Vol.2011-AL-137 No.9
2011/11/18

情報処理学会研究報告
IPSJ SIG Technical Report

資源増加を許した OVSF 符号割当問題に対する $(1+\varepsilon)$-競合アルゴリズム

朝 廣 雄 一[†1]　上米良 謙太[†2]　宮 野 英 次[†2]

著者らは 8) で，資源増加を許したオンライン OVSF 符号割当問題に対して，$2\lg^* h$ 個の符号木を用いた 2-競合アルゴリズムを提案した．ここで h は OVSF 符号木の高さである．本稿では，このアルゴリズムを拡張し，任意の $\alpha > 1$ に対して，$(1+\lceil\alpha\rceil)\lg^* h$ 個の符号木を用いた $(1+1/\alpha)$-競合アルゴリズムを提案する．

Resource Augmented $(1+\varepsilon)$-competitive Algorithm for Online OVSF Code Assignment Problem

Yuichi Asahiro ,[†1] Kenta Kanmera [†2]
and Eiji Miyano[†2]

In 8), for online OVSF code assignment problem, the authors propose a 2-competitive algorithm with $2\lg^* h$ trees, where h is the height of the OVSF code tree. In this paper, for any $\alpha > 1$, a $(1+1/\alpha)$-competitive algorithm with $(1+\lceil\alpha\rceil)\lg^* h$ trees is proposed.

1. はじめに

マルチメディアサービスにおいて，通信内容に応じた異なる品質でデータを送受信したい際に，直交可変拡散率（Orthogonal Variable Spreading Factor，略して OVSF）符号が用いられている．OVSF 符号を利用したシステム（簡単のために OVSF システムと呼ぶ．例えば W-CDMA: Wideband Code-DivisionMultiple-Acces 方式）においては，それぞれの通信に割り当てられる符号の長さや拡散率が異なり，高いデータ通信率は，低い拡散率での通信により実現される．OVSF 符号は再帰的に生成されるが，それは高さ h の完全 2 分木（OVSF 符号木と呼ぶ）により表現され，OVSF 符号木の各頂点が，正確に 1 つの符号に対応するので，以下では頂点と符号を区別せずに説明を行う．OVSF 符号木のレベル h にある根頂点は，符号 1 に対応する．ある頂点が符号 c を持つとき，その左右の子はそれぞれ符号 cc と $c\overline{c}$ を持つというように，再帰的に符号が生成される．ここで $\overline{c}$ は c を反転したものである．これにより，OVSF 符号木のレベル 0 にある，2^h 個の葉は，それぞれ長さ が 2^h ビットの符号を持つことになる．

OVSF システムにおいては，新しい通信要求に対して OVSF 符号木の一つの頂点（符号）を割り当てる．その際，新しく割り当てられた符号と，それ以前に割り当てられ利用されている符号が**直交**，すなわち根から葉に至るどの道においても要求が割り当てられている頂点は高々1 個しかないという条件を満たす必要がある．この要求をいかに割りあてるかという問題をオンライン問題として定式化したものが，OVSF 符号割当問題である[6)]．この問題における入力は，**割当**と**解放**の要求（**リクエスト**）の列である．割当リクエストには，レベルを表す整数 ℓ $(0 \le \ell \le h)$ がついており，OVSF 符号木のレベル ℓ にある頂点を割り当てる必要がある．この割当処理のコストは 1 と考える．一方，解放リクエストに対しては，そのリクエストに対応する割当済の頂点を解放するが，この処理にはコストはかからないものとする．頂点にリクエストを割り当てる際には，直交条件を満たす必要があり，既にどこかの頂点が割り当てられているリクエストに別の頂点を再割当（移動）する必要があるかもしれない．この再割当処理のコストも 1 であると考える．以上により，オンラインアルゴリズムのコストは，割当と再割当の合計回数と定義される．本問題の目的は，割当と再割当の処理列の中でコストを最小化するようなものを探す（そのような処理を行うアルゴリズムを設計する）ことである．

競合比解析によりオンラインアルゴリズムのコストを評価する．オンラインアルゴリズム ALG の**競合比**は，ALG のコストと最適オフラインアルゴリズムのコストの比の最悪値と定義する．アルゴリズム ALG の競合比が σ の時，ALG は σ-競合であるという．高さ h の OVSF 符号木は帯域幅 2^h を持っており，レベル ℓ にある割当済の頂点は，帯域幅 2^ℓ を消費する．本稿では，割当済の頂点が消費する帯域幅の合計は，常に 2^h 以下であり，レベル h の割当リクエストは存在しないことを仮定する．よって（オンライン）アルゴリズムは，必要ならば既に頂点が割り当てられているリクエストへ別の頂点を再割当することも含め，

†1 九州産業大学情報科学部情報科学科
Department of Information Science, Kyushu Sangyo University
†2 九州工業大学情報工学研究院システム創成情報工学系
Department of Systems Design and Informatics, Kyushu Institute of Technology

情報処理学会研究報告
IPSJ SIG Technical Report

表 1 既知のオンラインアルゴリズム
Table 1 Known online algorithms.

競合比	符号木の数	参考文献
1	$(h+1)/2$	1)
$4/3+\delta$ $(\delta>0)$	$11/4+4/(3\delta)$	1)
2	$3h/8+2$	1)
4	2	6)
5	9/8	4)
6	1	3)
7	1	7)

全ての割当リクエストに対してどこかの頂点を割り当てねばならない．割当リクエストの分のコストは減らせないので，再割当処理の回数をできるだけ少なくすることが，アルゴリズム設計の方針となる．

オンライン OVSF 符号割当問題には，例えば，無線通信やメモリ管理など，様々な応用があり，活発に研究されている（関連研究のまとめとして 2) がある）．オンライン OVSF 符号割当問題においては，オフラインアルゴリズムは高さ h の符号木（主符号木と呼ぶ）を 1 個だけ利用できるのに対して，オンラインアルゴリズムは高さ h の符号木を $k(>1)$ 個，すなわち，主符号木に加え，高さ h の符号木をさらに $k-1$ 個余分に利用してよいというように資源増加を許す問題と，資源増加を許さない（主符号木のみでリクエストを処理する）問題設定の両方が考察されている．資源増加を許す問題設定においてもアルゴリズムの目標は相変わらずコスト，すなわち割当と再割当の合計回数の最小化である．容易に分かるように，多くの符号木を利用して良いなら，より良い競合比のアルゴリズムを設計することが可能である．既知のオンラインアルゴリズムの性能について表 1 にまとめる．競合比 6 と 7 のアルゴリズムは，資源を増加させない（符号木の数が 1 個の）設定におけるアルゴリズムである．なお，資源増加を許さない場合の競合比の下界としては，今のところ 2 が知られている[7]．資源増加を許す場合の競合比の下界として，再割当をしないどのようなオンラインアルゴリズム（すなわち 1-競合アルゴリズム）も $(h+1)/2$ 個以上の符号木を必要とすることも示されており[1]，この意味で 1) で提案された $(h+1)/2$ 個の木を用いた 1-競合アルゴリズムは最適である．

本稿では，資源増加を許したオンライン OVSF 符号割当問題に対して，8) で提案した 2-競合アルゴリズムを改良し，任意の $\alpha>1$ に対して $(1+\lceil\alpha\rceil)\lg^* h$ 木を用いることで競合比 $(1+1/\alpha)$ を実現するアルゴリズムを提案する．例えば $\alpha=3$ とすると，$4\lg^* h/3$ 木を用いた 4/3 競合アルゴリズムを得ることができる．また本結果により，1) で示された $(11/4+4/(3\delta))$ 個の符号木を用いた $(4/3+\delta)$-競合アルゴリズム（ただし $\delta>0$）とは別の，符号木数と競合比のトレードオフを与えたと考えることができる．

2. 諸 定 義

簡単のために，以下で用いる（部分）木は完全 2 分木とする．例えば，高さ $h-1$ である木の大きさは高さ h である木の大きさの 1/3 であるので，高さ $h-1$ の木を 3 個利用できる時は，高さ h の木を $1/2\times 3=3/2$ 個利用できると言うことにする．葉 v について，そのレベルは 0 とする．葉以外の頂点 v のレベル $l(v)$ は，v の子 u のレベル $l(u)$ に 1 を加えたものと考える．頂点 v の **利用可能帯域幅**（あるいは単に**帯域幅**） $abw(v)$ は $2^{l(v)}$ である．v がある割当リクエストに割り当てられており，まだ解放されていない時，v は**割当済**という．もし，根から葉に至る全ての道のうち，v を含むもの全てに割当済頂点がない場合，v は**自由**である．

オンラインアルゴリズムは **直交条件**を満たす必要がある．ここで直交条件とは，根から葉に至るどの道においても割当済の頂点は高々1 個であるという条件である．(部分）木の状態を，白，黒，灰の 3 色で表すことにする：(部分）木 T は，全ての頂点が自由である場合に白色であり，自由な頂点がない場合に黒色であり，白でも黒でもない場合に灰色である．(部分）木 T に対して，$A(T)$ は T 内の割当済頂点の集合を表す．(部分）木の集合 $\mathcal{S}$ に対して，$A(\mathcal{S})$ は $\mathcal{S}$ に属する木内の割当済頂点をすべて含む集合である．高さ h' である (部分）木 T の **利用可能帯域幅**（あるいは単に**帯域幅**）$abw(T)$ とは，T の根の帯域幅，すなわち $2^{h'}$ と考えられる．(部分）木の集合 $\mathcal{S}$ の利用可能帯域幅 $abw(\mathcal{S})$ は $\sum_{T\in\mathcal{S}} abw(T)$ と定義される．(部分）木 T 内にある割当済頂点 v の**消費帯域幅** $cbw(v)$ は，$2^{\ell(v)}$ である．よって（部分）木 T の消費帯域幅は $\sum_{v\in A(T)} cbw(v)$ と定義され，$cbw(A(T))$ や $cbw(T)$ と書くことにする．(部分）木の集合 $\mathcal{S}$ の消費帯域幅 $cbw(\mathcal{S})$ は $\sum_{T\in\mathcal{S}} cbw(T)$ である．

記号 $\lg n$ は底が 2 である n の対数を表すとする．すなわち $\lg n=\log_2 n$ である．$f^{(i)}(n)$ は，関数 f を n に対して i 回繰り返して適用したものを表す．例えば $\lg^{(3)} n=\lg\lg\lg n$ である．記号 $\lg^* n$ により，$\lg^* n=\min\{i\geq 0 \mid \lg^{(i)} n\leq 1\}$ を表す．主符号木の高さ h に対して，$\ell_0=h$，$\ell_i=\lceil\lg^{(i)} h\rceil$ と定義する．ここで i は $1\leq i\leq \lg^* h$ を満たす．すなわち，$\ell_1=\lceil\lg^{(1)} h\rceil$，$\ell_2=\lceil\lg^{(2)} h\rceil$，…，$\ell_{\lg^* h-1}=\lceil\lg^{(\lg^* h-1)} h\rceil(=2)$，$\ell_{\lg^* h}=\lceil\lg^{(\lg^* h)} h\rceil(=1)$ となる．高さ $h-\ell_i-1$ を持つ部分木を i-**部分木** と呼ぶ．

本稿では入力に対して以下の 2 つの仮定を設けるが，これらは既存研究 1)，4)，5) でも

同様に仮定されていたものである．最初の仮定は，最適オフラインアルゴリズムが消費するのは，高々，主符号木の帯域幅 2^h であるというもので，資源増加モデルにおいての通常の仮定である．また，2 個目の仮定は，それほど強い制限ではない： もしレベル h の割当リクエストが来たら，そのリクエストは帯域幅として 2^h を必要とするので，主符号木の全ての頂点が自由である必要がある，すなわち，それまでに割当てられた頂点すべてが解放されている必要がある．これにより，そのようなレベル h の割当リクエストのところで入力を分割して，個別に解くことができる．

仮定 1　割当済である頂点の消費帯域幅の合計は高々2^h である．

仮定 2　入力中には，レベル h の割当リクエストは存在しない．

3. アルゴリズム

本節では，任意の $\alpha > 1$ について，$(1+1/\alpha)$-競合であるアルゴリズム ALG を示す．基本的なアイディアは，以下である．(i) まず 0 から $h-1$ までのレベルを $\lg^* h$ 個のグループに分割する．(ii) 次に，これらのグループそれぞれに対して，ある高さの部分木をある個数ずつ用意する．ここで用意する部分木は，そのグループに属するレベルを持つ割当リクエスト間で共有され，割り当てが行なわれるが，一旦あるレベルのリクエストが割り当てられると，他のレベルのリクエストはその部分木には割り当てられない．ただし割り当てられていた全てのリクエストが解放された場合，その部分木は，その後，他のレベルのリクエストを割り当てても良い．これら部分木の高さや個数については後で詳しく説明するが，各グループで利用される部分木の利用可能帯域幅の合計は高々2×2^h であり，高さ h である 2 個の木を利用すると言える．(iii) 再割当の処理は，$\lceil\alpha\rceil$ 回の解放リクエスト毎に高々1 回しか行なわない．

3.1 部分木の準備

まず，$1 \le j \le \lg^* h - 1$ に対して，$L_j = \{i \mid h - \ell_{j-1} \le i \le h - \ell_j - 1\}$ とし，$L_{\lg^* h} = \{h-2,\ h-1\}$ とすることにより，レベルを $\lg^* h$ 個のグループに分ける．例えば，$h = 7$ とすると，$\ell_0 = 7$, $\ell_1 = \lceil \lg 7 \rceil = 3$, $\ell_2 = \lceil \lg \lg 7 \rceil = 2$ であるので，$L_1 = \{0,1,2,3\}$, $L_2 = \{4\}$, $L_3 = \{5,6\}$ となる．

$1 \le j \le \lg^* h - 1$ に対して，L_j に属するレベルのリクエストを処理するために，$\lceil\alpha\rceil(\ell_{j-1} - \ell_j) + 2^{\ell_j+1}$ 個の j-部分木を用意し，これらの集合を $\mathcal{T}_j$ と名付ける．この j-部分木の数は，$(\ell_{j-1} - \ell_j)$ が L_j に属するレベルの個数であり，この数は高々 2^{ℓ_j+1} であるとともに，2^{ℓ_j+1} 個の j-部分木の総帯域幅が 2^h （すなわち高さ h の木の帯域幅）であることに由来する．最後のグループ $L_{\lg^* h}$ に対しては，高さ h の木を 2 個用意し，それぞれ T_{h-1} と T_{h-2} と名付ける．ここで，高さ h の主符号木は，この T_{h-1} と T_{h-2} のいずれかとして利用されるものとする．以下の補題は，以上のように用意した（部分）木の総量に対する上界を与える．

補題 1　ALG のために用意した高さ h の木は高々$(1+\lceil\alpha\rceil)\lg^* h$ 個である．

証明.　$1 \le j \le \lg^* h - 1$ に対して，以下のように，$\ell_{j-1} \le 2^{\ell_j+1}$ が成り立つ．

$$\lg \ell_{j-1} = \lg\lceil \lg^{(j-1)} h\rceil \le \lg(2\lg^{(j-1)} h) = \lg^{(j)} h + 1 \le \lceil \lg^{(j)} h\rceil + 1 = \ell_j + 1$$

グループ L_j のために用意した $\mathcal{T}_j$ の利用可能帯域幅 $abw(\mathcal{T}_j)$ を求める．$\mathcal{T}_j$ には，$\lceil\alpha\rceil(\ell_{j-1}-\ell_j)+2^{\ell_j+1}$ 個の j-部分木が含まれており，それらの高さは $h-\ell_j-1$ であるので，$abw(\mathcal{T}_j)$ は以下の通り見積もれる．

$$\begin{aligned} abw(\mathcal{T}_j) &= (\lceil\alpha\rceil(\ell_{j-1}-\ell_j)+2^{\ell_j+1})\cdot 2^{h-\ell_j-1} < \lceil\alpha\rceil\cdot\ell_{j-1}\cdot 2^{h-\ell_j-1} + 2^h \\ &\le \lceil\alpha\rceil\cdot 2^{\ell_j+1}\cdot 2^{h-\ell_j-1} + 2^h \le (1+\lceil\alpha\rceil)\cdot 2^h. \end{aligned}$$

よって，L_1 から $L_{\lg^* h-1}$ までの $\lg^* h - 1$ 個のグループに対して，高さ h の木が 高々$(1+\lceil\alpha\rceil)$ 個ずつ用意されていると考えられる．グループ $L_{\lg^* h}$ については，高さ h の木を 2 個用意したので，アルゴリズム ALG のために用意した高さ h である木の個数は，高々$(\lg^* h - 1)(1+\lceil\alpha\rceil)+2 = (1+\lceil\alpha\rceil)\lg^* h - \lceil\alpha\rceil + 1 < (1+\lceil\alpha\rceil)\lg^* h$ である．　□

3.2 部分木へのラベル付け

アルゴリズム ALG は $\mathcal{T}_j$ に含まれる各 j-部分木に対して，その状態である，白色または灰色，黒色を示すためにラベルを付けたり消したりする．また，それらのラベルは，色と共に，その j-部分木に対してどのレベルのリクエストが割り当てられているかも示す．先にも述べたように，灰色や黒色の各 j-部分木には，それぞれある単独のレベルのリクエストのみしか割り当てされていない状態になる．最初は，$\mathcal{T}_j$ 中の各 j-部分木は白色であり，ラベルがついていない．ある時点での白色の j-部分木の数を w_j，排他的にレベル i のために使用されている黒色および灰色の j-部分木の数をそれぞれ b_i および g_i とする．それぞれの数とラベルは割当や解放リクエストへの処理実行後に更新される．j-部分木につけるラベルとして次の 2 種類を用いる．

ラベル $B_{i,k}$　：その j-部分木はレベル i のリクエストが割り当てられており，黒色である．また黒い j-部分木の中で k 番目のものである[⋆1]．

⋆1 黒い j-部分木の順序は任意で構わない．添字の k を導入する理由は，黒い j-部分木を単に区別するためである．

ラベル $G_{i,k}$ ：その j-部分木はレベル i のリクエストが割り当てられており，灰色である．添え字 k は，$\mathcal{T}_j$ の中でレベル i のために使用されている灰色の j-部分木の割り当て済み頂点数の降順になるように決められる．すなわち，$|A(G_{i,1})| \geq |A(G_{i,2})| \geq \cdots \geq |A(G_{i,g_i})| > 0$，つまり $cbw(G_{i,1}) \geq cbw(G_{i,2}) \geq \cdots \geq cbw(G_{i,g_i}) > 0$ が成り立つ．ただし，ラベル $G_{i,k}$ の添え字 k は，説明のために用いているだけである．アルゴリズムの中では単に灰色の j-部分木を整列するだけでよく，添え字 k 自体を管理する必要はない．整列操作は j-部分木のラベル更新の際に実行されるが，灰色の j-部分木を降順にしておくことは，リスト構造を利用することにより 1 回のラベル更新あたり定数時間で可能である．レベル i のために用いられる灰色の j-部分木のリストを D_i とし，この整列順序を更新する手続きを $\mathsf{Reorder}(D_i)$ と表す．

j-部分木 T のラベルと b_i，g_i，w_j の更新操作には 8 種類がある．アルゴリズム ALG の中では，j-部分木 T の更新操作を $\mathsf{Update}_\alpha(T)$ と表す．

- 白色の j-部分木 T が，レベル i の頂点をリクエストに割り当てることにより，黒色に変わった場合：T にラベル B_{i,b_i+1} をつけ，$w_j := w_j - 1$，$b_i := b_i + 1$ とする．
- 白色の j-部分木 T が，レベル i の頂点をリクエストに割り当てることにより，灰色に変わった場合：T にラベル G_{i,g_i+1} をつけ，$w_j := w_j - 1$，$g_i := g_i + 1$ とする．
- ラベル $B_{i,k}$ を持つ黒の j-部分木 T が白に変わった場合：ラベル $B_{i,k}$ を T から消す．もし $b_i \geq 2$ かつ $k \neq b_i$ ならば，j-部分木 B_{i,b_i} のラベルを $B_{i,k}$ に変更する．そして $b_i := b_i - 1$，$w_j := w_j + 1$ とする．
- ラベル $B_{i,k}$ を持つ黒の j-部分木 T が灰色に変わった場合：ラベル $B_{i,k}$ を G_{i,g_i+1} に変え，$\mathsf{Reorder}(D_i)$ を実行する．もし $b_i \geq 2$ かつ $k \neq b_i$ ならば，j-部分木 B_{i,b_i} のラベルを $B_{i,k}$ に変更する．そして，$g_i := g_i + 1$，$b_i := b_i - 1$ とする．
- ラベル $G_{i,k}$ を持つ j-部分木 T が白色に変わった場合：ラベル $G_{i,k}$ を T から削除し，$\mathsf{Reorder}(D_i)$ を実行する．そして，$g_i := g_i - 1$，$w_j := w_j + 1$ とする．
- ラベル $G_{i,k}$ を持つ灰色の j-部分木 T が，黒色に変わった場合：ラベル $G_{i,k}$ を B_{i,b_i+1} に変え $\mathsf{Reorder}(D_i)$ を実行する．そして，$g_i := g_i - 1$，$b_i := b_i + 1$ とする．
- ラベル $G_{i,k}$ を持つ灰色の j-部分木 T において，頂点をリクエストに割り当てたり，T に割り当てられていたリクエストを解放したりしても灰色のままだった場合：$\mathsf{Reorder}(D_i)$ を実行する．
- 黒色や白色の T の色が変わらなかった場合：何もしない．

レベル $i \in \{h-1, h-2\}$ $(= L_{\lg^* h})$ に対する処理

レベル i の割り当てリクエスト R に対して ：T_i 中のレベル i にある自由な頂点を R に割り当てる．

レベル i の解放リクエスト R に対して R に割り当てられている頂点を解放する．

図 1 レベル $h-1$ と $h-2$ に対する処理
Fig. 1 Procedure for levels $h-1$ and $h-2$.

3.3 アルゴリズム ALG

まずレベル $h-1$ と $h-2$ のリクエストに対する処理を図 1 に示す．この部分については 8) で提案したアルゴリズムと同一であり，以下の補題が成り立つ．

補題 2 任意の $i \in \{h-1, h-2\}$ に対して，レベル i の割当リクエストが届いた時，T_i 中のレベル i に自由な頂点が必ず存在する （証明省略）． □

$h-2$ より小さいレベルのリクエストに対する処理は図 2 の通りである．この処理において、各レベル i に対して，カウンタ r_i を用意し，この初期値は 0 とする．各 r_i は，最初から R までの全ての解放リクエストの中で，ある条件を満たすものの個数を保存している．この条件については，図 2 中に記載する．

以下の補題と系は，$g_i \geq \lceil \alpha \rceil + 1$ を満たす場合に，レベル i に利用されている灰色の j-部分木全体で消費する帯域幅に対して下界を与える．

補題 3 もし $g_i = \lceil \alpha \rceil + 1$ ならば，j-部分木 $G_{i,1}, \ldots, G_{i,\lceil \alpha \rceil + 1}$ のレベル $i \in L_j$ にある割当済頂点の消費帯域幅，すなわち $cbw(\{G_{i,1}, \ldots, G_{i,\lceil \alpha \rceil + 1}\})$ は，$2^{h-\ell_j-1}$ 以上である．

証明． これまでに ALG が L_j に属するレベルの集合に対する p 番目のリクエストまでを処理したとし，この時点を時刻 p と言うことにする．もし，灰色の j-部分木 $G_{i,1}$ が存在するなら，ALG は **場合 A1** により p 番目のリクエストに対して $G_{i,1}$ の頂点を割り当てるはずである．つまり，$\lceil \alpha \rceil + 1$ 個の j-部分木が存在するという事実は，(少なくとも) $\lceil \alpha \rceil$ 個の j-部分木が同時に黒色であった時刻が時刻 p 以前にあり，そのため **場合 A2** により，$(\lceil \alpha \rceil + 1)$ 個目の j-部分木にある頂点が，そのレベル i の割当リクエストに対して割り当てられたと考えられる．そのような時刻のうち，最も遅いものを q $(< p)$ とおくことにし，この q に関する条件を（C1）と言うことにする．

黒色の j-部分木の消費帯域幅は $2^{h-\ell_j-1}$ であるので，時刻 q における $\lceil \alpha \rceil$ 個の黒い j-部分木の消費帯域幅は $\lceil \alpha \rceil \cdot 2^{h-\ell_j-1}$ であり，割当済頂点の数は $\lceil \alpha \rceil \cdot 2^{h-\ell_j-1}/2^i = \lceil \alpha \rceil \cdot 2^{h-\ell_j-1-i}$ である．時刻 p においてラベル $G_{i,\lceil \alpha \rceil + 1}$ を持つ j-部分木を G で表し，残りの $G_{i,1}, \ldots, G_{i,\lceil \alpha \rceil}$

レベル $i \in L_j$ に対する処理

レベル i の割り当てリクエスト R に対して ：g_i の値に応じて，次のどちらかの処理を行う．

場合 A1 （$g_i \geq 1$）：$G_{i,1}$ 中のレベル i にある自由な頂点を R に割り当て，$\mathsf{Update}_\alpha(G_{i,1})$ を実行する．

場合 A2 （$g_i = 0$）：$\mathcal{T}_j$ から，任意の白い j-部分木 T を選び，そのレベル i にある自由な頂点を R に割り当てる．そして，$\mathsf{Update}_\alpha(T)$ を実行する．

レベル i の解放リクエスト R に対して ：R に割り当てられていた頂点 u を解放する（これにより u は自由になる）．u が含まれる j-部分木を T とする．g_i と T のラベルに従い，以下のいずれかの処理を行う．

場合 R1 （$g_i \geq 1$, かつ T のラベルが，ある k に対して $B_{i,k}$ である）：$r_i := r_i + 1$ と更新する．この場合は，さらに r_i の値に従い，以下のどちらかの処理を実行する．

場合 R1-(i) （$r_i \mod \lceil\alpha\rceil = 0$）：$j$-部分木 G_{i,g_i} のレベル i にある割当済頂点 v を選び，v を割り当てていたリクエストに u を再割当する．そして，$\mathsf{Update}_\alpha(G_{i,g_i})$ を実行する．

場合 R1-(ii) （$r_i \mod \lceil\alpha\rceil \neq 0$）：$\mathsf{Update}_\alpha(T)$ を実行する．

場合 R2 （$g_i \geq 2$，かつ $1 \leq k \leq g_i - 1$ である k に対して，T がラベル $G_{i,k}$ を持つ）：$r_i := r_i + 1$ と更新する．この場合も，r_i の値に従い，以下のどちらかの処理を行う．

場合 R2-(i) （$r_i \mod \lceil\alpha\rceil = 0$）：$j$-部分木 G_{i,g_i} のレベル i にある割当済頂点 v を選び，$G_{i,1}$ にあるレベル i の自由頂点を，v を割り当てていたリクエストに再割当する．そして，$\mathsf{Update}_\alpha(T)$ と $\mathsf{Update}_\alpha(G_{i,g_i})$ を実行する．

場合 R2-(ii) （$r_i \mod \lceil\alpha\rceil \neq 0$）：$\mathsf{Update}_\alpha(T)$ を実行する．

場合 R3 （$g_i = 0$ または T がラベル G_{i,g_i} を持つ）：$\mathsf{Update}_\alpha(T)$ を実行する．

図 2 $h-2$ よりも低いレベルに対する処理
Fig. 2 Procedure for levels lower than $h-2$.

のラベルを持つ j-部分木の集合を $\mathcal{G}$ で表す．q は $\lceil\alpha\rceil$ 個の j-部分木が黒色である最も遅い時刻であるので，これらの $\lceil\alpha\rceil + 1$ 個の j-部分木 $G_{i,1}, \ldots, G_{i,\lceil\alpha\rceil}$ 全てと G は，時刻 q と p の間は，黒色または灰色である．

以下で議論する場合（I）においては，もし j-部分木 G が，時刻 q と p の間，リスト D_i の最後尾に位置し続けるなら，この補題が成立することを述べる．そして，場合（II）においては，リスト D_i の再後尾の j-部分木が時刻 q と p の間に入れ替わる場合を扱い，その場合の消費帯域幅は場合（I）以上になるとを示すことで，この補題が成立することを示す．

（I） まず以下を仮定する．（仮定 3） j-部分木 G は時刻 q と p の間，リスト D_i の最後尾に位置し続ける．**場合 A1** では，割当リクエストに対して，D_i の先頭の j-部分木 $G_{i,1}$ の頂点が割り当てられなければならず，また，上で述べたように，q は条件（C1）を満たす最後の時刻であるので，（仮定 3）より，時刻 q から p の間には，G の頂点が割り当てられないことがわかる．また，条件（C1）の下で，（仮定 3）は，時刻 q に，$\mathcal{G}$ のすべての j-部分木が黒であることを保証する．すなわち，時刻 q における $\mathcal{G}$ の割当済み頂点の総数 $|A(\mathcal{G})|$ は $\lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i}$ である．時刻 q から p の間に，$\mathcal{G}$ に属する j-部分木の頂点が割り当てられたレベル i の割当リクエストの数を n_a で表す．また，時刻 q から p の間に，$\mathcal{G}$ に属する j-部分木の頂点に対する解放リクエストの数を n_r で表す．

場合 R1 と **R2** は，$\lceil\alpha\rceil$ 回の解放リクエスト毎に 1 回，G （ALG の中では G_{i,g_i} と表されている）の頂点が割り当てられているリクエストに対して，$\mathcal{G}$ の j-部分木の頂点を再割当する．すなわち，時刻 p での $\mathcal{G}$ の割当済み頂点数 $|A(\mathcal{G})|$ は以下である:

$$|A(\mathcal{G})| \geq \lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i} - n_r + \left\lfloor \frac{n_r}{\lceil\alpha\rceil} \right\rfloor + n_a. \tag{1}$$

矛盾を導くため，時刻 p において $cbw(\mathcal{G} \cup \{G\}) < 2^{h-\ell_j-1}$，すなわち，以下を仮定する．

$$\lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i} - n_r + \left\lfloor \frac{n_r}{\lceil\alpha\rceil} \right\rfloor + n_a + n_G \leq |A(\mathcal{G})| + n_G < 2^{h-\ell_j-1-i}. \tag{2}$$

ここで，n_G (≥ 1) は時刻 p での $G_{i,\lceil\alpha\rceil+1}$ の割当済み頂点数である．$\lfloor n_r/\lceil\alpha\rceil\rfloor > n_r/\lceil\alpha\rceil - 1$ が成り立つので，式 (2) より，以下の不等式が得られる．

$$(\lceil\alpha\rceil - 1) \cdot 2^{h-\ell_j-1-i} + n_a + n_G < n_r - \left\lfloor \frac{n_r}{\lceil\alpha\rceil} \right\rfloor < \frac{\lceil\alpha\rceil - 1}{\lceil\alpha\rceil} \cdot n_r + 1. \tag{3}$$

これは以下を意味する．

$$n_r > \lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i} + \frac{\lceil\alpha\rceil}{\lceil\alpha\rceil - 1}(n_a + n_G - 1). \tag{4}$$

これは，時刻 p において，G が灰色であるという以下の仮定に矛盾することになる．式 (4) より，時刻 q から p の間，G の頂点が割り当てられているリクエストについて，少なくとも $\lfloor n_r/\lceil\alpha\rceil\rfloor \geq 2^{h-\ell_j-1-i}$ 個のリクエストに $\mathcal{G}$ の j-部分木の頂点を再割当しなければならないことが分かる．しかし，これは不可能である．時刻 q での G の割当済み頂点数の最大は，高々$2^{h-\ell_j-1-i}$ であり，時刻 p においても灰色である．G を灰色（または黒色）のままにするためには，時刻 q から p の間に，G の少なくとも 1 個の頂点が割当済みでなければならないが，G は D_i の最後尾にあり，G の頂点が割当リクエストに対して割り当てられた最後の時刻が q であるので，時刻 q から p の間に割当は起こらない．よって，仮定 (2) は間違っている．すなわち，時刻 p での $\mathcal{G}$ と G の割当済み頂点の総数 $|A(\mathcal{G})|+n_G(=|A(\mathcal{G}\cup\{G\})|)$ は少なくもと $2^{h-\ell_j-1-i}$ である．以上から，(仮定 3) の下では，消費帯域幅 $cbw(\mathcal{G}\cup\{G\})$ は少なくとも $2^{h-\ell_j-1}$ となる．（(I) はここまで〕

(II) 考えなければならない残りの場合は，時刻 q から p の間に j-部分木 G が D_i の最後尾にとどまらない場合である．以下では，場合 (II) の場合にも場合 (I) と同じく，消費帯域幅 $cbw(\mathcal{G}\cup\{G\})$ が少なくとも $2^{h-\ell_j-1}$ となることを示す．まず，(仮定 4) $q<t\leq p$ について，第 t 番目のリクエスト R_t により，リスト D_i において，最後尾の灰色 j-部分木と別の部分木の交換が 1 回のみ起きると仮定する．この，D_i における最後尾の灰色 j-部分木と別の部分木の交換を，簡単に，**交換**と言うことにする．ここで，幾つかの表記を考える．時刻 s において，レベル i のために使われている灰色 j-部分木の数 g_i を $g_i^{(s)}$ とする．また，時刻 s においてラベル $G_{i,1},\ldots,G_{i,g_i^{(s)}-1}$ を持つ灰色 j-部分木の集合を $\mathcal{G}^{(s)}$ とする．このとき，時刻 s から p の間の $\mathcal{G}^{(s)}$ の j-部分木に対する解放リクエストと割当リクエストの数をそれぞれ $n_r^{(s)}$ と $n_a^{(s)}$ で表す．時刻 $t-1$ と時刻 t に，D_i の最後尾にある j-部分木をそれぞれ $U^{(t-1)}$ と $U^{(t)}$ とする．(交換が起きるため $U^{(t-1)}$ は $U^{(t)}$ と異なる)．時刻 s における j-部分木 T の割当済み頂点の集合と j-部分木の集合 $\mathcal{S}$ の割当済み頂点の集合を，それぞれ，$A^{(s)}(T)$ と $A^{(s)}(\mathcal{S})$ で表す．ここで，時刻 s における関数 $LB(s)$ を考える．これは，以下の議論の中で，次の式を簡単にするために用いられる:

$$LB(s)=\lceil\alpha\rceil\cdot 2^{h-\ell_j-1-i}-n_r^{(s)}+\left\lfloor\frac{n_r^{(s)}}{\lceil\alpha\rceil}\right\rfloor.$$

時刻 q から $t-1$ の間，j-部分木 $U^{(t-1)}$ は D_i の最後尾に常に留まっているので，時刻 $t-1$ での $\mathcal{G}^{(t-1)}$ の割当済み頂点の数 $|A^{(t-1)}(\mathcal{G}^{(t-1)})|$ は式 (1) と同じく以下のように見積もることができる:

$$\begin{aligned}|A^{(t-1)}(\mathcal{G}^{(t-1)})| &\geq \lceil\alpha\rceil\cdot 2^{h-\ell_j-1-i}-n_r^{(t-1)}+\left\lfloor\frac{n_r^{(t-1)}}{\lceil\alpha\rceil}\right\rfloor+n_a^{(t-1)} \qquad (5)\\ &= LB(t-1)+n_a^{(t-1)}.\end{aligned}$$

第 t 番目のリクエスト R_t が処理される時に場所の交換が，場合 (I) のように起きなかった場合には（ただし，ここでは交換が起こると仮定している)，$\mathcal{G}^{(t)}$ の割当済み頂点の数 $|A^{(t)}(\mathcal{G}^{(t)})|$ は少なくとも $LB(t)+n_a^{(t)}$ であることに注意する．以下では，$|A^{(t)}(\mathcal{G}^{(t)})|\geq LB(t)$ が成り立つことを示す．

$|A^{(t)}(U^{(t)})|<|A^{(t)}(U^{(t-1)})|$ となる状況が起きるので，第 t 番目のリクエスト R_t が処理されるとき $U^{(t-1)}$ と $U^{(t)}$ の交換が実行される．この状況は，$U^{(t)}$ の頂点に対する解放リクエストにのみ引き起こされる．なぜなら，**場合 A1** において，$G_{i,1}$ の頂点が割当リクエストに対して割り当てられ，これにより D_i の灰色 j-部分木の順序変更は無いからである．ここで，**場合 A2** は，時刻 q について条件（C1）に従わないので，考慮していないことに注意する．解放リクエストに関して，そのような交換は**場合 R1-(ii)** と **R2-(ii)** のみで起きることが次のことからわかる．**場合 R1-(i)** では，(ALG の中では G_{i,g_i} と表されている) $U^{(t-1)}$ の割当済み頂点数は 1 つ減る．すなわち，$|A^{(t)}(U^{(t-1)})|=|A^{(t-1)}(U^{(t-1)})|-1$ である．よって，$U^{(t-1)}$ は D_i の最後尾に留まり，交換は起きない．**場合 R2-(i)** では，(ALG の中では T と表されている) 灰色 j-部分木の割当済み頂点数が 1 つ減る．しかし，D_i の最後尾に $U^{(t-1)}$ が留まるために，再割当が起きるので，(ALG の中では G_{i,g_i} と表されている) $U^{(t-1)}$ の割当済み頂点数も 1 つ減るので，交換は起きない．**場合 R3** では，$g_i^{(t-1)}=0$ の場合には 1 個の灰色 j- 部分木しか現れない，もしくは，(ALG の中では G_{i,g_i} と表されている) $U^{(t-1)}$ の割当済み頂点数は減るので，交換は起きない．以上により，時刻 $t-1$ での割当済み頂点数に関しては，$U^{(t)}$ の頂点に対する 1 回の解放リクエストの結果として $|A^{(t)}(U^{(t)})|<|A^{(t)}(U^{(t-1)})|$ が成立し，$|A^{(t-1)}(U^{(t)})|=|A^{(t-1)}(U^{(t-1)})|$ が成立する．

$\mathcal{G}^{(t)}$ の割当済み頂点数 $|A^{(t)}(\mathcal{G}^{(t)})|$ を見積もる．j-部分木 $U^{(t)}$ に対する解放リクエスト R_t により，$|A^{(t)}(\mathcal{G}^{(t)})|+|A^{(t)}(U^{(t)})|=|A^{(t-1)}(\mathcal{G}^{(t-1)})|-1+|A^{(t-1)}(U^{(t-1)})|$ が成り立つ．$|A^{(t-1)}(U^{(t-1)})|=|A^{(t-1)}(U^{(t)})|$ であり，$U^{(t)}$ から 1 個の頂点に対して解放リクエストがあるので $|A^{(t)}(U^{(t)})|=|A^{(t-1)}(U^{(t)})|-1)$ となり，

$$\begin{aligned}|A^{(t)}(\mathcal{G}^{(t)})| &= |A^{(t-1)}(\mathcal{G}^{(t-1)})|-1+|A^{(t-1)}(U^{(t-1)})|-|A^{(t)}(U^{(t)})|\\ &= |A^{(t-1)}(\mathcal{G}^{(t-1)})|-1+|A^{(t-1)}(U^{(t)})|-(|A^{(t-1)}(U^{(t)})|-1)\\ &= |A^{(t-1)}(\mathcal{G}^{(t-1)})|\geq LB(t-1),\end{aligned}$$

が成り立つ．ここで，最後の不等式は式 (5) から得られる．**場合 R1-(ii)** または **R2-(ii)** の実行によって $n_r^{(t)} = n_r^{(t-1)} + 1$ および $n_r^{(t)} \mod \lceil\alpha\rceil \neq 0$ となるので，$LB(t-1) = LB(t)+1$ が成り立つ．これにより，$|A^{(t)}(\mathcal{G}^{(t)})| \geq LB(t)+1$ となる．

（仮定 4）より，時刻 t の後には交換が起きないので，$|A^{(p)}(\mathcal{G}^{(p)})|$ の値を以下のように見積もることができる．ただし，n'_a (≥ 0) は，時刻 t から p の間の $\mathcal{G}^{(t)}(=\mathcal{G}^{(p)})$ に対する割当リクエストの数を表している．

$$\begin{aligned}
|A^{(p)}(\mathcal{G}^{(p)})| &= |A^{(t)}(\mathcal{G}^{(t)})| - (n_r^{(p)} - n_r^{(t)}) + \left\lfloor \frac{n_r^{(p)} - n_r^{(t)}}{\lceil\alpha\rceil} \right\rfloor + n'_a \\
&\geq LB(t) + 1 - (n_r^{(p)} - n_r^{(t)}) + \left\lfloor \frac{n_r^{(p)} - n_r^{(t)}}{\lceil\alpha\rceil} \right\rfloor \\
&= \lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i} - n_r^{(p)} + \left\lfloor \frac{n_r^{(t)}}{\lceil\alpha\rceil} \right\rfloor + \left\lfloor \frac{n_r^{(p)} - n_r^{(t)}}{\lceil\alpha\rceil} \right\rfloor + 1 \\
&\geq \lceil\alpha\rceil \cdot 2^{h-\ell_j-1-i} - n_r^{(p)} + \left\lfloor \frac{n_r^{(p)}}{\lceil\alpha\rceil} \right\rfloor = LB(p)
\end{aligned}$$

最後の不等号は $\lfloor (n_r^{(p)} - n_r^{(t)})/\lceil\alpha\rceil \rfloor + 1 \geq \lfloor n_r^{(p)}/\lceil\alpha\rceil\rfloor - \lfloor n_r^{(t)}/\lceil\alpha\rceil\rfloor$ という事実による．よって，$\mathcal{G}^{(p)} \cup \{U^{(t)}\}$ $(= \mathcal{G} \cup \{G\})$ の割当済み頂点数は $LB(p) + n_G$ 以上となる．ここで，n_G (≥ 1) は時刻 p においても灰色となる G（j-部分木 $U^{(t)}$）の割当済み頂点数である．これ以降は (I) の場合と同じ議論により，$|A(\mathcal{G} \cup \{G\})| = |A(\mathcal{G}^{(p)} \cup \{U^{(t)}\})| \geq 2^{h-\ell_j-1-i}$ となり，これは $cbw(\mathcal{G} \cup \{G\}) \geq 2^{h-\ell_j-1}$ を意味する．よって，(仮定 4) の下で，補題が成り立つ．

以上は交換が 1 回のみの場合について議論であるが，交換が 2 回以上起きる場合についても，$\mathcal{G}$ の灰色 j-部分木の割当済み頂点数が少なくとも $LB(p)$ となることを，交換が起きるリクエスト列を部分列に分割して，同じ議論を繰り返すことにより示すことができる．

〔(II) はここまで〕□

系 1　整数 $\beta \geq 2$ について $g_i = \lceil\alpha\rceil + \beta$ の場合，すべての j-部分木 $G_{i,1}, \ldots, G_{i,\lceil\alpha\rceil+\beta}$ のレベル $i \in L_j$ における割当済み頂点の消費帯域幅は，少なくとも $\beta \cdot 2^{h-\ell_j-1}$ である．

証明．証明は補題 3 とほぼ同様である．式 (1) は次のように変更される．

$$|A(\mathcal{G})| \geq (\lceil\alpha\rceil + \beta - 1) \cdot 2^{h-\ell_j-1-i} - n_r + \left\lfloor \frac{n_r}{\lceil\alpha\rceil} \right\rfloor + n_a.$$

また，矛盾を導くための仮定 (2) は次のように変更される．

$$(\lceil\alpha\rceil + \beta - 1) \cdot 2^{h-\ell_j-1-i} - n_r + \left\lfloor \frac{n_r}{\lceil\alpha\rceil} \right\rfloor + n_a + n_G \leq |A(\mathcal{G})| + n_G < \beta \cdot 2^{h-\ell_j-1-i}.$$

同様に，式 (3) と (4) により，$\lfloor n_r/\lceil\alpha\rceil\rfloor \geq 2^{h-\ell_j-1-i}$ が得られる．これは，時刻 q から p の間に，$G_{i,\lceil\alpha\rceil+\beta}$ を除く j-部分木の $\lfloor n_r/\lceil\alpha\rceil\rfloor$ 個の頂点が $G_{i,\lceil\alpha\rceil+\beta}$ に割り当てられていたリクエストに再割当されるので，時刻 p において $G_{i,\lceil\alpha\rceil+\beta}$ が灰色であるという事実に矛盾する．よって，本系が成り立つ．□

補題 3 と系 1 より，$\lg^* h - 2$ 以下のレベルへの割当リクエストに対して，ALG は，常に自由な頂点を見つけることが出来ることを以下の補題により保証できる．

補題 4　$1 \leq j \leq \lg^* h - 1$ について，レベル $i \in L_j$ の割当リクエスト R が到着した時刻を考える．このとき，以下の （i） または （ii）が成り立ち，ALG は割当リクエストに対して常に自由な頂点を見つけることができる：（i） $\mathcal{T}_j$ の中でラベル $G_{i,1}$ を持つ灰色 j-部分木が存在する．(ii) $\mathcal{T}_j$ の中に白色 j-部分木が残っている．

証明．矛盾を導くため，(仮定 5) として，ラベル $G_{i,1}$ を持つ灰色 j-部分木は存在せず，さらに，$\mathcal{T}_j$ に白色 j-部分木は残っていないと仮定する．補題 3 と系 1 より，もし $g_k \geq \lceil\alpha\rceil + 1$ ならば，$(g_k - \lceil\alpha\rceil) \cdot 2^{h-\ell_j-1}$ の帯域幅が，レベル $k \in L_j$ で消費されている．よって，$\mathcal{T}_j$ の灰色 j-部分木により消費されている総帯域幅は，少なくとも以下のようになる．

$$2^{h-\ell_j-1} \cdot \sum_{k \in L_j} (g_k - \lceil\alpha\rceil).$$

黒色 j-部分木は $2^{h-\ell_j-1}$ の帯域幅を消費して，各 $k \in L_j$ について b_k 個の黒色 j-部分木が存在するので，$\mathcal{T}_j$ の総消費帯域幅 $cbw(\mathcal{T}_j)$ は以下の不等式を満たす．

$$cbw(\mathcal{T}_j) \geq 2^{h-\ell_j-1} \cdot \sum_{k \in L_j} (b_k + g_k - \lceil\alpha\rceil).$$

$\mathcal{T}_j$ の j-部分木数は $\lceil\alpha\rceil(\ell_{j-1} - \ell_j) + 2^{\ell_j+1}$ であり，(仮定 5) よりすべての j-部分木は L_j のあるレベルで用いられているので，$\sum_{k \in L_j}(b_k + g_k) = \lceil\alpha\rceil(\ell_{j-1} - \ell_j) + 2^{\ell_j+1}$ が成り立つ．よって，グループ L_j は $(\ell_{j-1} - \ell_j)$ 個のレベルを含むので，$\mathcal{T}_j$ の総消費帯域幅 $cbw(\mathcal{T}_j)$ は以下のように見積もることができる:

$$\begin{aligned}
cbw(\mathcal{T}_j) &\geq 2^{h-\ell_j-1} \cdot \sum_{k \in L_j} (b_k + g_k - \lceil\alpha\rceil) \\
&= 2^{h-\ell_j-1} \cdot \left(\sum_{k \in L_j} (b_k + g_k) - \lceil\alpha\rceil(\ell_{j-1} - \ell_j) \right) \\
&= 2^{h-\ell_j-1} \cdot 2^{\ell_j+1} = 2^h.
\end{aligned}$$

これは，リクエスト R が到着する直前に，割当済み頂点の総消費帯域幅が少なくとも 2^h で

あり，解放リクエストが来る前に新しい割当リクエスト R が来たことを意味する．R の帯域幅を追加することにより，総帯域幅は 2^h を超えてしまい，仮定 1 に矛盾する．よって，本補題が成り立つ．□

補題 1 で ALG が必要とする木の数を示した．補題 2 と補題 4 は，ALG の正しさを保証する．**場合 R1** と **R2** において，$\lceil\alpha\rceil$ 回の解放リクエスト毎に高々1 回の再割当を行うので，ALG の総コストは高々$n + m/\alpha$ である．ここで，n と m は，それぞれ，入力列における割当リクエストと解放リクエストの数である．最適オフラインアルゴリズムのコストは少なくとも n であるので，以下の定理を得ることができる．

定理 1　ALG は $(1+1/\alpha)$-競合であり，高さ h の木を高々$(1+\lceil\alpha\rceil)\log^* h$ 個必要とする．

補足 1　ALG のコストが，漸近的に，最適アルゴリズムのコストの $1+1/\alpha$ になるような都合の悪い例が存在する．簡単のため，$\lceil\alpha\rceil = 2$ の場合を考える．高さ 7 の OVSF 木と最下のグループ L_1 を考える．また，図 3 で示すように，2 個の高さ 3 の 1-部分木を考える．(1) レベル 0 の 10 個の割当リクエスト R_1 から R_{10} があり，v_1 から v_{10} の 10 頂点がそれぞれ割り当てられたとする．(2) 左部分木 $B_{0,1}$ に割り当てられていた R_7 と R_8 に対して解放リクエストがあったとする．このとき，ALG は R_7 と R_8 を解放して，頂点 v_7 を R_{10} に再割当を行い，頂点 v_{10} が自由になる．(3) 2 個の割当リクエスト R_{11} と R_{12} があったとき，ALG は v_8 と v_{10} をそれぞれ割り当てる．この時点で状況 (1) と同じ状況に戻る．状況 (2) と (3) と同じように，2 個の解放リクエストと 2 個の割当リクエストが繰り返されたとき，ALG の再割当数は，漸近的に，割当リクエスト数の半分になる．一方，状況 (2) において，最適アルゴリズムは R_7 と R_8 を解放するのみで再割当を行わずに，状況 (3) において，自由頂点 v_7 を R_{11} に割り当て，自由頂点 v_8 を R_{12} に割り当てる．すなわち，再割当を行わない．よって，ALG のコストは，最適なコストの $1+1/\lceil\alpha\rceil$ 倍になる．α が大きい場合にも，連続する $\lceil\alpha\rceil$ 個の解放と $\lceil\alpha\rceil$ 個の割当リクエストを考えることで，同様の都合の悪い例を示せる．よって，定理 1 の証明で示した競合比解析はタイトである．

謝辞　本研究の一部は，文部科学省の科学研究費補助金（22700019 と 23500020）の助成を受けた．

参 考 文 献

1) Chan, J.W.T., Chin, F.Y.L., Ting, H.F., Zhang, Y.: Online tree node assignment with resource augmentation. COCOON 2009, pp.358–367 (2009).
2) Chan, J.W.T., Chin, F.Y.L., Ting, H.F., Zhang, Y.: Online problems for frequency assignment and OVSF code assignment in wireless communication networks. ACM SIGACT News, Vol. 40 (3), pp.86–98 (2009)
3) Chin, F.Y.L., Ting, H.F., Zhang, Y.: Constant-competitive tree node assignment. manuscript.
4) Chin, F.Y.L., Zhang, Y., Zhu, H.: Online OVSF code assignment with resource augmentation. AAIM 2007, pp.191–200 (2007).
5) Epstein, L., Stee, R.v.: Online bin packing with resource augmentation. Discrete Optimization, 4, pp.322-333 (2007).
6) Erlebach, T., Jacob, R., Mihalák, M., Nunkesser, M., Szabó, G., Widmayer, P.: An algorithmic view on OVSF code assignment. STACS 2004, pp.270–281 (2004).
7) Miyazaki, S., Okamoto, K.: Improving the competitive ratio of the online OVSF code assignment problem. Algorithms 2009, 2(3), pp.953–972 (2009).
8) 朝廣雄一，上米良謙太，宮野英次：資源増加を許した OVSF 符号割当問題に対する 2 競合アルゴリズム．情報処理学会研究報告，Vol.2011-AL134-13, pp.1-7, 2011 年 3 月．

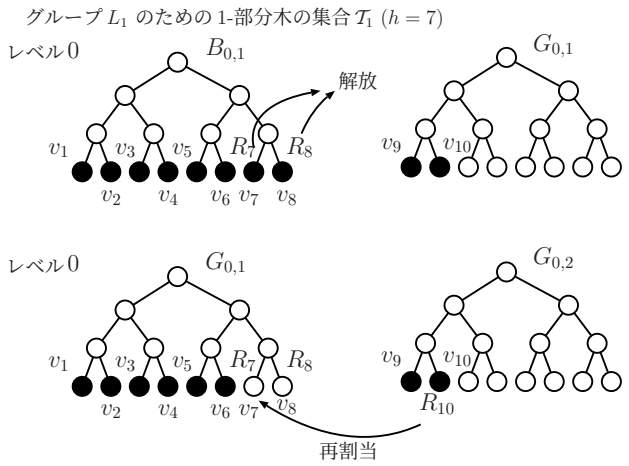

図 3　$\alpha = 2$ の場合の，ALG にとって都合の悪い例 (状況 (2))
Fig. 3　A worst case example for $\alpha = 2$ (Situation (2)).