オンラインナップサック問題に対するアルゴリズム

河瀬　康志

ナップサック問題は，容量をもつナップサックが一つと，価値と重さをもつ品物の集合が与えられたとき，重さの総和が容量を超えないという制約の下で，価値の総和を最大にする品物集合を求める最適化問題である．オンラインナップサック問題とは，品物が逐次的に与えられる状況におけるナップサック問題である．本稿では，オンラインナップサック問題に対するアルゴリズムを紹介し，その性能を競合比によって議論する．

キーワード：ナップサック問題，オンライン最適化，競合比解析

1. はじめに

ナップサック問題とは，与えられた価値と重さをもつ品物のうちいくつかを，容量の決まったナップサックに詰め込むとき，ナップサックに詰め込む品物の価値の和を最大化する問題である．この問題は最も基本的な組合せ最適化問題の一つであり，予算内で購入する物資の選択や，制限時間内に処理するプロジェクトの選択など，さまざまな実世界の問題に応用されている．本稿を通して，ナップサックの容量は 1 であるとする．また，入力に表れるどの品物も，価値と重さは非負であり，重さは 1 以下であると仮定する．たとえば，表 1 のような四つの品物が与えられたとしよう．

表 1　ナップサック問題の例

品物	価値	重さ
1	4	0.2
2	2	0.1
3	6	0.8
4	5	0.5

このとき，品物の選び方は $2^4 = 16$ 通りあるが，ナップサックの容量を満たす中で最適な選び方は $\{1, 2, 4\}$ であり，そのときの価値の合計は $4 + 2 + 5 = 11$ となる（重さの合計は $0.2 + 0.1 + 0.5 = 0.8$ であり，ナップサックの容量を超えない）．ナップサック問題は NP 困難であり，最適解を効率よく求めることは困難であるが，効率的に近似解を求めることは可能であることが知られている．

かわせ　やすし
東京大学大学院情報理工学系研究科
〒 113–8656　東京都文京区本郷 7–3–1
kawase@mist.i.u-tokyo.ac.jp

本稿では，選択可能な品物が逐次的に与えられる状況におけるナップサック問題である，**オンラインナップサック問題**を扱う．オンラインナップサック問題では，まず最初に容量が既知のナップサックが与えられ，その後品物が一つずつ与えられる．そして，新しい品物が与えられるたびに，そのアイテムをナップサックに入れるかどうかを選択する．選択は，将来に来る品物についての情報なしに行う必要があり，一度決定したことを後から変更することはできない．この問題でモデル化できるものとしては，予算内で購入する物資を選択する例において，購入可能な物資の候補が逐次的に来る状況で，新しい候補が来るたびに（次の候補が来る前に）購入するかどうかを決める問題がある．

具体的には，次のように逐次的な意思決定を行う．はじめに，容量 1 のナップサックが与えられるが，どのような品物が何個あるかという情報はなにもない状態から開始する．まず，価値が 4 で重さが 0.2 の品物が与えられたとすると，これをナップサックに入れるかどうかを決定する．たとえば，入れたとする．次に，価値が 2 で重さが 0.1 の品物が与えられ，これは入れなかったとする．そして，三つ目の品物として，価値が 6 で重さが 0.8 の品物が来たとし，これは入れたとする．さらに，四つ目の品物として，価値が 5 で重さが 0.5 の品物が来たとする．このとき，ナップサックに入っている品物の重さの合計はすでに $0.2 + 0.8 = 1$ となっているため，この品物を入れることはできない．もう品物は来ないとすると，最終的に得られた価値の合計は $4 + 6 = 10$ となる．実は今回与えられた四つの品物は，表 1 と全く同じものであり，はじめからすべての品物を知っていれば合計価値 11 を達成できる．

2. オンライン問題と競合比解析

オンラインナップサック問題のように，入力が少しずつ逐次的に与えられ，入力が与えられるごとに選択（出力）を行わなければならないような問題を**オンライン問題**といい，オンライン問題に対するアルゴリズムを**オンラインアルゴリズム**という．逆に，初めから入力をすべて知った状態で解く問題を**オフライン問題**といい，オフライン問題に対するアルゴリズムを**オフラインアルゴリズム**という．オンラインアルゴリズムは，未来の情報を知ることなく選択を行う必要があるため，十分な時間を使えたとしても最適な解を出力できるとは限らない．そこで，最適な解を効率よく求めることは考慮せず，十分な時間を使えるときにどの程度よい解を出力できるかについて考えよう．オンラインアルゴリズムの性能は，オフラインアルゴリズムで得られる最適解に対し，どの程度よい解を出力できるかという**競合比**で評価することができる[1]．

オンラインの最大化問題に対し，入力列 I に対するオンラインアルゴリズム ALG での利得を $\mathrm{ALG}(I)$，I に対するオフラインでの最適利得を $\mathrm{OPT}(I)$ と表すことにする．このとき，アルゴリズム ALG の入力列 I における競合比は，

$$\frac{\mathrm{OPT}(I)}{\mathrm{ALG}(I)}$$

によって定義される[2]．また，アルゴリズム ALG の競合比は，最悪の入力に対する競合比

$$\sup_I \frac{\mathrm{OPT}(I)}{\mathrm{ALG}(I)}$$

によって定義される．ただし，上限 (sup) において I は，対象としているオンライン問題におけるすべての可能な入力列を動くとする．この値は，（入力をすべて知っていた場合に達成できる）最適利得が，アルゴリズム ALG によって得られる利得の何倍で抑えられるかを表す．さらに，オンライン問題の競合比は，最良のオンラインアルゴリズムの競合比

$$\inf_{\mathrm{ALG}} \sup_I \frac{\mathrm{OPT}(I)}{\mathrm{ALG}(I)}$$

によって定義される．競合比の値は少なくとも 1 であり，小さいほど嬉しい（最適利得に近い利得を得られる）ということになる．

[1] オンライン学習の文脈では**リグレット**がよく用いられるが，本稿では扱わない．

[2] ただし目的関数値は常に非負であると仮定し，$0/0 = 1$，$1/0 = \infty$ とみなす．

3. オンラインナップサック問題の不可能性

残念ながらオンラインナップサック問題に対する競合比は無限大になってしまうことが知られている．形式的には，任意の正数 M について，どんなオンラインアルゴリズム ALG を用いても，悪い入力列 I が存在し，競合比が M 以上となる $(\mathrm{OPT}(I)/\mathrm{ALG}(I) \geq M)$ ことを示すことができる．

定理 1（Marchetti-Spaccamela and Vercellis [1]）**.** オンラインナップサック問題の競合比は無限大である．

証明. 正数 M とオンラインアルゴリズムを固定して考える．最初の品物が価値も重さも 1 である入力列を考える．もしアルゴリズムが最初の品物をナップサックに入れなかったならば，ここで入力が終わりの場合に競合比が無限大となる．一方，ナップサックに入れたとしても，次の品物が価値 M で重さが 1 だとすると，競合比は M となる． □

これにより，（競合比の意味で）有意義なオンラインアルゴリズムを設計するためには，適用したい状況に合わせた適切な仮定が必要ということになる．現実的な仮定としては，たとえば，以下のものが考えられる：

(i) 一度選んだ品物を後から除去可能，
(ii) 品物の価値が重さに比例，
(iii) 品物の価値が一定，
(iv) 品物の重さが一定，
(v) 入力される品物数が事前にわかる，
(vi) 品物がある確率分布に従って与えられる．

次節以降では (i) の**除去可能**設定について，何ができて何ができないかを概観する．(ii)，(iii)，(iv) については，どれも単独の仮定では競合比が無限大となってしまうことを示すことができる．ただし，二つを同時に仮定すれば，入力される品物の価値と重さが一意に決まるため，ナップサックに入る限り貪欲に入れるだけで最適解を得られる（競合比 1）．(v) の仮定については，入力される品物数がとても多いかもしれず，価値が 0 であるような品物も存在するかもしれないため，あまり（単独では）意味がない．

(vi) の確率的入力モデルに関して，いくつかの確率モデルが提案されている．最も単純なものとして，各品物の重さと価値が何らかの独立同一分布に従うというものがある [1, 2]．また，そのほかにも，独立で非同一な分布に従う場合 [3] や，品物集合は決定的だが来る順序がランダムである場合 [4] についてなどの研究

がされている．

4. 除去可能設定

以降では，一度ナップサックに入れた品物を後から除去できる除去可能設定を扱う．除去可能設定でのオンラインナップサック問題を**除去可能オンラインナップサック問題**と呼ぶことにする．ナップサックをもっている人が移動していて，新しい品物を見つけるたびに，うまくナップサックの中の品物を詰め替えて（一部捨てて），最終的になるべく良い品物集合を保持しようとしている状況を想像するとよい．ここで，一度入れないと決めた品物や，捨ててしまった品物を後から取りに戻ることはできないと仮定する．このような状況は，予算内で購入する物資を選択する例でいうと，購入を後から無料[3]でキャンセル（返品）できる場合に対応する．また，除去可能オンラインナップサック問題に対するアルゴリズムを応用することで，予算制約付き安定マッチング問題に対して近似的に安定なマッチングを計算することができる [6, 7].

残念ながら除去可能設定においても，追加の仮定なしには競合比が無限大になってしまうことが知られている．

定理 2（Iwama and Zhang [8]）**.** 除去可能オンラインナップサック問題の競合比は無限大である．

証明. 正整数 M とオンラインアルゴリズムを固定して考える．価値も重さも 1 である品物が最初に与えられ，その後，価値 $1/M$ で重さ $1/M^2$ の品物が何個か与えられる入力列について解析する．アルゴリズムが最初に与えられた品物を保持し続け，その後に品物が M^2 個与えられた後も保持していた場合，アルゴリズムは最初の品物のみしか得ることができず，利得は 1 である．それに対し，オフラインでの最適解は最初の品物以外の M^2 個を選ぶものであり，利得は $M^2 \cdot 1/M = M$ となる．よって，この場合の競合比は M である．一方，アルゴリズムが最初の品物をどこかで除去した場合，除去した時点以降には品物が来ないような入力列では，利得を高々 $1/M$ しか得られない．それに対し，最初の品物だけで利得 1 を得ることができるので，競合比は少なくとも M となる．よって，競合比は M 以上となり，M はいくらでも大きい数を選んでくることができるため，競合比は無限大になる． □

[3] キャンセルコストを支払うことで除去ができる問題についても**買い戻し問題**という名前で研究がされている [5].

しかしながら，除去可能であることに加えて，入力される品物に仮定をおいた状況であれば，定数競合比を得ることができる．以下では，入力される品物について，重さが一定の場合，価値が一定の場合，価値が重さに比例する場合の三つについて考察する．

まず，品物の重さが一定の場合について考察する．品物の重さはすべて w であるとする．すると，ナップサックに入る品物の組合せは，$k := \lfloor 1/w \rfloor$ 個以下の任意の組合せとなる（$w = 0$ の場合は $k = \infty$）．新しい品物がやってくるごとに，現在保持している品物集合と新しい品物を合わせた集合から，品物の価値が高いものを優先して上位 k 個の品物を保持して残りを捨てる貪欲アルゴリズムを考える．このアルゴリズムは常に最適解を出力し，次の定理が成立する．

定理 3. 品物の重さが一定の場合，除去可能オンラインナップサック問題の競合比は 1 である．

続いて，品物の価値が一定の場合について解析する．この場合は，ナップサックに詰め込む品物の数を最大化することが目的となる．沢山の品物を詰め込むには，重さが軽い品物を優先して保持すればよいので，重さが軽い品物を優先する貪欲アルゴリズムを考える．すなわち，新しい品物がやってくるごとに，現在保持している品物集合と新しい品物を合わせた集合を考え，重さが軽いものを優先して，なるべく沢山の品物を保持する．別の言い方をすると，ナップサックにすべての品物が入らない場合には，最も重い品物を諦めるアルゴリズムということになる．このアルゴリズムは，最終的に最適解を得ることができる．これは，最適解に含まれる品物はどの時点でも除去されないことから確認できる．ただし，同じ重さの品物が複数存在する場合には最適解に含まれる品物を除去してしまう可能性があるが，その場合も同じ重さである別の品物が代わりに選択される．よって，次の定理が成立する．

定理 4. 品物の価値が一定の場合，除去可能オンラインナップサック問題の競合比は 1 である．

最後に，価値が重さに比例する場合について考察する．単位重さあたりの価値は品物の**密度**と呼ばれ，この設定は密度一定設定だということもできる．説明を簡単にするため，価値と重さが等しい（密度 1）と仮定して議論を進める．

この設定では，価値の高い品物を優先する貪欲アルゴリズムを用いることで競合比 2 を達成できる[4]．もう少し工夫の余地もあり，中くらいの価値の品物については価値の低いものを優先することで競合比を改善できる．直感的には，価値が 0.51 と 0.52 である二つの品物が

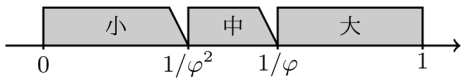

図 1　品物の分類

あるときには，価値 0.51 の品物を残す方が，次に価値 0.49 の品物が来るかもしれないので良さそうだということである．実際，最適なアルゴリズムを用いれば，競合比は黄金数 $\varphi = \frac{1+\sqrt{5}}{2} \approx 1.618$ となることを示すことができる．黄金数 φ について，$1/\varphi = \frac{\sqrt{5}-1}{2} \approx 0.618$ であり，$(1/\varphi)+(1/\varphi^2) = 1$ を満たすことに注意する．

定理 5 (Iwama and Taketomi [9])**.**　品物の密度が一定の場合，除去可能オンラインナップサック問題の競合比は φ である．

証明.　品物を大・中・小の三種類に分けることによりアルゴリズムを設計する．具体的には，

- 価値が $1/\varphi$ 以上の品物を**大**，
- 価値が $1/\varphi^2$ 以上 $1/\varphi$ 未満の品物を**中**，
- 価値が $1/\varphi^2$ 未満の品物を**小**

とする（図 1）．

このとき，現在保持している品物と新しく来た品物に対し，以下のような優先順位で選択する貪欲アルゴリズムを考える：

1. 大品物について価値が高い品物を優先，
2. 中品物について価値が低い品物を優先，
3. 小品物について価値が高い品物を優先．

ただし，一度 $1/\varphi$ 以上の利得を得たら，その後の入力は無視し現状の品物集合を保持し続けるとする．密度一定の場合の最適利得は，どのような入力列についても 1 以下であるので，一度 $1/\varphi$ 以上の利得を得られれば，競合比は φ 以下であることを保証できる．以後，どのような入力列に対しても，このアルゴリズムの競合比は φ 以下であることを示す．

大品物は最も優先されるので，入力に一つでも大品物が存在すれば，必ず $1/\varphi$ 以上の利得を得られる．そこで，大品物が存在しないような入力のみを考える．また，中品物の価値が低いものを優先していて，二つの中品物の価値の和は $2/\varphi^2 > 1/\varphi$ であることから，入力に含まれる中品物の二つを同時にナップサックに入れられるなら必ず $1/\varphi$ 以上の利得を得られる．よって，入力に含まれるどの二つの中品物も，同時にはナップサックに入れられないと仮定する．さらに，アルゴリズムが途中で小品物を捨てたとすると，その時点で $1-(1-1/\varphi) = 1/\varphi$ 以上の利得を得られていることがわかるので，アルゴリズムは小品物すべてを保持すると仮定する．

上記のすべての仮定を満たす入力列において，入力に表れる中品物の価値の多重集合を P，小品物の価値の多重集合を Q と表記する．中品物が存在しなければ，アルゴリズムはすべての品物を保持しているので，競合比は 1 となる．一方，中品物が存在する場合，競合比は高々

$$\frac{\max_{v\in P} v + \sum_{v\in Q} v}{\min_{v\in P} v + \sum_{v\in Q} v} \le \frac{\max_{v\in P} v}{\min_{v\in P} v} < \frac{1/\varphi}{1/\varphi^2} = \varphi$$

となる．以上より，競合比が φ 以下であることを確認できた．

最後に，任意の整数 ϵ (≤ 1) について，競合比が $\varphi-\epsilon$ 以下のアルゴリズムは存在しないことを示す．最初の二つの品物の価値が $1/\varphi^2$ と $1/\varphi+\epsilon/\varphi^2$ であるような入力列を考える．$1/\varphi+1/\varphi^2 = 1$ であることからこの二つの品物は同時にナップサックに入れることはできず，アルゴリズムは価値 $1/\varphi^2$ の品物を保持するものと保持しないものに分類することができる．前者のアルゴリズムについては，二つの品物で入力が終わりだとすれば，アルゴリズムの利得は $1/\varphi^2$ で最適利得は $1/\varphi+\epsilon/\varphi^2$ であることから，競合比は $\frac{1/\varphi+\epsilon/\varphi^2}{1/\varphi^2} > \varphi-\epsilon$ となる．また，後者のアルゴリズムについては，3 番目の品物として価値 $1/\varphi$ のものが来て終了する入力列を考えれば，アルゴリズムによる利得は $1/\varphi+\epsilon/\varphi^2$ 以下であり，最適な利得は 1 であることから，競合比は $\frac{1}{1/\varphi+\epsilon/\varphi^2} > \varphi-\epsilon$ 以上となる．　□

5.　資源拡大モデルとバッファモデル

前節では，除去可能設定においても入力される品物に何の仮定もない場合については，競合比は無限大であり意味のある評価ができないことをみた．そこで，評価指標としてもっと大まかなものを用いて評価するという方法を考える．そのような方策の一つとして，**資源拡大モデル**というものが提案されている [10]．これは，オンラインアルゴリズムが使える資源を増やすことでハンデをつけ，競合比解析を行うというものである．具体的には，

- ページング問題においてキャッシュサイズを拡大
- k サーバ問題においてサーバ数を増やす

[4] ちなみに，価値が低い（重さが軽い）品物を優先する貪欲アルゴリズムでは，競合比が無限大になってしまう．また，各ステップで容量制約を満たす最適な組合せを保持するような貪欲アルゴリズムの競合比は 2 である．

- スケジューリング問題においてマシンの処理速度や台数を増やす
- ビンパッキング問題においてビンの容量を増やす

といった使い方がされている．

オンラインナップサック問題においては，オンラインアルゴリズムが使うナップサックの容量を R に増やすということが自然である．すなわち資源拡大モデルにおける競合比を，容量 1 のナップサックを用いた場合のオフラインでの最適利得と，容量 R のナップサックを用いるオンラインアルゴリズムの利得の比によって定義する．予算内で購入する物資を選択する例でいうと，基準となる予算が 1 なのに対し，購入金額は R まで許す状況だと解釈することもできる．たとえば，$R = 1.5$ で表 1 の品物が逐次的に与えられる状況を考える．すると，最初の三つの品物は重さの合計が 1.1 なのですべて保持することができる．四つ目の品物を加えると重さの合計が $1.6 > R$ となるので，いくつかの品物を捨てる必要がある．品物 2 を捨てて $\{1, 3, 4\}$ の品物を保持したとすると，四つで入力は終わりなので，最終的な利得は 15 ということなる．この例における容量 1 での最適利得は 11 なので，競合比は 11/15 ということになる．このように $R > 1$ の場合は，アルゴリズムと入力列のペアによっては競合比が 1 より小さくなることもある．ただし，空入力における競合比はどのアルゴリズムでも 1 なので，アルゴリズムの競合比は必ず 1 以上となる．

$R = 1$ の場合は，通常の競合比と同じである．$R > 1$ の場合，除去可能オンラインナップサック問題の競合比は定数になる．特に $R \geq 2$ の場合は，基準である容量 1 での最適値より悪くない目的関数をもつ解を必ず得ることができる．

定理 6 (Iwama and Zhang [8])**.** 資源拡大モデル $(R > 1)$ における除去可能オンラインナップサック問題の競合比は $\max\{1, 1/(R-1)\}$ である．

証明. 密度の高い品物を優先する貪欲アルゴリズム（各ラウンドでは，現在保持している品物と新たに与えられた品物を合わせて密度の降順に並べ，ナップサックに入る限り品物を入れる）によって競合比 $\max\{1, 1/(R-1)\}$ を達成できることを示す．入力される品物の重さの合計が R 以下の場合，すべての品物を保持することができるので競合比は 1（以下）となる．また，入力される品物の重さの合計が R より大きい場合は，密度が高いものから重さ $R-1$ 以上を得ることができるので，競合比は $1/(R-1)$ 以下であることがわかる．よって，この貪欲アルゴリズムの競合比は $\max\{1, 1/(R-1)\}$ 以下である．

このアルゴリズムの最適性（どんなアルゴリズムを用いても競合比は $\max\{1, 1/(R-1)\}$ よりもよくできないこと）の証明については詳細を省略するが，以下のような入力列を用いることで証明できる：

$$(1, 1), (\epsilon, \epsilon^3), (\epsilon, 2\epsilon^3), \ldots, (\epsilon, k\epsilon^3).$$

ただし，品物を重さと価値のペアで表していて，ϵ は十分小さい正の数だとする． □

通常の競合比よりも大まかであるが資源拡大モデルよりも精密な評価指標として，**バッファモデル**というものも考えられている [11]．このモデルでは，オンラインアルゴリズムは容量 R (≥ 1) の「バッファ」に品物を保持するが，入力が終わった後には容量 1 のナップサックに品物を詰め替える．予算内で購入する物資を選択する例において，最終的に使える金額は 1 であるが，一時的には R まで使うことを許す状況だと解釈することもできる．たとえば，$R = 1.5$ で表 1 の品物が逐次的に与えられ，四つの品物すべてが与えられた後にアルゴリズムが品物 $\{1, 3, 4\}$ を保持していたとする．これらの品物を容量 1 のナップサックに移す必要があり，最終的には品物 $\{1, 3\}$ を残して利得は 10，競合比は 11/10 ということになる．

バッファモデルにおける除去可能オンラインナップサック問題では，R が 2 以下の場合は密度の高い品物を優先する貪欲アルゴリズムが最適であり，その競合比は $\max\{2, 1/(R-1)\}$ となる．

定理 7 (Han et al. [11])**.** バッファモデル $(2 \geq R > 1)$ における除去可能オンラインナップサック問題の競合比は $\max\{2, 1/(R-1)\}$ である．

また，R がどんなに大きくても競合比は 1 より大きいが，R が無限大に近づくとき競合比は 1 に漸近的に近づく．

定理 8 (Han et al. [11])**.** バッファモデル $(R > 1)$ における除去可能オンラインナップサック問題の競合比は $1 + \tilde{\Theta}(1/R)$ である．

以降では，競合比 $1+O(\log R/R)$ のアルゴリズムを示す．R は十分に大きいと仮定し，正数 ϵ を $O(\log R/R)$ となるように設定する（正確には $\log_{1+\epsilon}(1/\epsilon) = \lfloor (R-4)/2 \rfloor$ を満たす値）．

まず，品物を，重さが ϵ 以下であるかどうかによって，S 品物と M 品物の 2 種類に分類する．さらに M 品物について，価値が $(1+\epsilon)^k$ 以上 $(1+\epsilon)^{k+1}$ 未満の

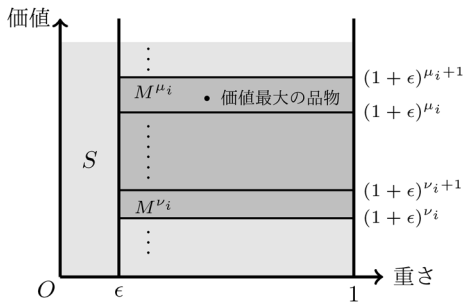

図 2　バッファモデルに対するアルゴリズムでの品物の分類

物を M^k 品物と呼ぶことにする．すなわち，重さが ϵ より大きく価値が v である品物は，$M^{\lfloor \log_{1+\epsilon} v \rfloor}$ 品物ということになる．i 番目の品物が与えられたとき，次のように保持する品物を決めるアルゴリズムを考える．

- S 品物については，重さの合計を 3 以下に保ちつつ，密度の高い物を優先する貪欲アルゴリズムにより品物を選択する，
- $k \geq \nu_i$ について，M^k 品物は，重さの合計を 1 以下に保ちつつ，重さの昇順についての貪欲アルゴリズムにより品物を選択する，
- $k < \nu_i$ について，M^k 品物はすべて捨てる．

ここで，i 番目の品物までで価値最大の M 品物は M^{μ_i} 品物であるとし，$\nu_i = \mu_i - 2\log_{1+\epsilon}(1/\epsilon)$ $(= \mu_i - 2\lfloor (R-4)/2 \rfloor)$ とする（図 2）．バッファに入っている M 品物の重さの和は $\mu_i - \nu_i + 1 = 1 + 2\lfloor (R-4)/2 \rfloor \leq R-3$ 以下であるので，このアルゴリズムは容量 R のバッファで実行可能である．

全部で n 個の品物が入力されたとすると，最終的にバッファに入っている品物は

- S 品物については，容量 3 のナップサックを用いて，密度の高い物を優先する貪欲アルゴリズムにより品物を保持した解
- 各 $k \geq \nu_n$ について M^k 品物は，容量 1 のナップサックを用いて，重さの小さいものを優先する貪欲アルゴリズムにより品物を保持した解

となる．これは，ν_i が i について単調非減少であることから確認できる．

オフラインでの最適解を OPT とし，上記のアルゴリズムにより最後の品物が来た後にバッファに入っている品物集合を B_n と表記する．このとき，容量 1 のナップサックに対して実行可能な品物集合 $B^* \subseteq B_n$ を以下のように構成する[5]:

- 各 $k \geq \nu_n$ について，$B_n \cap M^k$ から重さの小さいものを優先して $|\mathrm{OPT} \cap M^k|$ 個をナップサックに入れる，
- $B_n \cap S$ について密度を優先した貪欲アルゴリズムでナップサックの残りを埋める．

ここで，各 $k \geq \nu_n$ について，$|B_n \cap M^k|$ が $|\mathrm{OPT} \cap M^k|$ 以上であることは，重さの小さいものを優先する貪欲アルゴリズムにより品物を保持していることから確認できる．

最後に，OPT の価値は B^* の価値の $1 + O(\epsilon)$ 倍以下であることの概要を説明する．M^k 品物について価値の高い物と低い物の比は高々 $(1+\epsilon)$ 倍であることから，各 $k \geq \nu_n$ に対し，$\mathrm{OPT} \cap M^k$ の価値は $B^* \cap M^k$ の価値の高々 $(1+\epsilon)$ 倍であることがわかる．また，$\mathrm{OPT} \cap \bigcup_{k<\nu_n} M^k$ の価値はとても低い．なぜならば，どの M 品物も ϵ 以上の重さをもつことから要素数は $1/\epsilon$ 以下であり，各品物の価値は最大価値の品物の ϵ^2 倍以下であることから，総価値は OPT の価値の ϵ 倍以下である．さらに，十分にたくさんの密度の高い S 品物を保持していることから，$\mathrm{OPT} \cap S$ の価値と $B^* \cap S$ の価値の差は，高々 OPT の価値の $O(\epsilon)$ 倍であることを示すことができる．これらにより，OPT の価値は B^* の価値の高々 $(1+O(\epsilon))$ 倍であることが示せ，このアルゴリズムの競合比は $1 + O(\epsilon) = 1 + O(\log R/R)$ 以下であることがいえる．

6. おわりに

本稿ではさまざまなオンラインナップサック問題のモデルとアルゴリズムを紹介した．ナップサック制約は最も基本的な制約であり，多くの問題に現れるため，オンラインナップサック問題の解析はオンライン最適化の基盤となるといえる．オンラインナップサック問題における競合比解析では，一般的な状況を標準的な指標で考えるとうまく評価できないため，状況に応じた適切な仮定と適切な評価指標を選択する必要がある．

本稿では触れなかったが，乱択アルゴリズムを用いることで競合比を改善する研究も行われている．また，目標となる解の品質達成するためには，どの程度の「助言」が必要であるかを解析する**助言複雑性**についての解析も行われている．これらの詳細については Komm による本 [12] の 6 章を参照することをおすすめする．

参考文献

[1] A. Marchetti-Spaccamela and C. Vercellis, “Stochastic on-line knapsack problems,” *Mathematical Pro-*

[5] この解 B^* は解析のために構成したものであり，実際には最適な組合せを取るのがよい（計算時間についての制限がないので可能である）．

gramming, **68**, pp. 73–104, 1995.
[2] G. S. Lueker, "Average-case analysis of off-line and on-line knapsack problems," *Journal of Algorithms*, **29**, pp. 277–305, 1998.
[3] P. Dutting, M. Feldman, T. Kesselheim and B. Lucier, "Prophet inequalities made easy: Stochastic optimization by pricing non-stochastic inputs," In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, pp. 540–551, 2017.
[4] M. Babaioff, N. Immorlica, D. Kempe and R. Kleinberg, "A knapsack secretary problem with applications," In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM'07)*, pp. 16–28, 2007.
[5] M. Babaioff, J. D. Hartline and R. D. Kleinberg, "Selling banner ads: Online algorithms with buyback," In *Proceedings of the 4th Workshop on Ad Auctions*, 2008.
[6] Y. Kawase and A. Iwasaki, "Approximately stable matchings with budget constraints," In *Proceedings of the 32nd AAI Conference on Artificial Intelligence (AAAI'18)*, pp. 1113–1120, 2018.
[7] Y. Kawase and A. Iwasaki, "Approximately stable matchings with general constraints," In *Proceedings of the 19th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'20)*, pp. 602–610, 2020.
[8] K. Iwama and G. Zhang, "Optimal resource augmentations for online knapsack," In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM'07)*, pp. 180–188, 2007.
[9] K. Iwama and S. Taketomi, "Removable online knapsack problems," In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP'02)*, pp. 293–305, 2002.
[10] B. Kalyanasundaram and K. Pruhs, "Speed is as powerful as clairvoyance," *Journal of the ACM*, **47**, pp. 617–643, 2000.
[11] X. Han, Y. Kawase, K. Makino and H. Yokomaku, "Online knapsack problems with a resource buffer," In *Proceedings of the 30th International Symposium on Algorithms and Computation (ISAAC'19)*, pp. 28:1–28:14, 2019.
[12] D. Komm, *An Introduction to Online Computation*, Springer International Publishing, 2016.