

# 修士論文

## タスクスケジューリング アルゴリズムに関する研究: DAGの高さが4以下の場合



平成18年度修了  
三重大学大学院 工学研究科  
博士前期課程 情報工学専攻

清水 豪樹



# 目次

はじめに	1
第1章 通信遅延を考慮したタスクスケジューリング問題	3
1.1 通信遅延を考慮したタスクスケジューリング問題	3
1.2 通信遅延	4
1.3 タスクの複製	4
1.4 DAG $G$ のスケジューリング	5
1.5 本研究の前提条件	6
第2章 従来近似アルゴリズム	7
2.1 近似アルゴリズムと近似精度	7
2.2 C. H. Papadimitriou と M. Yannakakis の手法	7
2.2.1 e-value	8
2.2.2 Papadimitriou らの近似アルゴリズム	8
第3章 本手法の近似アルゴリズム	9
3.1 準備	9
3.2 DAG の変換	9
3.3 本研究のアルゴリズム	10
3.4 本手法の時間計算量	20
3.5 従来手法との近似精度の比較	21
おわりに	22
謝辞	23

## はじめに

近年、様々な分野において、コンピュータを用いて大規模かつ複雑な計算が行われるようになってきた。しかし、1台のコンピュータで大規模な計算を短時間で行うことには限界がある。そこで、与えられた問題を分割し、複数のプロセッサに分散して並列に処理することで、処理速度の向上を図る並列処理技術の研究が盛んに行われている。

並列処理システムの性能を最大限に発揮するためにはタスクスケジューリングが重要である。タスクスケジューリングとは、複数のタスクで構成されるプログラムの処理を短時間で終わるように、各タスクの実行時間、通信遅延、タスク間の依存関係を与えて、各タスクの実行プロセッサとその実行開始時刻を決定することである。並列処理システムではタスクの処理時間と比べ、プロセッサ間の通信に多くの時間がかかるため、プロセッサ間の通信遅延時間も考慮しなければ良いスケジューリングを行うことはできない。よって、通信遅延を考慮したタスクスケジューリングが重要となる。しかし、通信遅延を考慮したタスクスケジューリング問題は一般に NP 完全であり、最適スケジュールを現実的な時間で求めることは困難であるとされている。そこで、できる限り最適解に近い近似解を短時間で求める効率の良い近似アルゴリズムの研究が盛んに行われてきた。

通信遅延を考慮したタスクスケジューリング問題に対し、タスク間の依存関係を DAG(非循環有向グラフ) で表現し、文献 [1] において、Papadimitriou らは、この問題の NP 完全性と、近似精度 2 を持つアルゴリズムを示し、近似精度が 2 より小さいアルゴリズムが存在するかどうかを未解決問題として提示している。文献 [2] において河田らは、DAG の高さが 2、各タスクの実行時間が 1、通信遅延時間が一定の値で与えられるという条件下でも、この問題が NP 完全であることを証明した。文献 [3] において片柳らは、各タスクの実行時間が 1、通信遅延時間が一定の値で与えられる場合において、近似精度  $2 - \frac{1}{3c}$  のアルゴリズムを提示した。このアルゴリズムは、DAG から再構成される二部グラフの最大マッチングの辺の本数に着目することにより、タスク間の依存関係をより詳細に解析し、その解析結果を用いて近似精度を改善している。ここで、 $c = \lfloor \frac{\text{low-value}}{\tau+1} \rfloor \geq 1$  であり、low-value は e-value よりも正確なタスクの開始時刻の下界である。

各タスクの通信遅延時間が一定の値で与えられない場合では、近似精度が 2 よりも小さいアルゴリズムがあるかどうかは未解決問題のままであった。文献 [5] において DAG の高さが 2 の場合における近似精度  $\frac{3}{2}$  のアルゴリズムが提示され、初めて通信遅延時間が可変の場合の近似精度が 2 よりも小さいアルゴリズムが示された。さらに文献 [6] では、より近似精度を改善した近似精度  $\frac{7}{5}$  のアルゴリズムが提示されている。また、高さ 3 の DAG に対しては、文献 [7] において、最大マッチングを繰り返し適用することで近似精度 1.8 のアルゴリズムが提示され、さらに文献 [8] において、より近似精度を改善した近似精度  $\frac{5}{3}(= 1.66)$  のアルゴリズムが提示されている。しかし、

DAG の高さが 4 以上の場合に近似精度が 2 よりも小さいアルゴリズムが存在するかどうかは未解決問題のままであった。本研究では、DAG の高さが 4 のときに近似精度  $\frac{49}{26}$  のアルゴリズムを与える。このアルゴリズムは、文献 [7] と同様に最大マッチングを繰り返し適用する方法をとるが、レベル 3 のタスクのスケジュール時刻を精密に計算することにより、近似精度  $\frac{49}{26}$  を達成した。

本論文の構成は以下の通りである。第 1 章では、通信遅延を考慮したタスクスケジューリング問題について述べる。第 2 章では、e-value[1] の与え方、近似精度 2 の近似アルゴリズム [1]、について解説する。第 3 章では、DAG の高さが 4 の場合における、本研究で提案する近似アルゴリズムについて解説する。最後に、本論文のまとめを述べる。

# 第1章 通信遅延を考慮したタスクスケジューリング問題

本章では、通信遅延時間を考慮したタスクスケジューリング問題について説明する。また、本研究で用いる前提条件についても述べる。

## 1.1 通信遅延を考慮したタスクスケジューリング問題

通信遅延時間を考慮したタスクスケジューリング問題 [1] とは、複数のタスクと、各タスク間の依存関係、各タスクの実行時間、各タスクの通信遅延時間が与えられたときに、最短に全てのタスクの実行を終了するスケジュールを求める問題である。スケジュールとは、各タスクの処理を行うプロセッサと処理の開始時刻を決定した割り当てである。問題の入力である複数のタスクとタスク間の依存関係を DAG（非循環有向グラフ：Directed Acyclic Graph）で表わす。DAG の頂点がタスク、辺がタスク間の依存関係を表わす。図 1.1 に、DAG の例を示す。

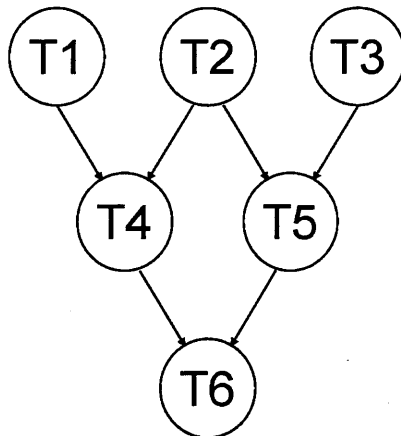


図 1.1: DAG の例

図 1.1 の DAG の T4 は、その先祖である T1 と T2 の実行結果を受け取らなければ、実行を開始できない。これは他のタスクについても同様であり、先祖を持つタスクの実行を開始するには、その先祖のタスク全ての実行結果が必要である。

## 1.2 通信遅延

通信遅延とは、あるプロセッサが処理したタスクの結果を他のプロセッサへ送る際に必要な通信にかかる時間のことである。一般に、通信遅延はタスクの処理時間よりも大きいので、良いスケジュールを得るためにはこの通信遅延による遅れを回避することを考慮しなければならない。図 1.2 に通信遅延が生じないスケジュール、生じるスケジュールの例を示す。

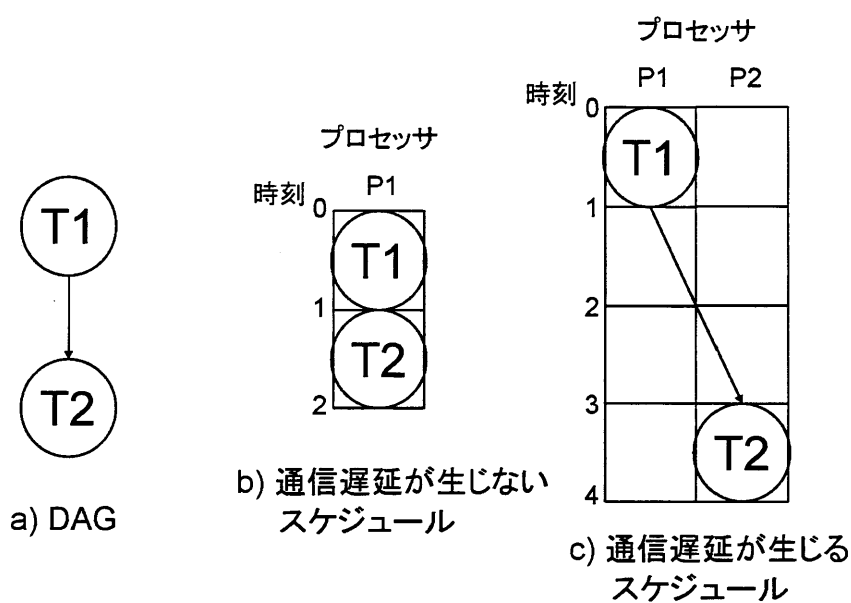


図 1.2: 通信遅延の生じないスケジュールと生じるスケジュールの例

図 1.2 の b), c) はそれぞれ a) に対する、通信遅延が生じないスケジュールと、通信遅延が生じるスケジュールである。ここで、a) の矢印はタスク間の依存関係を、c) の矢印は通信遅延を表している。b) のようにタスクを同一プロセッサにスケジュールした場合には通信遅延は生じないが、c) のように異なるプロセッサにスケジュールした場合には通信遅延が生じる。

通信遅延は異なるプロセッサ間の通信によって発生するが、通信遅延時間は各タスクが持つ値であり、プロセッサ間ごとに持つ値ではない。すなわち、通信遅延時間はタスクの実行結果のサイズに依存する。

## 1.3 タスクの複製

通信遅延を回避する手段の一つにタスクの複製がある。図 1.3 にタスクの複製を許さないスケジュールと許すスケジュールの例を示す。

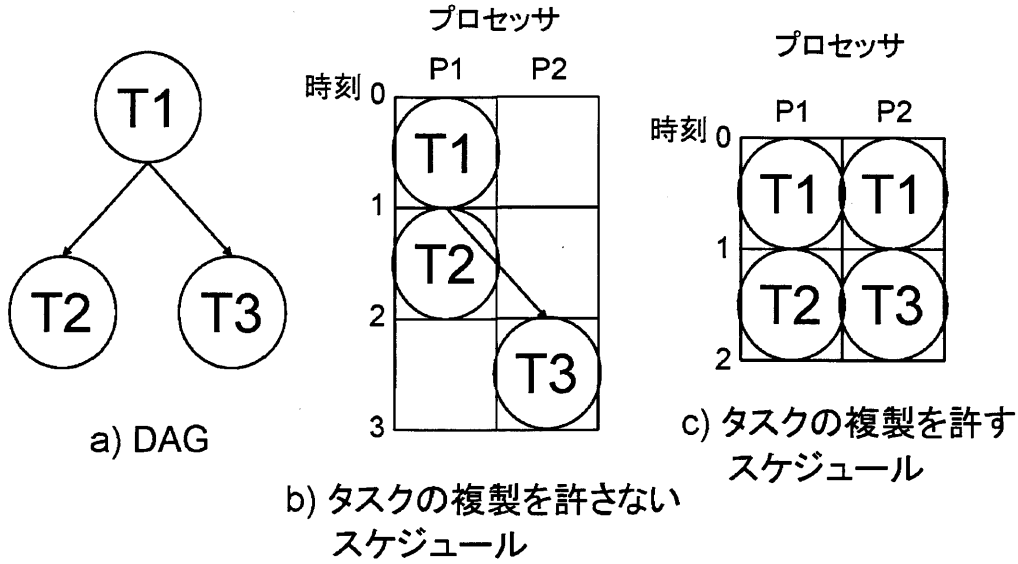


図 1.3: タスクの複製を許さないスケジュールと許すスケジュールの例

図 1.3 の b) ではタスク T1 から T3 に通信遅延が生じているが、c) では T1 をプロセッサ P2 に複製してスケジュールすることで通信遅延を解消している。このように、同一プロセッサにスケジュールされたタスク間では通信遅延が生じないことを利用して、タスクの複製により全タスクの総実行時間を短縮できる場合がある。

## 1.4 DAG $G$ のスケジューリング

$V$  をタスクの集合、 $A$  をタスク間の依存関係の集合、 $P$  をプロセッサの集合、 $T$  をタスクの開始時刻の集合  $\{0, 1, 2, \dots\}$  とし、 $v \in V$  に対して  $x(v)$  を  $v$  の実行時間、 $\tau(v)$  を通信遅延時間としたとき、DAG  $G = (V, A)$  のスケジュール  $S$  とは以下の条件を満たす  $V \times P \times T$  の部分集合である。

1. 各  $v \in V$  に対し、少なくとも一つの  $(v, p, t) \in S$  が存在する。
2.  $(v, p, t), (v', p, t) \in S$  ならば、 $v = v'$ 。
3.  $(u, v) \in A, (v, p, t) \in S$  ならば、 $(u, p, t') \in S$  かつ  $t' \leq t - x(u)$  または、 $(u, p', t') \in S$  かつ  $t' \leq t - x(u) - \tau(u)$ 。

通信遅延を考慮したタスクスケジューリング問題の目的は、タスクの実行開始時刻の最大値  $\max\{t | (v, p, t) \in S\}$  を最小化することである。

## 1.5 本研究の前提条件

本研究で用いる前提条件を以下に示す.

1. 等しい機能を持つプロセッサを無制限に使用できる
2. 各タスクの実行時間は正整数
3. 各タスクの通信遅延時間は正整数
4. タスクの複製を許す

ここで, ブロードキャスト (複数個のプロセッサへの同時通信) を許せば, プロセッサ数は高々全タスク数で十分ながわっている [4]. この前提条件下で, 高さ 4 の DAG  $G = (V, A)$  をスケジューリングの対象とする.

また, プロセッサを無制限に使用できることと, タスクの複製を許すことにより,  $n$  個の葉を持つ DAG の全タスクの実行終了時刻は, 各葉へのパスを持つ頂点から構成される  $n$  個の DAG の全タスクの実行終了時刻の中で最大のものとなる. よって以下では, 葉の数が 1 である DAG のスケジューリングについてのみ考える. そして, その葉のタスクを  $v^*$  と呼ぶ.

## 第2章 従来の近似アルゴリズム

本章では、通信遅延を考慮したタスクスケジューリングに対する、従来の近似アルゴリズムについて述べる。

### 2.1 近似アルゴリズムと近似精度

前述した通り、通信遅延を考慮したタスクスケジューリング問題は NP 完全であり、最適解を現実的な時間で計算することは困難であると考えられている。そこで、最適解を求める代わりに、短時間で、できるだけ最適解に近い解（近似解）を求める手法が研究されており、近似解を求めるアルゴリズムを近似アルゴリズムという。近似アルゴリズムの良否を示す指標の一つに近似精度がある。近似精度は以下の式で定義される。

$$\text{近似精度} = \frac{\text{近似解}}{\text{最適解}}$$

定義からわかるように、最小化問題における近似精度 2 のアルゴリズムは最適解の 2 倍以下の解を出力する。近似精度を求めるためには最適解が必要であるが、上述の通り最適解を求めることは困難であるため、最適解の代わりに最適解の下界を用いる。ここで、最適解の下界とは、最も遅い時刻にスケジュールされるタスク  $v^*$  の実行開始時刻の下界である。この値をスケジュール時刻の下界とし、近似アルゴリズムの出力したスケジュールでの、タスク  $v^*$  の実行開始時刻をスケジュール時刻の上界とすると、近似精度は以下の関係を満たす。

$$\text{近似精度} = \frac{\text{近似解}}{\text{最適解}} \leq \frac{\text{スケジュール時刻の上界}}{\text{スケジュール時刻の下界}}$$

近似アルゴリズムは、スケジュール時刻の上界の改善による近似精度の改善が可能である。また、スケジュール時刻の下界を改善し、下界をより最適解に近づけることによって近似精度の改善が可能である。本研究では、この両方のアプローチを用いて近似精度を改善する。

### 2.2 C. H. Papadimitriou と M. Yannakakis の手法

Papadimitriou らの定義した e-value [1] と、それを用いた近似精度 2 の近似アルゴリズムについて述べる。

### 2.2.1 e-value

タスクの e-value とは、そのタスクの実行開始時刻の下界となる値である。DAG  $G = (V, A)$  に対し、タスク  $v \in V$  の e-value  $e(v)$  を以下で定める。また、タスク  $v \in V$  の f-value  $f(v)$  を  $f(v) = e(v) + x(v) + \tau(v)$  と定義する。

- $v$  が DAG の根である場合、 $e(v) = 0$ .
- そうでない場合、次のように計算する。
  - $v$  の先祖のタスクの集合を  $U$  とし、 $u \in U$  の f-value  $f(u) = e(u) + x(u) + \tau(u)$  を求め、降順に並べたものを  $u_1, u_2, \dots, u_{|U|}$  とする。従って、 $f(u_1) \geq f(u_2) \geq \dots \geq f(u_U)$  が成立する。
  - $f(u_k) > j \geq f(u_{k+1})$  を満たす整数  $j$  について、タスク  $u_i (i \leq k)$  のうち、 $v$  への経路を持つタスクのみで構成される部分 DAG を  $N_j(v)$  と表す。
  - タスク  $v_i \in N_j(v)$  を e-value の降順に並べたものを  $v_1, v_2, \dots, v_l$  とする。ただし、 $(l = |N_j(v)|)$  である。このとき、 $e(v_1) \geq e(v_2) \geq \dots \geq e(v_l)$  が成立する。
  - $j \geq \max_{1 \leq i \leq l} [e(v_i) + \sum_{q=1}^i x(v_q)]$  を満たす最小の  $j$  を  $e(v)$  とする。

このように計算された e-value  $e(v)$  がタスクの開始時刻の下界となる証明については文献 [1] を参照されたい。

### 2.2.2 Papadimitriou らの近似アルゴリズム

次に、Papadimitriou らの近似精度 2 のタスクスケジューリングアルゴリズムを示す。タスク  $v$  をスケジューリングする手続きは以下の通りである。

- $u \in N_{e(v)}(v)$  は、e-value の昇順に時刻  $e(v) + e(u)$  以降に同一のプロセッサにスケジューリングし、最後にタスク  $v$  をスケジューリングする。
- $u \notin N_{e(v)}(v)$  は、それぞれ異なるプロセッサにスケジューリングする。

このようにスケジューリングすることで、最適スケジューリング時刻の 2 倍以下のスケジューリングが得られる。詳細については文献 [1] を参照されたい。

## 第3章 本手法の近似アルゴリズム

本章では、本研究で考案した近似アルゴリズムについて述べる。

### 3.1 準備

以下に、必要な定義を述べる。

**定義 1**  $v^*$  の最適スケジュール時刻を  $opt$  と表す。

**定義 2**  $v \in V$  のスケジュール時刻の下界  $e$ -value  $e(v)$  と,  $f$ -value  $f(v)$  を  $[1]$  と同様に定義する。

**定義 3**  $v^*$  のスケジュール時刻の下界を  $low$  とし,  $low$  の初期値を  $e(v^*)$  とする。

**定義 4**  $v \in V$  に対して, DAG  $G$  の根から  $v$  への最大パス長を  $level(v)$  と表記し, DAG の高さを  $\max\{level(u) | u \in V\}$  と定義する。

**定義 5** 互いに素な頂点集合  $U, U'$  において  $U$  に始点,  $U'$  に終点がある  $G$  上の有向経路の集合  $A$  のマッチング  $M$  とは,  $A$  の部分集合で, すべての頂点  $v \in U \cup U'$  について  $v$  を端点とする経路が高々1つしかないものである。

**定義 6** 任意のマッチング  $M'$  に対して  $|M| \geq |M'|$  が成り立つようなマッチング  $M$  を最大マッチングと呼ぶ。

### 3.2 DAG の変換

与えられた DAG において,  $V$  の要素である処理時間が  $x(v)(> 1)$  のタスクを, 処理時間が1である  $x(v)$  個のタスクに並列に分割し, 分割したそれぞれのタスクの通信遅延時間を  $\tau(v) + x(v) - 1$  にすることで, 全てのタスクの処理時間を1にする。また, タスク  $v^*$  をスケジュールするプロセッサをメインプロセッサ, それ以外のプロセッサを外部プロセッサと呼ぶ。図 3.1 に DAG の変換の例を示す。

図 3.1 は, 実行時間が 3, 通信遅延時間が 4 のタスクの変換の例である。変換後のタスクの実行時間が 1 になり, 通信遅延時間は  $3 + 4 - 1 = 6$  となる。

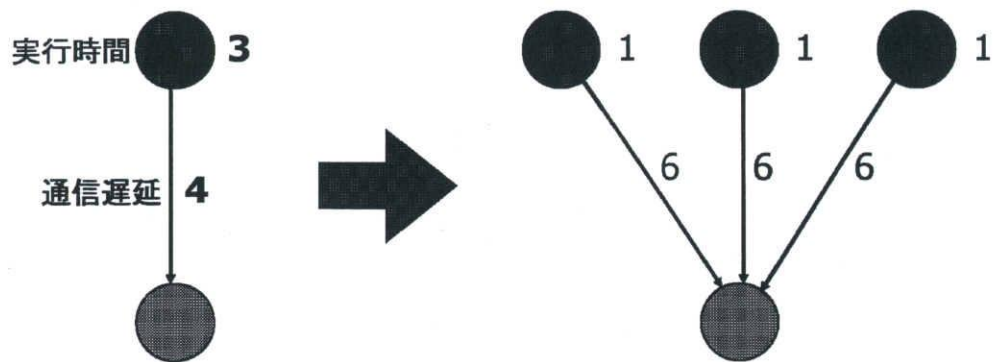


図 3.1: DAG の変換

**補題 1** 変換後の DAG を入力として得られた各タスクのスケジュール時刻で、変換前の DAG のスケジュールが可能である。

**証明)** タスクの処理時間を  $x(v) (> 1)$  とし、 $v$  を分割してできた処理時間が 1 のタスクを  $W = \{w_1, \dots, w_{x(v)}\}$  と定義する。  $w \in W$  の  $e$ -value は  $e(w) = e(v)$  になる。

- $w \in W$  が全てメインプロセッサにスケジュールされた場合  
 $w \in W$  が全て連続してスケジュールされているときは、最も早い時刻にスケジュールされた  $w \in W$  を  $v$  のスケジュールにする。連続していないときは、 $w' \notin W$  かつ  $e(w') = e(w)$  となる  $w'$  が存在し、これを  $w \in W$  の後にスケジュールすることで、前述と同様に  $v$  をスケジュールできる。
- $w \in W$  が全て外部プロセッサにスケジュールされた場合  
 任意の  $w$  を選び、それを  $v$  スケジュールにする。  $w$  と  $v$  の  $f$ -value は同じなので、 $v$  の子のスケジュールには影響を与えない。
- $w \in W$  がメインプロセッサと外部プロセッサにスケジュールされた場合  
 外部プロセッサにスケジュールされたタスクの中から任意の  $w$  を選び、それを  $v$  のスケジュールにする。

□

よって、各タスクの処理時間を 1 と仮定した場合におけるタスクスケジューリングアルゴリズムは、各タスクの処理時間を正整数とした場合に拡張できる。

### 3.3 本研究のアルゴリズム

本節では、DAG の高さが 4 の場合における近似精度  $\frac{49}{26} (= 1.88)$  のアルゴリズムを示す。以下では、タスクの集合  $X$  をプロセッサにスケジュールするとは、 $X$  のすべての要素を下界  $e$ -value

の値の小さい順にプロセッサにスケジュールすることを表す。

ここで、補題 1 より、 $v \in V$  の処理時間を 1 と仮定する。また、以下では  $opt \geq low$  の場合について述べる。

**補題 2** 入力の DAG の高さが 0, 1 のとき、最適スケジュールが可能。

**証明)** DAG の高さが 0, 1 のそれぞれについて、最適スケジュールが可能であることを示す。

- 高さが 0 のとき、 $v^*$  をメインプロセッサに時刻 0 からスケジュールする。
- 高さが 1 のとき、 $f$ -value が  $low$  よりも大きいタスクを時刻 0 からメインプロセッサにスケジュールし、その他のタスクは時刻 0 から外部プロセッサにスケジュールする。 $v^*$  は時刻  $low$  にメインプロセッサにスケジュールする。このスケジュールが最適であることは、 $e$ -value の定義より明らか。

□

ここで、頂点集合  $U_1$  を以下のように定義する。

$$U_1 = \{u \mid f(u) > low\}$$

$$l = |U_1|$$

また、 $U_1$  の要素を  $e$ -value の降順に並べたものを  $u_1, u_2, \dots, u_l$  とする。したがって、 $e(u_1) \geq e(u_2) \geq \dots \geq e(u_l)$  が成立する。このとき、定義 2 より、 $v^*$  の下界は  $\max_{1 \leq i \leq l} (e(u_i) + i)$  である。ここで、

$$\lfloor 0.4e(u_j) \rfloor + j = \max_{1 \leq i \leq l} (\lfloor 0.4e(u_i) \rfloor + i)$$

と定義する。

**補題 3**  $low + \lfloor 0.4e(u_j) \rfloor + j$  は  $v^*$  の上界となる。

**証明)**  $U_1$  の要素  $u$  の親  $w$  ( $f(w) \leq low$ ) について考える。

- (1)  $level(w) = 0, 1$  のタスクは、補題 2 より、最適なスケジューリングが可能のため、これらのタスクの実行結果は時刻  $low$  にはメインプロセッサへの通信が完了している。
- (2)  $level(w) = 2$  のタスクは、近似精度 1.4 でのスケジューリングが可能であり [6],  $u$  はメインプロセッサに時刻  $\lfloor 1.4e(w) \rfloor + \tau(w) + 1$  以降にスケジュール可能となる。ここで、 $f(w) = e(w) + \tau(w) + 1 \leq low$ ,  $e(w) \leq e(u)$  より、 $\lfloor 1.4e(w) \rfloor + \tau(w) + 1 = f(w) + \lfloor 0.4e(w) \rfloor \leq low + \lfloor 0.4e(u) \rfloor$  が成立し、 $u$  のスケジュール時刻の上界は  $low + \lfloor 0.4e(u) \rfloor$  となる。

(3) レベル 3 のタスクは、近似精度  $\frac{5}{3}$  でスケジューリングが可能であるが [8], この結果を必要とするタスクは  $v^*$  のみであり, このような  $w$  は存在しない.

以上より,  $u_1, u_2, \dots, u_j$  を, 時刻  $low + \lfloor 0.4e(u_j) \rfloor$  以降に  $u_j, u_{j-1}, \dots, u_1$  と連続してスケジューリングする. また,  $u_l, u_{l-1}, \dots, u_{j+1}$  を, 時刻  $low + \lfloor 0.4e(u_j) \rfloor$  より前に  $u_{j+1}, u_{j+2}, \dots, u_l$  と連続してスケジューリングする. 任意の  $u_i (1 \leq i \leq l)$  に対して,  $\lfloor 0.4e(u_j) \rfloor + j \geq \lfloor 0.4e(u_i) \rfloor + i$  より,  $u_i$  のスケジューリング時刻  $low + \lfloor 0.4e(u_j) \rfloor + j - i \geq low + \lfloor 0.4e(u_i) \rfloor$  が成立する. よって,  $u_i$  はそれぞれ  $u_j$  の前後に連続してスケジューリング可能であり,  $low + \lfloor 0.4e(u_j) \rfloor + j$  は  $v^*$  の上界となる.  $\square$

このときの様子を図 3.2 に示す.

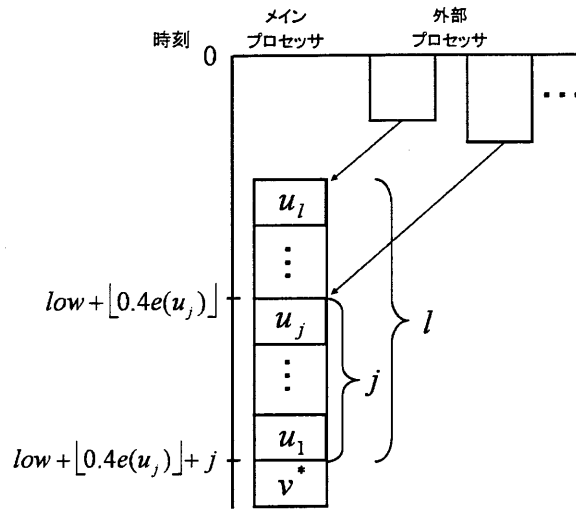


図 3.2: 補題 3 :  $v^*$  の上界  $low + \lfloor 0.4e(u_j) \rfloor + j$

#### 補題 4

$$\begin{aligned} e(u_j) &\geq \frac{5}{3}(low - \lfloor \alpha low \rfloor) \\ \alpha &\geq \frac{5}{3} \end{aligned}$$

のとき近似精度  $(1 + \alpha)$  でスケジューリング可能.

証明) 補題 3 より,  $u_j$  を時刻  $low + \lfloor 0.4e(u_j) \rfloor$  にスケジューリングし, その他の  $U_1$  に含まれるタスクを  $u_j$  の前後に連続してスケジューリングするとき,  $v^*$  の上界は  $low + \lfloor 0.4e(u_j) \rfloor + j$  となる. ここで,  $\lfloor 0.4e(u_j) \rfloor + j > \lfloor \alpha low \rfloor$  と仮定すると  $0.4e(u_j) + j \geq \lfloor 0.4e(u_j) \rfloor + j > \lfloor \alpha low \rfloor$ ,  $0.6e(u_j) \geq low - \lfloor \alpha low \rfloor$  より,  $e(u_j) + j > low$  となり,  $e(u_j) + j \leq low$  に矛盾する. よって,  $low + \lfloor 0.4e(u_j) \rfloor + j \leq low + \lfloor \alpha low \rfloor$  となる. また, レベル 3 のタスクは近似精度  $\frac{5}{3}$  でスケジューリング可能であり [8], これらのタスクの実行結果は時刻  $\frac{5}{3}low$  にはメインプロセッサへの通信が

完了している。したがって、 $\alpha \geq \frac{5}{3}$  ならば、 $v^*$  のスケジュール時刻の上界は  $low + \lfloor \alpha low \rfloor$  となり、近似精度  $(1 + \alpha)$  でスケジュール可能。□

以下では、 $\alpha \geq \frac{5}{3}$  と仮定してスケジューリングを行う。

#### 補題 5

$$\begin{aligned} e(u_j) &< \frac{5}{3}(low - \lfloor \alpha low \rfloor) \\ l &\leq \frac{5}{3}\lfloor \alpha low \rfloor - \frac{2}{3}low + 1 \end{aligned}$$

のとき近似精度  $(1 + \alpha)$  でスケジュール可能。

証明) 補題 4 と同様に、 $u_j$  を時刻  $low + \lfloor 0.4e(u_j) \rfloor$  にスケジュールし、その他の  $U_1$  に含まれるタスクを  $u_j$  の前後に連続してスケジュールするとき、 $v^*$  の上界は  $low + \lfloor 0.4e(u_j) \rfloor + j$  となる。また、 $j \leq l$  であるので、

$$\begin{aligned} low + \lfloor 0.4e(u_j) \rfloor + j &< low + \lfloor \frac{2}{5} \frac{5}{3}(low - \lfloor \alpha low \rfloor) \rfloor + l \\ low + \lfloor 0.4e(u_j) \rfloor + j &\leq low + \frac{2}{3}(low - \lfloor \alpha low \rfloor) + l - 1 \\ &= low + \lfloor \alpha low \rfloor \end{aligned}$$

となり、近似精度  $(1 + \alpha)$  でスケジュール可能。□

補題 4, 5 のときの様子を図 3.3 に示す。ここで、整数  $t (0 \leq t < low)$  を考え、頂点集合  $U_2$  を以

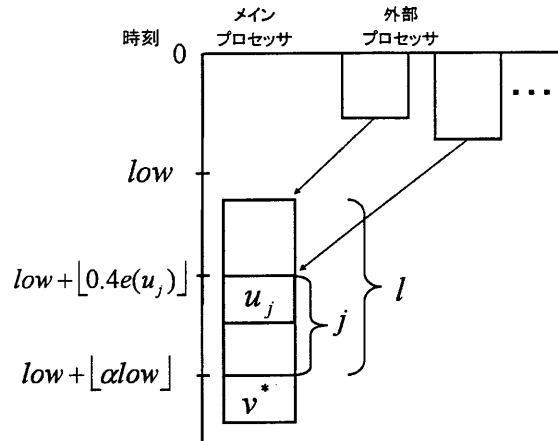


図 3.3: 補題 4, 補題 5 : 近似精度  $(1 + \alpha)$  のスケジュール

下のように定義する。

$$U_2 = \{u \mid level(u) = 0, low \geq f(u) > t\}$$

$\mathcal{M}_1$  を  $U_1$  と  $U_2$  の最大マッチングと定義する。

$$M_1 = \{u \mid (u, u') \in \mathcal{M}_1\}, \quad M'_1 = \{u' \mid (u, u') \in \mathcal{M}_1\}$$

$$m_1 = |M_1|, \quad N_1 = U_1 \setminus M'_1$$

このときの様子を図 3.4 に示す.

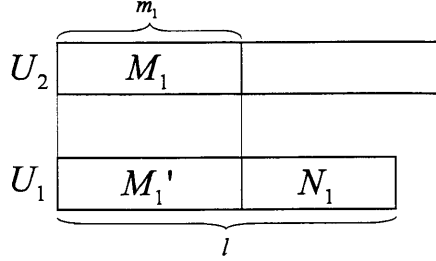


図 3.4: 最大マッチング  $\mathcal{M}_1$

図 3.4 において,  $U_2$  に含まれていて  $N_1$  の先祖であるタスクは,  $M_1$  にしか存在しない. よって,  $M_1 \cup \{u \mid t \geq f(u)\} \cup \{u \mid 2 \geq \text{level}(u) \geq 1\}$  に含まれるタスクの実行結果があれば,  $N_1$  に含まれるタスクは実行を開始できる.

#### 補題 6

$$\begin{aligned} l &\geq \frac{5}{3} \lfloor \text{allow} \rfloor - \frac{2}{3} \text{low} + 2 \\ m_1 &> 2\text{low} - t - l - 1 \end{aligned}$$

のとき,  $\text{opt} \geq \text{low} + 1$  が成立する.

証明)  $M_1$  をメインプロセッサにスケジュールする場合と, 外部プロセッサにスケジュールする場合に分けて考える.

- $M'_1 \subseteq M_1$  を時刻 0 からメインプロセッサにスケジュールする場合, メインプロセッサには  $M'_1$  以外に  $U_1$  のタスクを  $l$  個スケジュールする必要があるので,  $|M'_1| > \text{low} - l$  であれば  $\text{opt} \geq \text{low} + 1$  となる.
- $M'_1 \subseteq M_1$  を外部プロセッサにスケジュールする場合,  $M'_1$  のマッチング相手がメインプロセッサに時刻  $t+1$  以降にスケジュールされるため,  $|M'_1| > \text{low} - t - 1$  であれば  $\text{opt} \geq \text{low} + 1$  となる.

以上より,  $m_1 > \text{low} - l + \text{low} - t - 1 = 2\text{low} - t - l - 1$  であれば  $\text{opt} \geq \text{low} + 1$  が成立する.  $\square$

ここで, 頂点集合  $U_3$  を以下のように定義する.

$$U_3 = \{u \mid 2 \geq \text{level}(u) \geq 1, \text{low} \geq f(u) > t\}$$

また,  $\mathcal{M}_2$  を  $U_3$  と  $N_1$  の最大マッチングと定義する.

$$M_2 = \{u \mid (u, u') \in M_2\}, \quad M'_2 = \{u' \mid (u, u') \in M_2\}$$

$$m_2 = |M_2|, \quad N_2 = N_1 \setminus M'_2$$

このときの様子を図 3.5 に示す.

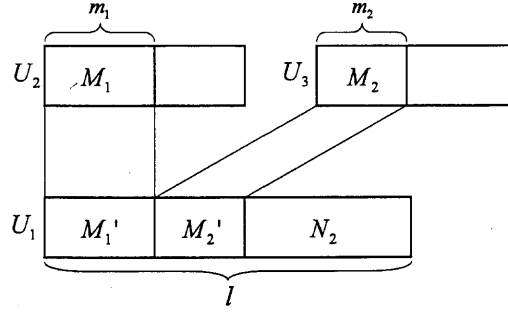


図 3.5: 最大マッチング  $M_2$

図 3.5 において,  $U_2 \cup U_3$  に含まれていて  $N_2$  の先祖であるタスクは,  $M_1 \cup M_2$  にのみ存在する. よって,  $M_1 \cup M_2 \cup \{u \mid 2 \geq \text{level}(u) \geq 0, t \geq f(u)\}$  に含まれるタスクの実行結果があれば,  $N_2$  に含まれるタスクは実行を開始できる. また,  $U_2$  に含まれていて  $M_2$  の先祖であるタスクは,  $M_1$  にのみ存在する. よって,  $M_1 \cup \{u \mid 1 \geq \text{level}(u) \geq 0, t \geq f(u)\}$  に含まれるタスクの実行結果があれば,  $M_2$  に含まれるタスクは実行を開始できる.

補題 7

$$l \geq \frac{5}{3} \lfloor \alpha \text{low} \rfloor - \frac{2}{3} \text{low} + 2$$

$$m_1 + m_2 > 2\text{low} - t - l - 1$$

のとき,  $\text{opt} \geq \text{low} + 1$  が成立する.

証明) 補題 6 の  $M'_1$  の定義を,  $M'_1 \subseteq M_1 \cup M_2$  とすることで, 補題 6 と同様に証明できる.  $\square$

補題 8

$$l \geq \frac{5}{3} \lfloor \alpha \text{low} \rfloor - \frac{2}{3} \text{low} + 2$$

$$m_1 + m_2 \leq 2\text{low} - t - l - 1$$

$$m_2 \leq \lfloor \alpha \text{low} \rfloor + \text{low} - t - l$$

のとき, 以下の式が成立するならば近似精度  $(1 + \alpha)$  でスケジュール可能.

$$2\text{low} - t - l - 1 \leq \lfloor \alpha \text{low} \rfloor + \text{low} - l \quad (3.1)$$

$$\lfloor \alpha \text{low} \rfloor + \text{low} - l > \lfloor 1.4t \rfloor \quad (3.2)$$

$$|N_2| \geq l - \lfloor \alpha \text{low} \rfloor + \lfloor \frac{2}{3}(\text{low} - \lfloor \alpha \text{low} \rfloor) \rfloor - 1 \quad (3.3)$$

証明)  $M = M'_1 \cup M'_2$  と定義し,  $M$  に含まれるタスクを  $e$ -value の降順に並べたものを  $w_1, w_2, \dots, w_{|M|}$  とする. したがって,  $e(w_1) \geq e(w_2) \geq \dots \geq e(w_{|M|})$  が成立する. ここで,

$$\lfloor 0.4e(w_k) \rfloor + k = \max_{1 \leq i \leq |M|} (\lfloor 0.4e(w_i) \rfloor + i)$$

と定義する.

- $M_1$  を時刻 0 からメインプロセッサにスケジュールする.
- $m_1 > t$  のとき,  $M_2$  は時刻  $m_1$  からメインプロセッサにスケジュールする.
- $m_1 \leq t$  のとき,  $M_2$  は時刻  $t$  からメインプロセッサにスケジュールする.

式 (3.1) より,  $m_1 + m_2 \leq 2low - t - l - 1 \leq \lfloor \alpha low \rfloor + low - l$ ,  $m_2 + t \leq \lfloor \alpha low \rfloor + low - l$  なので,  $M_1$  と  $M_2$  は必ず時刻  $\lfloor \alpha low \rfloor + low - l$  以前にメインプロセッサにスケジュールできる. また,  $M_2$  の親は  $M_1$  か  $\{u \mid 1 \geq level(u) \geq 0, t \geq f(u)\}$  である外部プロセッサにスケジュールされたタスクのどちらかであり, 後者のタスクの場合はその実行結果は時刻  $t$  までにメインプロセッサへの通信が完了している.

- $N_2$  を時刻  $\lfloor \alpha low \rfloor + low - l$  からメインプロセッサにスケジュールする.  $N_2$  の先祖は  $M_1 \cup M_2 \cup \{u \mid 2 \geq level(u) \geq 0, t \geq f(u)\}$  に含まれている. このうち, 外部プロセッサにスケジュールされたレベル 2 のタスクの実行結果は時刻  $\lfloor 1.4t \rfloor$  までにはメインプロセッサへの通信が完了しているので, 式 (3.2) より,  $N_2$  は必ず時刻  $\lfloor \alpha low \rfloor + low - l$  からメインプロセッサにスケジュールできる.
- $M$  を  $N_2$  の次にメインプロセッサにスケジュールする. 式 (3.3) より,  $\lfloor \alpha low \rfloor + low - l + |N_2| \geq low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1$  となり,  $M$  の開始時刻は必ず  $low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1$  以降となる. また, 以下の (1), (2) より  $M$  は必ず時刻  $low + \lfloor \alpha low \rfloor$  以前にスケジュール可能である.

(1)  $e(w_k) \geq \frac{5}{3}(low - \lfloor \alpha low \rfloor)$  のとき,  $M$  の要素  $w_i (1 \leq i \leq |M|)$  は時刻  $low + \lfloor 0.4e(w_i) \rfloor$  以降にスケジュールする. このとき,

$$\begin{aligned} low &\geq e(w_k) + k = 0.6e(w_k) + 0.4e(w_k) + k \geq 0.6e(w_k) + 0.4e(w_i) + i \\ &\geq (low - \lfloor \alpha low \rfloor) + 0.4e(w_i) + i \\ low + \lfloor \alpha low \rfloor &\geq low + \lfloor 0.4e(w_i) \rfloor + i \end{aligned}$$

となり,  $M$  は必ず時刻  $low + \lfloor \alpha low \rfloor$  以前にスケジュール可能.

(2)  $e(w_k) < \frac{5}{3}(low - \lfloor \alpha low \rfloor)$  のとき,  $M$  の要素  $w_i (1 \leq i \leq |M|)$  は時刻  $low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1$  以降にスケジュールする. このとき,  $w_i$  のスケジューリング時刻は

$$\begin{aligned} low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1 + |M| - i &\leq low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1 + |M| \\ &\leq low + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1 \\ &\quad + (\lfloor \alpha low \rfloor - \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor + 1) \\ &\leq low + \lfloor \alpha low \rfloor \end{aligned}$$

となり,  $M$  は必ず時刻  $low + \lfloor \alpha low \rfloor$  以前にスケジュール可能.

- 残りのタスクは全て外部プロセッサにスケジュールする.

このようにスケジュールすることで,  $v^*$  のスケジュール時刻の上界は  $low + \lfloor \alpha low \rfloor$  となり, 近似精度  $(1 + \alpha)$  でスケジュール可能.

□

このときの様子を図 3.6 に示す.

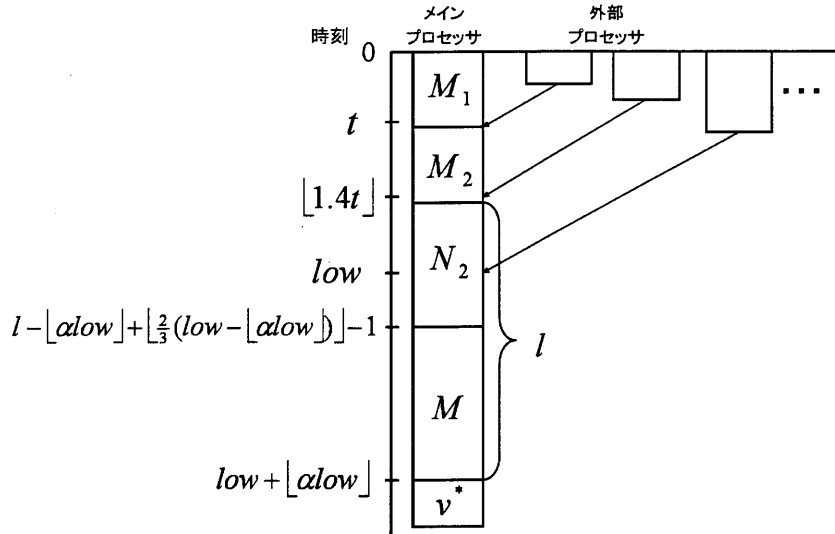


図 3.6: 補題 8 : 近似精度  $(1 + \alpha)$  のスケジュール

ここで,  $M_3$  を  $M_2$  と  $N_2$  の最大マッチングと定義する.

$$M_3 = \{u \mid (u, u') \in M_3\}, \quad M'_3 = \{u' \mid (u, u') \in M_3\}$$

$$m_3 = |M_3|, \quad N_3 = N_2 \setminus M'_3$$

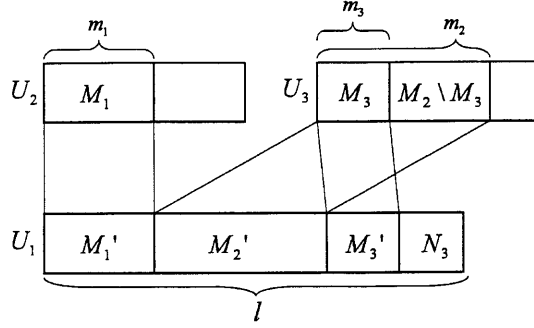


図 3.7: 最大マッチング  $M_3$

このときの様子を図 3.7 に示す.

補題 9

$$\begin{aligned}
 l &\geq \frac{5}{3} \lfloor \alpha low \rfloor - \frac{2}{3} low + 2 \\
 m_1 + m_2 &\leq 2low - t - l - 1 \\
 m_2 &> \lfloor \alpha low \rfloor + low - t - l \\
 m_3 &\leq \lfloor \alpha low \rfloor + low - t - l \\
 |N_3| &\geq l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1
 \end{aligned}$$

のとき, 近似精度  $(1 + \alpha)$  でスケジューリング可能.

証明)  $M_2 \setminus M_3$  に含まれるタスクは  $N_3$  と依存関係はなく, これらは外部プロセッサにスケジューリングする. そして, 補題 8 の  $M_2$  を  $M_3$  とし,  $M = M_1' \cup M_2' \cup M_3'$  とすることで, 補題 8 と同様に証明できる.  $\square$

補題 10

$$\begin{aligned}
 l &\geq \frac{5}{3} \lfloor \alpha low \rfloor - \frac{2}{3} low + 2 \\
 m_1 + m_2 &\leq 2low - t - l - 1 \\
 m_2 &> \lfloor \alpha low \rfloor + low - t - l \\
 m_3 &\leq \lfloor \alpha low \rfloor + low - t - l \\
 |N_3| &< l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 1
 \end{aligned}$$

のとき, 以下の式が成立するならば  $opt \geq low + 1$  が成立する.

$$t \geq 3low - 2l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 2 \quad (3.4)$$

証明)  $M_1$ ,  $M_3$ ,  $M_2 \setminus M_3$  に含まれるタスクの, 子の数を考える.  $M_3$  に含まれるタスクは,  $M'_2$  と  $M'_3$  にそれぞれ1つ以上の子を持つ. ここで,  $M_3$  に含まれるタスクを  $low - l$  個メインプロセッサにスケジュールし, その他のタスクを外部プロセッサに割り当てる. このとき,

$$\begin{aligned}
2(m_3 - (low - l)) + m_1 + (m_2 - m_3) &> low - (t + 1) \\
m_1 + m_2 + m_3 &> 3low - 2l - t - 1 \\
l - |N_3| &> 3low - 2l - t - 1 \\
\lfloor \alpha low \rfloor - \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor + 2 &> 3low - 2l - t - 1 \\
t &> 3low - 2l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 3 \\
t &\geq 3low - 2l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 2
\end{aligned}$$

が成立すれば,  $opt \geq low + 1$  が成立する.  $\square$

#### 補題 11

$$\begin{aligned}
l &\geq \frac{5}{3} \lfloor \alpha low \rfloor - \frac{2}{3} low + 2 \\
m_1 + m_2 &\leq 2low - t - l - 1 \\
m_2 &> \lfloor \alpha low \rfloor + low - t - l \\
m_3 &> \lfloor \alpha low \rfloor + low - t - l
\end{aligned}$$

のとき, 以下の式が成立するならば  $opt \geq low + 1$  が成立する.

$$t \leq 2 \lfloor \alpha low \rfloor - low + 1 \quad (3.5)$$

証明) 補題 10 と同様のスケジューリングを行い,  $m_1 = 0$  の場合を考えると,  $m_2 + m_3 > 3low - 2l - t - 1$  であり,  $m_2 + m_3 > 2(\lfloor \alpha low \rfloor + low - t - l)$  であるので,

$$\begin{aligned}
2(\lfloor \alpha low \rfloor + low - t - l) &\geq 3low - 2l - t - 1 \\
1 - low + 2 \lfloor \alpha low \rfloor &\geq t
\end{aligned}$$

が成立すれば,  $opt \geq low + 1$  が成立する.  $\square$

補題 12  $\alpha \geq \frac{23}{26}$  のとき, 式 (3.2) の範囲は式 (3.5) の範囲を真に含む.

証明) 式 (3.2), 式 (3.5) よりそれぞれ,

$$\begin{aligned}
\lfloor \alpha low \rfloor + low - l - 1 &\geq \lfloor 1.4t \rfloor \\
1.4(2 \lfloor \alpha low \rfloor - low + 1) &\geq \lfloor 1.4t \rfloor
\end{aligned}$$

となり、これらの左辺を比較すると、 $\alpha \geq \frac{23}{26}$  のとき、

$$\begin{aligned} 1.4(2\lfloor \alpha low \rfloor - low + 1) - (\lfloor \alpha low \rfloor + low - l - 1) &> \frac{2}{15}low(26\alpha - 23) + \frac{14}{15} \\ &> 0 \end{aligned}$$

となる。  $\square$

**補題 13**  $\alpha \geq \frac{23}{26}$  のとき、近似精度  $(1+\alpha)$  でスケジュール可能、または  $opt \geq low + 1$  が成立する。

**証明** 補題 4, 補題 5, 補題 11 より、補題 8 の式 (3.1), 式 (3.2), 式 (3.3), 補題 10 の式 (3.4), 補題 11 の式 (3.5) が成立する場合に、近似精度  $(1+\alpha)$  でスケジュール可能、または  $opt \geq low + 1$  が成立する。補題 12 より、式 (3.2) の範囲は式 (3.5) の範囲を真に含み、式 (3.4) の範囲は式 (3.1), 式 (3.3) の範囲を真に含むため、式 (3.2), 式 (3.4) について考える。この 2 式を満たす  $\alpha$  を求めると、

$$\begin{aligned} \frac{5}{7}(\lfloor \alpha low \rfloor + low - l) - 1 &> 3low - 2l - \lfloor \alpha low \rfloor + \lfloor \frac{2}{3}(low - \lfloor \alpha low \rfloor) \rfloor - 2 \\ \alpha low &> \frac{16}{19}(low - \frac{19}{20}) \\ \alpha &> \frac{16}{19} \end{aligned}$$

となり、 $\frac{23}{26} > \frac{16}{19}$  より、 $\alpha \geq \frac{23}{26}$  のとき、近似精度  $(1+\alpha)$  でスケジュール可能、または  $opt \geq low + 1$  が成立する。  $\square$

**定理 1** 入力の DAG の高さが 4 のとき、近似精度  $\frac{49}{26}$  のアルゴリズムが存在する。

**証明** 補題 13 より、 $\alpha = \frac{23}{26}$  のとき、近似精度  $\frac{49}{26}$  でスケジュール可能、または  $opt \geq low + 1$  が成立する。後者の場合は、 $low$  の値を 1 増やしたものを新しい  $low$  とし、補題 13 を繰り返し適用することで、定理 1 は成立する。  $\square$

### 3.4 本手法の時間計算量

本手法の時間計算量について述べる。

全タスクの e-value の計算は  $O(|V|^2(|A| + |V| \log |V|))$  で実行でき、DAG の変換は  $O(|V|)$  ができる。頂点集合  $U_1, U_2, U_3$  をとるのにそれぞれ  $O(|V|)$  かかり、二つの頂点集合の最大マッチングを計算するのは  $O(|V||A|)$  ができる。下界が改善され、再帰的に本手法が適用される回数は高々  $O(|V|)$  回であり、全体として  $O(|V|^2(|A| + |V| \log |V|))$  となる。

### 3.5 従来手法との近似精度の比較

本手法では, Papadimitriou らの手法 [1] の近似精度 2 を改善した近似精度  $\frac{49}{26}$  のアルゴリズムが得られた.

## おわりに

本研究では、DAG の高さが 4 の場合において未解決問題であった近似精度が 2 よりも良いアルゴリズムがあるかどうかの問題に取り組み、近似精度 2 よりも良い近似精度  $\frac{49}{26}$  を持つ近似アルゴリズムを与えた。本アルゴリズムでは、これまでの高さ 3 以下の DAG に対するスケジューリングでは問題とならなかった、メインプロセッサに割り当てた  $v^*$  以外のタスクに対する、高さ 2 のタスクからの通信を考慮したスケジューリング手法を考案した。

今後の課題として、さらなる近似精度の改善、DAG の高さの制限を除去することなどがあげられる。

## 謝辞

たいへん熱心に御指導いただいた大山口通夫教授，山田俊行講師，ならびに何かとお世話になりました落合美子事務官，また熱心に議論していただいた研究室の学生諸氏に感謝します。

## 参考文献

- [1] C.H.Papadimitriou and M.Yannakakis, "Towards an Architecture-Independent Analysis of Parallel Algorithms", SIAM Journal on Computing, vol.19, no.2, pp.322-328, 1990.
- [2] 河田俊郎, 大山口通夫, 大田義勝, "通信遅延を考慮したタスクスケジューリングアルゴリズムについて", 電子情報通信学会論文誌, Vol.J85-D-I, No.11, pp.1088-1092, 2002.
- [3] 片柳賢二 砂坂明宏 大山口通夫 太田義勝, "タスクスケジューリングに関する新しい近似アルゴリズムについて", 京大数解研講究録, No.1325, pp.185-190, 2003.
- [4] 新美信之助 大山口通夫 太田義勝 山本浩平, "最大マッチングを利用したタスクスケジューリングアルゴリズムの近似度の改善について", 京大数解研講究録, No.1426, pp.184-188, 2005.
- [5] 小松健悟 柳本貴之 大山口通夫 山田俊行, "DAGの高さを2に制限したタスクスケジューリングアルゴリズムについて", 2005年度電気関係学会東海支部連合大会 講演論文集, O-283, 2005.
- [6] 小松健悟, "タスクスケジューリングアルゴリズムに関する研究-DAGの高さ2の場合-", 三重大学, 修士論文, 2006.
- [7] 柳本貴之, "タスクスケジューリングアルゴリズムに関する研究-DAGの高さ3の場合-", 三重大学, 修士論文, 2006.
- [8] 清水豪樹 大山口通夫 山田俊行, "DAGの高さを3に制限したタスクスケジューリングの近似アルゴリズムについて", 2006年度電気関係学会東海支部連合大会 講演論文集, O-439, 2006.