lehmann のアルゴリズムは，比較的単純な，欲張り (greedy) アルゴリズムであり，その計算量は（事前に行われるソーティングを除けば）入札数に対して線形オーダに近い性質を示す．ここで，ある入札を $b = < a, s >$(ただし，$s \subseteq M$ and $a \in \mathcal{R}_+$) と表現することにする．また，2つの入札 $b = < a, s >$ および $b' = < a', s' >$ が競合するのは，$s \cap s' \neq \emptyset$ のときであるとする．このとき，Lehmann アルゴリズムは次のように表せる．

1. 入札は，ある基準によって事前にソートされる．文献[12]では，直感的に表現すると，入札のリスト L を，財あたりの入札額の平均値によって降順でソートする方法が提案されている．これを，より一般化したものとして，文献[12]では，入札のリスト L を，$a/|s|^c$ の値によってソートする方法を提案している．ここで，c の値は $c \geq 0$ の任意の値であり，全体の財の個数 k に関連させるとよいことが指摘されている．．
2. ソートされた入札のリスト L に対して，欲張りアルゴリズムにより財が割り当てられる．財の割り当ては，単純に入札リスト L を前方から見ていき，その入札の財のバンドルがまだ未割り当てであり，かつ他の（これまでに割り当てられ勝者となった）入札と競合しないとき，その入札を勝者とすることを繰り返し行う．

文献[12]では，ソート時のパラメータ c について，$c = 1/2$ とすると良好な結果が得られると述べられている．そこで，本論文では，以後は特に断りがない限り $c = 1/2$ を c の標準値として話を進める．

2.2 山登り探索による解の改善

Lehmann アルゴリズムでは，財の割り当て結果が下界となる場合があるが，この下界は条件によっては（とくに財の個数 k が非常に大きい場合）かなり低くなり，実際にアルゴリズムを適用した場合でも最適解に対して 50%前後の解が出てくるなど，結果のばらつきが多く見られる．我々はこれまでに，文献[7]などで，Lehmann アルゴリズムによって得られた財の割り当てを初期解として，その初期解を山登り探索により漸次的に洗練する手法を提案してきている．山登り探索では，現在の解の近傍を探索し，現在の解よりも良いものがあればそれを選んで次の解とする，というプロセスを，新しい解が見つからなくなるまで繰り返す．

次のアルゴリズムは，単一入札距離での山登り探索に基づく，近似勝者決定アルゴリズムである．
入力となるのは，入札のリスト L と 初期解 *Alloc* であり，ここでは，それぞれ Lehmann アルゴリズムで用いられた入札のリストと，Lehmann アルゴリズムで得られた解である．

```
function LocalSearch(Alloc, L)
  RemainBids:= L - Alloc;
  for each b ∈ RemainBids as sorted order
    if b conflicts Alloc then
    Conflicted:=Alloc - consistentBids({b}, Alloc);
      NewAlloc:= Alloc - Conflicted + {b};
      ConsBids:=
        consistentBids(NewAlloc, RemainBids);
      NewAlloc:=NewAlloc+ConsBids;
     if price(Alloc) < price(NewAlloc) then
       return LocalSearch(NewAlloc,L);
  end for each
  return Alloc
```

関数 “consistentBids” は，“RemainBids” にある入札を順に辿って，内部に競合入札のない新しい解 “NewAllocn” を見つける．これは，“Alloc” に対して，新たな入札を挿入し，その入札に競合する入札を解から取り除くことで行われる．競合していた入札を解から取り除いた後には，競合を起こさずに解に追加できる入札が存在する場合がある．その場合には，競合を起こさない範囲で，リストのソート順が上位のものから可能な限りの財が落札できるように入札を解に挿入する．

2.3 複数値のソート因子 c に対する並列探索

Lehmann アルゴリズムによる近似解は，入札リストをソーティングするときの因子である c の値の影響を強く受ける．我々は，山登り法による解の改善アルゴリズムに対して，複数の c の値に対して並列に探索をさせることで，解の最適性を改善する手法を提案してきている [9],[7].

異なる c に対する探索結果は 1 つにまとめられ，最終的な解としては，それらのうちでもっともよいものが取られる．本論文では，実行時並列度の低い状況下での制約された計算時間内での性能も考慮して，$C = \{0,\ 0.5,\ 1\}$ の 3 つの c の値を用いることとした．

3 近似アルゴリズムの拡張

3.1 直前の近似結果の部分的再利用による高速化

定期的に何度も資源の再割り当てを行うようなシナリオを考えた場合，その割り当て毎に全く異なる入札集合を扱うことはむしろ稀であり，直前の入札集合からわずかに変更されたものを対象にする場合が多い．しかしながら，これまでに用いられてきた勝者決定アルゴリズムでは，たとえ入札集合が似ている組み合わせオークションであっても，それらを独立した別個の問題として扱うようになっていたため，たとえわずかな入札の変更が加えられただけであっても，勝者を改めて最初から決定し直す必要があった．そこで，直前に勝者近似で用いられた入札集合とそこでの近似結果を再利用し，近似を高速化することが可能なアルゴリズムを，以下で提案する．以下のアルゴリズムでは，対象とするオークションの入札集合のみでなく，直前の入札集合 (*LastBids*) とそのときの勝者集合 (*LastWinners*) を与えることで，その差分を計算し，勝者近似をより短時間で行えるようにする．

```
Function PartialReallocationA(
            LastBids,LastWinners,CurrentBids)
  AddedBids :=
    CurrentBids - (LastBids ∩ CurrentBids);
  DeletedBids :=
    LastBids - (LastBids ∩ CurrentBids);
  Winners := LastWinners;
  foreach d ∈ DeletedBids
    if d ∈ Winners
      then Winners := Winners − d;
   foreach a ∈ AddedBids
     foreach w ∈ Winners
       if b(w) = b(a) and v(w) < v(a)
         then Winners := Winners − w + a;
  Winners := LocalSearch(Winners,CurrentBids);
   return Winners
```

本アルゴリズムでは，最初に，直前の入札集合と比較して「削除された」と考えられる入札を，勝者集合から削除する．次に，「新たに加えられた」入札について，それを必要に応じて勝者集合と置き換えていく．このときの置き換えでは，新たに加えられた入札がすでに勝者となっている入札とまったく同

一の財の集合に対して入札を行っており，かつその入札額がすでに勝者となっている入札より大きい場合にのみ，置き換えが適用される．なお，入札の改変 (modification) は，入札の削除と新規入札の追加の２つのオペレーションの組み合わせに分解されたものとして扱われる．

3.2 近似結果の再利用による性能劣化への対処

一般的には，類似した問題の答えを再利用することは，最適化問題を解く上では必ずしも有利には働かない．実際に，類似した問題の答えを利用することで，むしろ性能が劣化する場合が，しばしば生じる．このような性能劣化をできるかぎり少ない計算オーバヘッドで避ける方法を考える．

我々がここで採用する方法は，単純に，「再利用後の結果が単純な greedy 割り当てより劣るのであれば，それを捨て，greedy 割り当てに置き換える」というものである．これまでの実験結果により，greedy 割り当ての計算オーバヘッドは非常に小さいことがわかっており，我々が先に示した部分的再利用アルゴリズムにおける計算オーバヘッドは十分軽微なものであれば，仮に近似結果の再利用がうまくいかなかった場合でも，その計算オーバヘッドが全体の結果に悪影響を及ぼさないと考えられる．以下が，そのアルゴリズムの記述である．

```
Function PartialReallocationX(
            LastBids,LastWinners,CurrentBids)
  AddedBids :=
    CurrentBids - (LastBids ∩ CurrentBids);
  DeletedBids :=
    LastBids - (LastBids ∩ CurrentBids);
  Winners := LastWinners;
  foreach d ∈ DeletedBids
    if d ∈ Winners
      then Winners := Winners − d;
   foreach a ∈ AddedBids
    foreach w ∈ Winners
      if b(w) = b(a) and v(w) < v(a)
        then Winners := Winners − w + a;
  GreedyWinners := GreedySearch(CurrentBids);
  if price(Winners) =< price(GreedyWinners)
    then Winners := GreedyWinners;
  Winners := LocalSearch(Winners,CurrentBids);
  return Winners
```

4 比較実験

本論文では，比較に用いたアルゴリズムを以後は次のように表記する．“greedy(c=0.5)” は，Lehmann のアルゴリズムを ($c = 0.5$) で実行した結果である．“HC(c=0.5)” は，$c = 0.5$ の条件下で，山登り探索手法によって，Lehmann アルゴリズムによる結果を洗練したものである．“greedy-3” および “HC-3” で始まるものは，それぞれ，Lehmann のアルゴリズムあるいは山登り探索手法による洗練を ($c = \{0,\ 0.5,\ 1\}$) の３つのパラメータについて行い，そのうちで最良となる結果を選んだものである．また，本論文で提案する２つのアルゴリズム (`PartialReallocationA`, `PartialReallocationX`) を用い，後述の方法で近似解を再利用した結果を “AHC-3”, および “XHC-3” で始まる名前で表記している．なお，“-3” の表記からわかるとおり，いずれも，$c = \{0,\ 0.5,\ 1\}$ の３つのパラメータについて最良となる結果を選んだものである．それぞれ，後ろに “-seq” がついたものは３つのパラメータに対して計算を単一処理スレッドとして逐次的に行った場合の結果を示し，“-para” がついたものは３つのパラメータに対して計算を完全に並列に行った場合の理論値を示す．それぞれについて，実験時に設定した計算打ち切り時間 (“-100ms”,“-1000ms”) をアルゴリズムの表記の末尾に付記している．“Zurel-1st” は，Zurel らのアルゴリズム [17] で得られる初期解を示す．“Zurel” は，Zurel らのアルゴリズムが停止するまで計算を行わせ，最終的に得られた結果を示す．“casanova” で始まるものは，Hoos が提案した Casanova アルゴリズム [11] による結果を示し，その後ろに計算打ち切り時間 (“-10ms”,“-100ms”,“-1000ms”) を付記している．

組み合わせオークション問題の生成手法については，Leyton-Brown らの CATS [13] がよく知られている．CATS を用いた数万以上の多数の入札を含むオークションテストデータの作成は，それ自体に非常に計算時間がかかる．本論文では，入札数 20,000，財数 256 の条件で，CATS の標準パラメータを用い，現実的な時間で生成可能であった入札分散 (L2,L3,L4,L6,L7)

表 1　近似時間と近似の最適性 (20,000 入札,256 財,Zurel=1 に正規化)

	L2		L3		L4		L6		L7	
greedy(c=0.5)	1.0002	(23.0)	0.9639	(19.0)	0.9417	(23.0)	0.9389	(23.4)	0.7403	(22.1)
greedy-3-seq	1.0003	(69.1)	0.9639	(59.2)	0.9999	(72.9)	0.9965	(67.8)	0.7541	(66.8)
greedy-3-para	1.0003	(26.4)	0.9639	(20.9)	0.9999	(28.4)	0.9965	(26.0)	0.7541	(25.5)
HC(c=0.5)-100ms	1.0004	(100)	0.9741	(100)	0.9576	(100)	0.9533	(100)	0.8260	(100)
HC-3-seq-100ms	1.0004	(100)	0.9692	(100)	1.0000	(100)	0.9966	(100)	0.8287	(100)
AHC-3-seq-100ms	-		0.9690	(100)	1.0006	(100)	0.9974	(100)	1.0225	(100)
XHC-3-seq-100ms	-		0.9813	(100)	1.0005	(100)	0.9987	(100)	1.0217	(100)
HC-3-para-100ms	1.0004	(100)	0.9743	(100)	1.0001	(100)	0.9969	(100)	0.9423	(100)
AHC-3-para-100ms	-		0.9741	(100)	1.0006	(100)	0.9977	(100)	1.0249	(100)
XHC-3-para-100ms	-		0.9820	(100)	1.0006	(100)	0.9988	(100)	1.0249	(100)
HC(c=0.5)-1000ms	1.0004	(1000)	0.9856	(1000)	0.9771	(1000)	0.9646	(1000)	1.0157	(1000)
HC-3-seq-1000ms	1.0004	(1000)	0.9804	(1000)	1.0003	(1000)	0.9976	(1000)	1.0086	(1000)
AHC-3-seq-1000ms	-		0.9795	(1000)	1.0007	(1000)	0.9982	(1000)	1.0266	(1000)
XHC-3-seq-1000ms	-		0.9830	(1000)	1.0006	(1000)	0.9991	(1000)	1.0266	(1000)
HC-3-para-1000ms	1.0004	(1000)	0.9856	(1000)	1.0006	(1000)	0.9987	(1000)	1.0240	(1000)
AHC-3-para-1000ms	-		0.9847	(1000)	1.0008	(1000)	0.9990	(1000)	1.0272	(1000)
XHC-3-para-1000ms	-		0.9853	(1000)	1.0008	(1000)	0.9996	(1000)	1.0272	(1000)
Zurel-1st	0.5710	(11040)	0.9690	(537)	0.9983	(2075)	0.9928	(1715)	0.6015	(1795)
Zurel	1.0000	(13837)	1.0000	(890)	1.0000	(4581)	1.0000	(4324)	1.0000	(3720)
casanova-10ms	0.2583	(10)	0.0069	(10)	0.0105	(10)	0.0202	(10)	0.2577	(10)
casanova-100ms	0.2583	(100)	0.0069	(100)	0.0105	(100)	0.0202	(100)	0.2577	(100)
casanova-1000ms	0.5357	(1000)	0.1208	(1000)	0.0861	(1000)	0.1486	(1000)	0.7614	(1000)

(括弧内の数値はミリ秒単位での計算時間)

のみに対して，CATS を用いて各入札分散ごとに 100 試行分のテストデータを準備した．なお，ここで用いた入札分散は，過去に様々な論文内で提案された入札の人工的生成方法をほぼ網羅しており，入札分散の表記は，Leyton-Brown らの文献[13] による．各入札分散の標準パラメータや入札生成アルゴリズムの詳細については，Leyton-Brown らの文献[13] を参照されたい．

CATS の問題生成手法では，オークションにおける入札は静的なものとして扱われているため，本論文で述べるような，少量の入札差分をもつ一連のオークションの集まりとしては問題が構成されない．そこで，本論文では，手順 1 に示す方法で，CATS で生成した問題を，少量の入札差分をもつ一連のオークションの集まりとして再構成した．

手順 1　CATS で生成したそれぞれのオークション問題について，その入札集合を，入札生成順序に沿って 10 等分する．この 10 等分した入札集合を用いて，1 秒毎に入札集合を変更する操作を合計 10 回行う．ここで，各入札集合は，10 等分されたものの先頭から 1 つづつ順に「削除」印をずらしていき,「削除」印のない入札集合のみを集めたものを，その回のオークションの入札集合とする．たとえば，最初の 1 秒間における入札集合は，10 等分されたうちの先頭の 1 つに「削除」印が付され，それを除いた 9 つの入札部分集合の和で入札集合が構成される．その 1 秒後には，さきほど入札集合に含まれていなかった最初の 1 ブロック分の入札部分集合が入札集合に追加され，そのかわりに，先頭から 2 ブロック目の入札部分集合に「削除」印が付され，入札集合から削除される．これを繰り返し，最後の 1 回のみ，すべての入札部分集合からなる入札集合を対象として勝者決定が行われる．

この手順では，従来我々が提案，および比較対象としてきた手法では，入札集合の更新のたびに，あらためて勝者決定を最初からやり直すことになる．本論文で提案する手法では，入札集合の更新の際に，直前の入札集合に対する近似勝者を再利用しようとする．ここで重要な点は，最後の 1 回の試行ではすべての入札を対象に勝者決定が行われるため，その結果は，これまでに我々が比較実験に用いてきた結果と直接的に比較することができる点である．

表1に，本手順に基づいた最後の1回の試行時での，各近似アルゴリズムにおける計算時間と近似解の最適性の関係をまとめる．実験環境には， Mac OS X 10.4, CPU: CoreDuo 2.0GHz, 2GBytes memory のラップトップ型コンピュータを用いている．表1中のそれぞれの値は，左側が近似解の最適性 (Zurel比) を，右側の括弧内の数値がそれに要した計算時間 (ミリ秒) を，それぞれ示している．なお，入札数が 3,000 を越える場合には，CASS [4] や CPLEX 等の高速に動作することが知られている最適解探索アルゴリズムであっても，その最適解はおろか最大落札額を推定することすらきわめて困難となることが知られている [4]．本論文では，アルゴリズム間の相対的な性能比較を行うことを目的としているため，近似解の最適性を示す値として，各入札分散ごとに各アルゴリズムでの総落札価格の平均値を計算し，Zurel らの手法による値を1として正規化したものを提示している．†1

ほとんどの入札分散について，“Zurel-1st” では1秒以上の時間がかかっているにもかかわらず，その近似性能は “greedy-3-seq” に劣っている．我々がこれまでに提案してきた “HC-3” では，入札分散 L3 を除けば，その計算にかかる時間が “Zurel-1st” や “Zurel” と比較して十分小さいにもかかわらず，その近似性能は同程度となっている．本論文で提案している “XHC-3-seq-100msec” は，その計算時間が “HC-3-seq-1000ms” の 1/10 であるにもかかわらず，近似解の再利用によって，その近似性能が “HC-3-seq-1000ms” をほとんどの場合で上回っている．

図1に，3つの手法 “HC-3-para”, “AHC-3-para”, および “XHC-3-para” における，時間変化に伴う近似性能の変化を示す．ここでは，各入札分散ごとの違いがわかるように，入札分散ごとに分けて値をプロットしている．手順1による繰り返し入札更新では，1秒ごとの区間が各オークション問題に対応するため，手順1を行う途中で「総落札額の理論最大値」も入札の更新毎に変化する．そこで，図1では，各区間内での勝者近似性能の時間的な変化を相互に比較可能とするために，各区間内で “HC-3-para-1000msec” で得られた近似結果を1として，値を正規化してある．たとえば，2100msec 時点での値が 0.95 であったとすると，その意味は，2000msec から 3000msec までの期間に “HC-3-para-1000msec” で得られた近似結果 (総落札額) の 95 パーセントの合計落札額を持つ勝者を，その入札集合に対する近似計算の開始から 100msec 後に発見することができたことを示している．

入札分散が L7 のときには，“AHC-3” および “XHC-3” の結果が，“HC-3” と比較して良くなっている．しかしながら，他の入札分散 (L3,L4,L6) では，最終区間での近似結果を除けば，“AHC-3” の結果は “HC-3” の結果に大きく劣る場合がほとんどである．これは，本論文で示した手順1による入札集合の変化により，“AHC-3” が，近似勝者の「適切でない再利用」を行う場合が多くなったため，山登り方による解の改善のための初期値が非常に悪い点に定められたことによると考えられる．これに対して，“XHC-3” では，“AHC-3” に見られるような解の改悪の様子は見られない．“XHC-3” は，入札分散 L7 では “AHC-3” と同程度の性能向上を得られながら，他の入札分散でも少なくとも “HC-3” と同程度の性能を維持していることから，“XHC-3” の手法は，近似解の再利用を適切に行いながら，性能の低下を避けるという動作が効果的に動作していることがわかる．

5 関連研究

組み合わせオークションの勝者決定問題に対しては，CASS [4] を含め，その最良解を高速に探索するためのアルゴリズムが多く検討されてきている [2]．これらの最良解探索アルゴリズムの中には，近似解探索アルゴリズムとしても転用可能な物がある．CASS は，最良解を求めるためのアルゴリズムであるが，最良解を求める過程では高速な近似解探索アルゴリズムとしても利用できる．文献 [11] および [17] において，Casanova アルゴリズムおよび Zurel らの手法は，

†1 入札分散 L2 については，すでに “greedy-3” や “HC(c=0.5)-100ms” 程度の短時間の近似で十分に高い結果が得られており，入札分散の性質から “AHC-3” と “XHC-3” における近似性能もほぼ同様の値となり改善や改悪の余地がほとんどないことが明らかであるため，本論文ではこれらの値の計測を省略している．

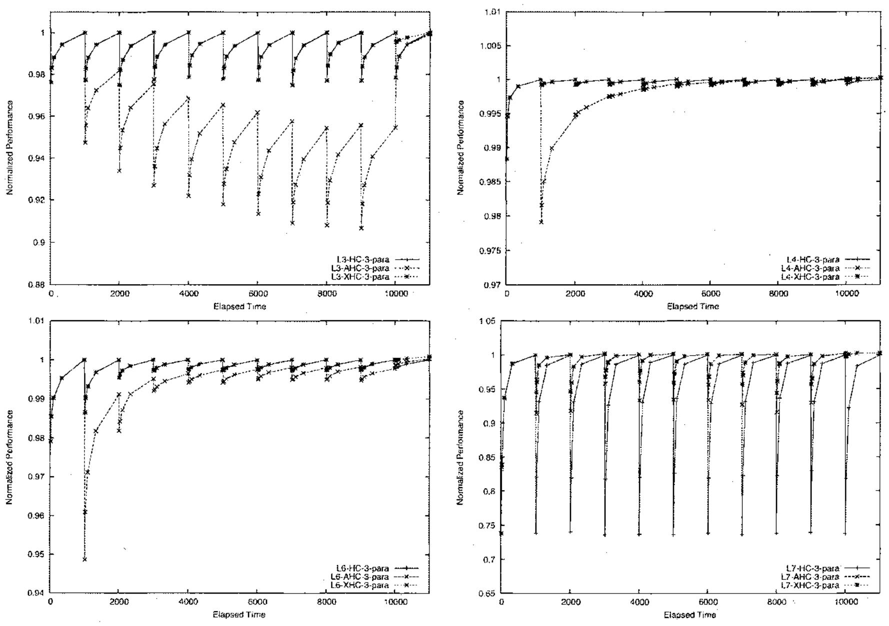

図 1　手順 1 による繰り返し入札更新時における近似性能の時間的変化

CASS と比較して，近似解探索アルゴリズムとしては非常に高い近似性能を発揮することが報告されている．したがって，本論文では，Casanova アルゴリズムおよび Zurel らの手法を比較対象にすれば十分であると判断し，本論文での比較実験では CASS を比較対象から省略した．

Guo らは，文献[10] で，局所探索に基づく手法で，組み合わせオークションの勝者決定の近似手法を提案している．Guo らの手法は，我々の提案する手法と Casanova アルゴリズムのちょうど中間的なものとなっている．文献[10] で示された比較実験結果では，Guo らの手法は Casanova アルゴリズムよりも 30%程度高い解を半分程度の時間で得られると言及されている．しかし，文献[10] で示された実験条件は本論文で扱ったものよりも問題の大きさが一桁以上小さく，なおかつ実験試行回数も各条件に対してわずか 10 回であるため，同文献中で示された結果の信頼性には疑問がある．また，本論文中の比較実験で示した結果の通り，Casanova アルゴリズム自体が，入札数がある程度多い場合には短時間での近似の性能がそれほど高くないため，文献[10] で示された結果はそれほど驚くべき結果ではないと考えられる．Casanova アルゴリズムを介した本研究との間接的な比較結果から，我々が従来提案している手法のほうがより短時間で同程度の近似解を探索できると考えられる．Guo らの手法には，本論文で述べた近似勝者の再利用手法は含まれていない．

CPLEX は，商用のソフトウェアであり，非常に高速な線形計画法のソルバーとして知られている．ところが，組み合わせオークションの勝者決定の近似に用いる場合には，文献[17] で，CPLEX と比較して，Zurel らの手法が少なくとも 10 から 100 倍程度は高速であることが言及されている．また，文献[10] でも，Guo らの手法は，CPLEX と比較して，1/40 から 1/80 の計算時間で同等以上の近似解を得られることが報告されている．これらの報告結果との間接的な比較から，我々の従来提案している手法は，CPLEX と比較しても十分に高速であると判断できる．

これらの関連研究は，基本的に offline アルゴリズムなので，本論文で提案するような，部分的な近似解

の再利用は行われない．

6 おわりに

本論文では，組み合わせオークションの近似的勝者決定における，部分的な近似解の再利用により，オークションへの入札の変化に対しての再計算を効率化する手法を提案し，その特性を解析した．現実問題として，オークションへの入札の変化が，burst 的に起きるのか，定常的に起きるのか，その頻度や変化の入札全体に対する割合がどの程度かは，対象とするアプリケーションによって異なる．入札の変化に対するアルゴリズムのロバスト性の評価方法の洗練は，今後の課題である．

ユビキタス環境での資源配分問題では，配分の効率性以外に，より多くの人が利用できる，特定の人が資源を占有し続けないなど，長期的なスパンでの割り当ての公平性の考慮が必要となる．本提案手法のこれらの目的への貢献の可能性の検討は，今後の課題である．

参考文献

[1] Cramton, P., Shoham, Y., and Steinberg, R.: *Combinatorial Auctions*, The MIT Press, 2005.

[2] de Vries, S. and Vohra, R. V.: Combinatorial Auctions: A Survey, *International Transactions in Operational Research*, Vol. 15,No. 3(2003), pp. 284–309.

[3] Epstein, R., Henriquez, L., Catalan, J., Weintraub, G. Y., Martinez, C., and Espejo, F.: A Combinatorial Auction Improves School Meals in Chile : A Case of OR in Developing Counteries, *International Transactions in Operational Research*, Vol. 11(2004), pp. 593–612.

[4] Fujishima, Y., Leyton-Brown, K., and Shoham, Y.: Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approarches, *Proc. of the 16th International Joint Conference on Artificial Intelligence (IJCAI99)*, 1999, pp. 548–553.

[5] Fukuta, N. and Ito, T.: Periodical Resource Allocation Using Approximated Combinatorial Auctions, *Proc. of The 2007 WIC/IEEE/ACM International Conference on Intelligent Agent Technology(IAT2007)*, 2007. (to appear.).

[6] Fukuta, N. and Ito, T.: Short-Time Approximation on Combinatorial Auctions – A Comparison on Approximated Winner Determination Algorithms, *Proc. of The 3rd International Workshop on Data Engineering Issues in E-Commerce and Services(DEECS2007)*, June 2007, pp. 42–55.

[7] Fukuta, N. and Ito, T.: Towards Better Approximation of Winner Determination for Combinatorial Auctions with Large Number of Bids, *Proc. of The 2006 WIC/IEEE/ACM International Conference on Intelligent Agent Technology(IAT2006)*, 2006, pp. 618–621.

[8] Fukuta, N. and Ito, T.: Toward A Large Scale E-Market : A Greedy and Local Search based Winner Determination, *Proc. of The 20th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems(IEA/AIE2007)*, June 2007, pp. 354-363. (poster presentation).

[9] 福田直樹, 伊藤孝行: 組み合わせオークションにおける多数入札時での勝者決定の近似解法に関する一考察, 電子情報通信学会論文誌, Vol. J90-D,No. 9(2007).

[10] Guo, Y., Lim, A., Rodrigues, B., and Zhu, Y.: A Non-exact Approach and Experiment Studies on the Combinatorial Auction Problem, *Proc. of HICSS2005*, 2005.

[11] Hoos, H. H. and Boutilier, C.: Solving Combinatorial Auctions using Stochastic Local Search, *Proc. of the AAAI2000*, 2000.

[12] Lehmann, D., O'Callaghan, L. I., and Shoham, Y.: Truth revelation in rapid, approximately efficient combinatorial auctions, *Journal of the ACM*, Vol. 49(2002), pp. 577–602.

[13] Leyton-Brown, K., Pearson, M., and Shoham, Y.: Towards a Universal Test Suite for Combinatorial Auction Algorithms, *Proc. of EC 2000*, 2000.

[14] Sandholm, T.: An Algorithm for Optimal Winnr Determination in Combinatorial Auctions, *Proc. of the 16th International Joint Conference on Artificial Intelligence(IJCAI'99)*, 1999, pp. 542–547.

[15] Sandholm, T., Suri, S., Gilpin, A., and Levine, D.: Winner Determination in Combinatorial Auction Generalizations, *Proc. of the 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS02)*, 2002, pp. 69–76.

[16] Sandholm, T., Suri, S., Gilpin, A., and Levine, D.: CABOB: A Fast Optimal Algorithm for Winner Determination in Combinatorial Auctions, *Management Science*, Vol. 51,No. 3(2005), pp. 374–390.

[17] Zurel, E. and Nisan, N.: An efficient approximate allocation algorithm for combinatorial auctions, *Proc. of the Third ACM Conference on Electronic Commerce (EC2001)*, 2001, pp. 125–136.