並列度・タスク実行時間の偏りを考慮した標準タスクグラフセット STG Ver3 を用いたスケジューリングアルゴリズムの評価

島岡 護† 今泉 和浩† 鷹野 芙美代†
木村 啓二† 笠原 博徳†

本稿では強 NP 困難な組み合わせ最適化問題である実行時間最小マルチプロセッサスケジューリング問題のための標準タスクグラフセット STG Ver3 を提案するとともに，それを用いたアルゴリズムの性能評価について述べる．STG Ver2 はタスク実行時間生成乱数，先行制約形状生成乱数により生成されたタスクグラフセットである．STG Ver3 は STG Ver2 にタスクの並列度，タスク実行時間の偏差に考慮を加えることにより生成した“並列度セット”，“正規化偏差セット”により構成される．評価の結果，最適化アルゴリズム DF/IHS (Depth First/ Implicit Heuristic Search) では 87.25%, PDF/IHS (Parallelized DF/IHS) では 92.25% の問題で 10 分以内に最適解を得られることを確認した．

Performance Evaluation of Minimum Execution Time Multiprocessor Scheduling Algorithms Using Standard Task Graph Set Ver3 Consider Parallelism of Task Graphs and Deviation of Task Execution Time

Mamoru Shimaoka† , Kazuhiro Imaizumi† , Fumiyo Takano† ,
Keiji Kimura† and Hironori Kasahara†

This paper proposes the “Standard Task Graph Set Ver3”(STG Ver3) to evaluate performance of heuristic and optimization algorithms for the minimum execution time multiprocessor scheduling problem. The minimum execution time multiprocessor scheduling problem is known as a strong NP-hard combinational optimization problem to the public. The STG Ver2 was created by random task execution times and random predecessors. In addition, the STG Ver3 considers parallelism of task graphs and deviation of task execution times to let us understand characteristics of algrithms. This paper describes evaluation results by applying the STG Ver3 to several algorithms. Performance evaluation show that DF/IHS can give us optimal solutions for 87.25%, and PDF/IHS 92.25% within 600 seconds.

1. はじめに

マルチコアやマルチプロセッサシステム上で効率よく負荷分散を行う為には，処理するタスク集合のプロセッサへの割り当て方法と実行順序を決定しなければならない[1)2)3)]．このマルチプロセッサスケジューリング問題は，タスク実行時間，先行制約，割り当てプロセッサ数に制限を加えない場合，強 NP 困難であることが知られており，組み合わせ最適化問題の中でも最も難しい問題の一つである．

この問題を解くために HLFET (Highest Levels First with Estimated Times)[4)]，CP (Critical Path)[1)]，CP / MISF (Critical Path / Most Immediate Successor First)[2)] など多くのヒューリスティックスケジューリングアルゴリズムが提案されている．またヒューリスティックアルゴリズムでは必ずしも最適解が得られないため，最適解を求める最適化アルゴリズムとして，Ramamoorthy らが DP (Dynamic Programming) を用いてプロセッサ数 2 台，タスク数を約 40 個として最適解を得た[5)] ほか，筆者らはタスク数が数百の問題の多くを数秒で求解することが出来る実用的最適化アルゴリズム DF/IHS (Depth First / Implicit Heuristic Search)[2)] と，その探索部を並列化した PDF/IHS (Parallelized DF/IHS)[6)] を提案している．

これらのスケジューリングアルゴリズムはそれぞれのアルゴリズムの提案者が用意した問題集を解くことで性能を評価された．そのため，評価に用いられた問題集が提案されたアルゴリズムに対して有利なものである可能性もあり，また他のアルゴリズムとの比較も容易ではなかった．そこで筆者らはアルゴリズムの公正な評価の為のベン

† 早稲田大学基幹理工学研究科 情報理工学専攻
Dept. of Computer Science, Waseda University

チマークを目指して，標準タスクグラフセット (STG) を提案している[7)8)]．この STG を用いた評価により，並列度[9)] やタスク実行時間の偏差がアルゴリズムの性能に大きな影響を与えることが分かった．本稿では，より公平かつアルゴリズムの特徴を捉えた性能評価のために並列度やタスク実行時間の偏差を考慮した STG Ver3 を提案すると共に，STG Ver3 を用いて FIFO，CP，CP/MISF，DF/IHS，PDF/IHS の性能評価を行い，並列度やタスク実行時間の偏差がアルゴリズムの性能に影響を与える原因を調査する．

2. 対象とするスケジューリング問題

本稿で扱うマルチプロセッサスケジューリング問題は，性能の等しい m 台のプロセッサで，任意の n 個のタスクからなるタスク集合 $T = (T_1, T_2, ..., T_n)$ を並列処理する際に，その実行時間 (スケジュール長) を最小にするようなスケジュールを求める問題である．またタスクの処理時間と先行制約形状は任意であり，処理割り込みは認めないものとする．タスク集合 T はタスクグラフと呼ばれる有限無サイクル有向グラフ (DAG) として $G(N, E)$(N は n 個のノード集合，E は有効エッジの集合) で記述される．このタスクグラフは 1 つの入口ノードと 1 つの出口ノードを持ち，全てのノードは入口及び，出口ノードから到達できるものとする．また，この入口ノード及び出口ノードは実行時間の無いダミーノードであり，そのため一般性を失うことは無い．図 1 に 8 個のタスクグラフからなるタスクグラフの例を示す．図中の各ノードは 1 つのタスクを表し，ノード内の数字はタスク番号 i，ノード左上の数字は各タスクの処理時間 t_i(単位時間: unit time) をそれぞれ表す．また，図中の各アークはタスク間の先行制約を表し，ノード i からノード j へのアークは，タスク T_i がタスク T_j に先行するという半順序制約を表す．

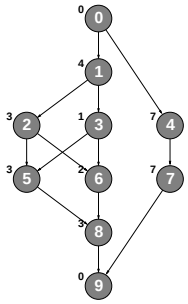

図 1 タスクグラフの例

3. マルチプロセッサスケジューリングアルゴリズム

本章では，本稿で性能評価を行うヒューリスティックアルゴリズム FIFO, CP, CP/MISF 及び最適化アルゴリズム DF/IHS，PDF/IHS について述べる．

3.1 ヒューリスティックアルゴリズム

本節では本稿で性能評価を行うヒューリスティックアルゴリズムについて述べる．FIFO (First In First Out) は，先行タスクの処理が終わり，レディ状態 (実行可能状態) になったタスクをレディキューに入れ，キューの最初のタスクから順に空きプロセッサに割り当てるスケジューリングアルゴリズムである．各タスクの CP 長はグラフ上で出口ノードからタスクまでの最長のパス長である．CP は CP 長が長いタスクから順に割り当て優先リストを作成し，レディ状態のタスクの中で最も優先度の高いタスクから順に空きプロセッサに割り当てるリストスケジューリングアルゴリズムである．また，CP では同一 CP 長を持つタスクが複数ある際の優先順位を一意に決定できないため，優先リスト作成時に同一 CP 長を持つタスクが複数ある場合には，直接後続タスク数の多いタスクを優先する手法が CP/MISF である[2)]．

3.2 逐次最適化アルゴリズム DF/IHS

本節では逐次最適化アルゴリズム DF/IHS について述べる．DF/IHS は，前処理部と分枝限定法による探索部から構成される．

前処理部では CP/MISF の割り当て優先度が高い順，すなわち長い CP 長を持つタスクのタスク番号が小さくなるようにタスク番号を付け替える．それにより探索部ではタスク番号の小さいタスクを割り当てるノードを先に探索するだけで，CP/MISF 優先順位にかなった解から先に探索することが出来る．

探索部は深さ優先探索と分枝限定操作を行う．限定操作は，探索中のノードから得られる可能性のあるスケジューリング長の下限値と探索中に得られた最も優れた解である暫定解を比較する．その結果，暫定解より優れた解が得られる可能性が無ければそのノードの探索をやめ探索木から次に優先順位の高い別のノードを探索する．前処理部により初期解は必ず CP/MISF 解になるため，優れた暫定解を得られる可能性が高く，その暫定解を使い限定操作を行うことで探索時間を大きく削減できる．また DF/IHS では CP 長，未割り当てタスクの実行時間の合計を割り当てプロセッサ数で割った値，そして Fernandez による拡張がされた Hu の下限[10)] の 3 つの下限関数を用いて下限値を計算する[2)]．

3.3 並列最適化アルゴリズム PDF/IHS

本節では並列最適化アルゴリズム PDF/IHS[6)] について述べる．PDF/IHS は，DF/IHS の探索部を並列化した並列最適化アルゴリズムである．PDF/IHS で k 台の PE (Processor Element) を用いて並列探索する場合，DF/IHS と同様の前処理後，PE_1(マスター PE) は DF/IHS と同様に探索木の左，つまり CP/MISF 優先順位の高いノードから右に深さ優先探索を行い，初期解と下限値が一致しない場合その事を他の $PE_j(j = 2, ..., k)$(スレーブ PE) に通知する．通知を受けたスレーブ PE はマスター PE の探索経路上のノードをルートノードとする部分探索木

で未探索かつより探索木のルートに近いノードを右から左に探索する．マスター PE，スレーブ PE による左右からの階層的挟み撃ち探索により，探索木の右側にある最適解も効率よく見つけることが出来る．また，この挟み撃ち探索により部分探索木のプロセッサへの負荷分散を行い低オーバーヘッドで並列探索を行うことが出来る．また，マスター PE が DF/IHS と同様の探索を行うため，同期オーバーヘッド以外の減速異常[11] は生じない．

4. 標準タスクグラフセット Ver3

本稿では標準タスクグラフセット Ver3(STG Ver3) を提案する．STG Ver2[8] は様々なタスク実行時間生成乱数と先行制約生成方法を組み合わせて生成されたタスクグラフセットである．また，問題の規模の変化が性能に与える影響を知るためにタスク数 50 から 5000 までの問題が用意された．STG Ver2 を用いた評価によりタスクグラフの並列度 para とタスクグラフ中のタスク実行時間の標準偏差を平均値で正規化した正規化偏差がスケジューリングアルゴリズムの性能に影響を与えることが分かった．並列度 para は (1) 式で定義される．

$$para = \frac{\text{タスク実行時間の合計}}{\text{タスクグラフの CP 長}} \quad (1)$$

STG Ver3 は並列度 para と正規化偏差に対して問題数を揃えることにより，より公平かつスケジューリングアルゴリズムの特徴を捉えた評価が可能である．STG Ver3 はそれぞれの指標に対して問題数を一様に揃えた “並列度セット” 及び “正規化偏差セット” により構成される．

タスク数は 100, 300, 500, 700, 1000 の 5 種類を用意した．STG Ver2 がタスク数 5000 までのタスク数を用意していたのと比べて STG Ver3 は 1000 より大きなタスク数が無くなっているが，これは STG Ver2 を用いた評価によりタスク数 1000 以上での各種アルゴリズムの性能の変化が小さかったためである．

タスクグラフの形状 (各タスク間の先行制約生成方法) は，samepred, layrpred, linkpred の 3 種を用意した．

- samepred: 各タスクの平均直接先行タスク数を指定して，各タスク間にエッジを生成する．
- layrpred: タスクグラフ中に作成する層 (レイヤ数) を決定し，次に各層にタスクを最低 1 つランダムに割り当てる．あとは samepred と同じように平均直接先行タスク数を指定して，各タスク間にエッジを生成するが，ここで同じ層に割り当てられたタスク同士の間ではエッジは生成しないようにする．
- linkpred: タスクグラフの中からタスクをランダムに 2 つ選び出し，番号の小さいタスクから大きいタスクへエッジを生成する．

各タスクの実行時間の決定方法は一様乱数，指数乱数，正規乱数の 3 種類の乱数を用い，その平均値は 5, 10, 20 とした．

STG Ver3 は並列度セットと正規化偏差セットから構成される．並列度セットでは para の値が，$1.5 \leq para < 2.5, ..., 19.5 \leq para < 20.5$ の各範囲 (計 19 種類) において，タスク数別，形状別，平均処理時間別，実行時間決定方法別にタスクグラフが一つずつとなるように生成する．一方正規化偏差セットでは正規化偏差 Normalized Deviation (ND) の値が，$0.0 \leq ND < 0.2, ..., 0.8 \leq ND < 1.0$ の各範囲 (計 5 種類) において，タスク数別，形状別，平均実行時間別にタスクグラフが一つずつとする．以上から並列度セットは，5(タスク数)*19(para の種類)*3(先行制約形状)*3(平均タスク実行時間)*3 (実行時間決定方法)=2565 通り，正規化偏差セットは 5(タスク数)*5(ND の種類)*3(先行制約形状)*3(平均タスク実行時間)=225 通り，あわせて 2790 通りのタスクグラフにより STG Ver3 は構成され，これを用いて 3 章で述べたアルゴリズムの性能を評価する．

5. STG Ver3 を用いた性能評価

本章では 4 章で述べた STG Ver3 を用いて FIFO, CP, CP/MISF, DF/IHS, PDF/IHS の性能評価を行う．評価には Power5+ 1.5GHz を 8 台，Main Memory を 12GB 搭載した IBM p550Q を使用した．なお，以下では用語の混乱を避けるため，スケジューリング問題の中で使われるプロセッサを単に “プロセッサ”，PDF/IHS の並列探索に使われる IBM p550Q のプロセッサを “PE” と表記する．

5.1 性能評価条件

本評価では，2790 例のタスクグラフを 2, 4, 8 プロセッサにスケジュールする問題，合計 8370 例に対して，FIFO, CP, CP/MISF，DF/IHS，PDF/IHS(PE を 8 台使用) で求解した．ただし，DF/IHS，PDF/IHS での探索上限時間を wall-clock time (プロセス起動後の経過時間) で 600 秒とした．また，各ヒューリスティックアルゴリズムの最適解求解率及び最適解求解数は自身で得られた解が最適解かそうでないか判断出来ない. そのため，各ヒューリスティックアルゴリズムは DF/IHS や PDF/IHS で得られた最適解と同じ値を得る，あるいは DF/IHS の下限値と一致した解を得れば最適解を得たとする．

5.2 性能評価結果

表 1 にタスク数別の最適解求解数を示す．表 1 より，FIFO は全問題の 6.68 %，CP は 59.06 %，CP/MISF は 59.01 %で最適解を得られた．また，探索時間 600 秒以内に DF/IHS は 87.25 %，8PE を用いた PDF/IHS では 92.25 %の問題で最適解が得られた．どのアルゴリズムもタスク数の増加に従い最適解求解数が低下している．タスク数増加による求解数の低下の原因は NP 困難であるスケジューリング問題はタスク数の増加に従い劇的に問題が難しくなるためである．また，DF/IHS，PDF/IHS はヒューリスティックアルゴリズムより 25 %以上高い求解率を得ることが出来，その高い性能が確かめられた．

表 2 に割り当てプロセッサ台数別の最適解求解率を示す．表 2 よりヒューリスティックアルゴリズムは割り当て

表 1　タスク数別最適解求解数

タスク数	問題数	FIFO	CP	CP/MISF	DF/IHS	PDF/IHS
100	1674	161	1088	1106	1544	1636
300	1674	119	1006	999	1476	1569
500	1674	105	954	967	1441	1526
700	1674	90	982	945	1437	1509
1000	1674	84	913	922	1405	1481
total	8370	559	4943	4939	7303	7721
%	100.00	6.68	59.06	59.01	87.25	92.25

プロセッサ数 4 台において最も求解率が低下している．一方，最適化アルゴリズムは割り当てプロセッサ数が増加するに従い，求解率が低下する．

図 2 から図 4 に各アルゴリズムの並列度セットを用いた並列度別求解率を割り当てプロセッサ別に示す．なお図 2，3，4，6 における並列度 $para$ は各タスクグラフの並列度の小数点以下を四捨五入した値である．図 2，3，4 より，各アルゴリズムは並列度が割り当てプロセッサ数と等しくなる問題において求解率が最も低い．また，並列度が割り当てプロセッサ数より低い問題では 100 %の求解率を得ることもあり，並列度が割り当てプロセッサ数よりも高くなれば並列度の上昇に従い求解率は緩やかに高くなる．また，PDF/IHS において DF/IHS と比べて新たに最適解が得られた問題は並列度が割り当てプロセッサ以上の問題に集中している．並列探索は特に並列度が割り当てプロセッサより高い問題に有効ということが分かる．

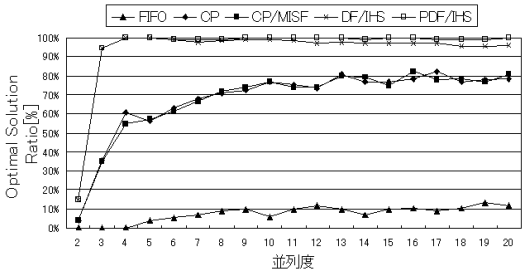

図 2　並列度セット最適解求解率 (プロセッサ台数 2)

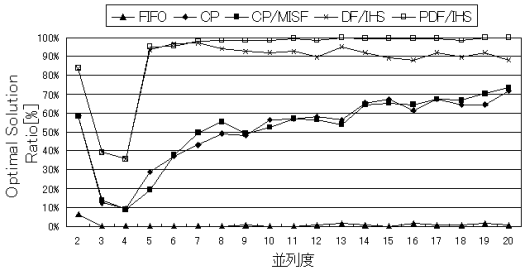

図 3　並列度セット最適解求解率 (プロセッサ台数 4)

図 5 に各アルゴリズムの正規化偏差セットを用いた正規化偏差別求解率を示す．図 5 より CP，CP/MISF，DF/IHS は正規化偏差の小さい問題において求解率が低くなる．DF/IHS と PDF/IHS を比較した結果，PDF/IHS は特に

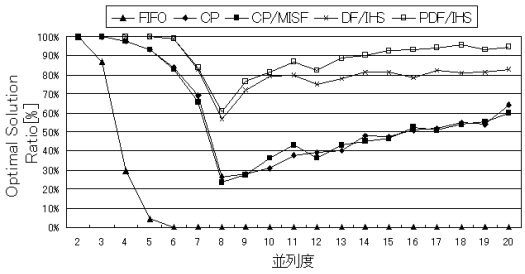

図 4　並列度セット最適解求解率 (プロセッサ台数 8)

正規化偏差の小さい問題に有効であることが分かる．

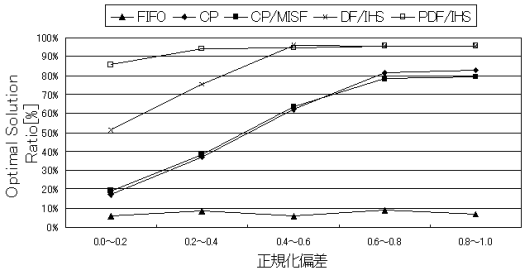

図 5　正規化偏差セット最適解求解率

図 2，3，4 より，ヒューリスティックアルゴリズムは並列度により求解率が変化した．図 4 より並列度が割り当てプロセッサ数よりも十分に低いならば FIFO でも 100 %の求解率を得る．並列度の低い問題において FIFO や CP，CP/MISF などのヒューリスティックアルゴリズムで十分に最適解を得ることが出来る理由はアイドルプロセッサ数が常にレディタスク数を上回り，全てのレディタスクをプロセッサに割り当てる以外の選択肢が存在しないためである．また，CP や CP/MISF は FIFO と較べ並列度が低い問題の中でより高い並列度の問題での求解率が高くなっている．並列度が高くなるとレディタスク数は増え，アイドルプロセッサ数を上回ることが多くなる．その際には CP 長の長いタスクを優先することが効果的ということが分かる．

また，並列度が割り当てプロセッサ数より高い問題で並列度が高くなるほど求解率が高くなる．これは並列度が上昇すると CP 長は短くなり，スケジュール長に対して CP 長が短いと，各タスクは比較的どの時点で実行しても全体のスケジュール長に影響を与えなくなるためであると考

表 2　プロセッサ台数別最適解求解率

プロセッサ台数	問題数	FIFO	CP	CP/MISF	DF/IHS	PDF/IHS
2	2790	7.67	67.46	67.35	93.30	95.13
4	2790	1.00	51.43	51.65	85.70	91.83
8	2790	11.36	58.28	58.03	82.76	89.78
total	100.00	6.68	59.06	59.01	87.25	92.25

えられる．そのためヒューリスティックアルゴリズムは並列度が高くなると求解率が高くなる．また，FIFO は並列度が割り当てプロセッサ数よりも高い問題では求解率がほぼ 0 となった．これは FIFO がレディキューの中から先にキューに入ったタスクを優先して配置するため新たにレディタスクとなった CP 長の長い，早めに配置するべきタスクを後回しにするためである．

図 5 より，CP と CP/MISF は正規化偏差の小さい問題で求解率が低下する．実行時間が平均値を大きく上回るタスクと多くの 1 unit time のタスクからなるタスクグラフの正規化偏差は大きくなる．逆に正規化偏差の小さいタスクグラフは平均と近い実行時間のタスクが大半を占め，実行時間 1 unit time のタスクは少なくなる．1 unit time のタスクが多くあればタスクを全てのプロセッサに均等に分配することは容易になる．リストスケジューリングは最も優先度の高いタスクをアイドルプロセッサに配置していく．アイドルプロセッサへの配置によりプロセッサへのタスクの分配はその時点で最も均等になる．この時点で全てのプロセッサにおいてこれまでのタスクの実行終了時間が等しく，全てがアイドルプロセッサになるとする．残り未割り当てのタスクの実行時間がほぼ等しく，その個数が割り当てプロセッサ数の倍数で無い場合，均等な分配は不可能となる．未割り当てタスクの実行時間が 1 unit time であればスケジュール長は最適解となるが，そうでなければスケジュール長は最適解とはならない．そのため正規化偏差の小さい問題で CP と CP/MISF の求解率が低下すると考えられる．

最適化アルゴリズムの求解率も並列度により変化する．最適化アルゴリズムである DF/IHS と PDF/IHS の性能を左右する最大の要素は下限値である．そのため DF/IHS と PDF/IHS 共通で使用している下限関数の精度の観点から DF/IHS 及び PDF/IHS の求解率を評価する．

図 6 に並列度セットにおいて全てのタスクを未割り当ての時点での DF/IHS と PDF/IHS の下限値と最適解が一致していない割合を並列度別割り当てプロセッサ別に示す．なお，図 6 及び図 7 におけるタスクグラフの母集合は DF/IHS あるいは PDF/IHS で最適解が得られた問題とする．

DF/IHS，PDF/IHS で使用している下限関数の中で最も精度が高いものは Fernandez により拡張された Hu の下限[10] である．Hu の下限は他の 2 つより必ず良い精度が得られ，特に並列度と割り当てプロセッサ数が近い問題で効果を発揮する．ところが図 6 より最適解と下限値の誤差が生じる問題は並列度と割り当てプロセッサ数が近い問題にのみ存在する．つまり Hu の下限の精度が高い事を期待している問題において Hu の下限の精度が十分でないことがわかる．これは 600 秒以内に最適解を得ることが出来た問題の中での結果であるから，最適解を得ることが出来ていない問題の中には更に多くの下限値の精度が十分でない問題が含まれることが予想される．この原因および解決策はすでに Fernandez により提案されている[10]．DF/IHS で採用している Fernandez による拡張がされた Hu の下限はタスクの最遅実行条件 (タスクグラフを CP 長以内でスケジューリングするためにあるタスクを実行していなければならないリミット) のみから算出されている．だが最早実行条件 (あるタスクを最も早く実行できる時間) を考慮していない．そのため下限値の計算において一部の先行制約が無視されてしまう．Fernandez は最遅実行条件，最早実行条件を共に考慮した下限値を提案しているが，計算量が $O(n^2)$ となることから，最早実行条件を無視した Hu の下限を代替案として提案した．代替案の計算量は $O(n)$ となり，DF/IHS ではこれを採用している．そのため DF/IHS，PDF/IHS において並列度と割り当てプロセッサ数が近い問題の求解率を向上させるには最遅実行条件，最早実行条件を共に考慮したうえで計算量も少ない下限値が必要となる．

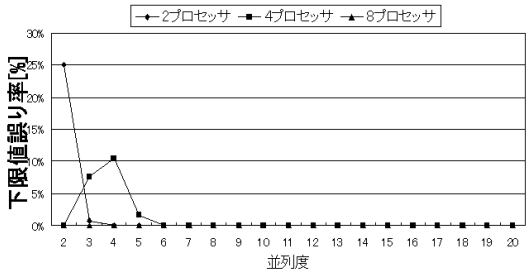

図 6　並列度セット下限値誤り率

次に正規化偏差の小さい問題で DF/IHS の求解率が低下する原因を考察する．図 7 に正規化偏差セットにおいてタスクを全く未割り当ての時点での DF/IHS と PDF/IHS の下限値と最適解が一致している割合を正規化偏差別に示す．図 7 より正規化偏差の値に関わらずタスク未割り当て時の下限値の誤りは存在しない．これらの問題はタスク未割り当て時の下限値がおそらく正しいこと，PDF/IHS の求解率が高くなることから DF/IHS の求解率の低下には 2 つの原因が考えられる．一つは実行時間が同じタスクが多いことから，DF/IHS が同じようなスケジューリングと

なるノードを探索し続けるためである．一方 PDF/IHS はスレーブ PE が全く別のノードを探索しているため，新たなスケジュール長の解に素早く辿り着くことができる．もう一つの原因は未割り当てタスクが少ない時の下限値の精度が低いためと考えられる．タスクが少なく，またタスクの大きさが近い場合，どうしてもプロセッサへの均等な分配は不可能な事がある．並列度が高い問題に対しては下限関数は均等な分配が可能として下限値を算出するために誤差が生じてしまう．よって正規化偏差の小さい問題で DF/IHS の求解率を向上させるには実行時間と先行タスク，後続タスクが等しいタスクの順序の制御，あるいは正規化偏差の小さい問題に適した下限関数が必要である．

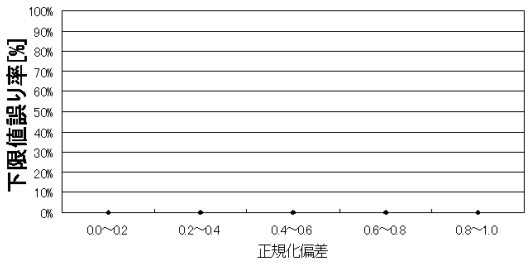

図 7 正規化偏差セット下限値誤り率

6. ま と め

本稿では，強 NP 困難な最適化問題である実行時間最小マルチプロセッサスケジューリング問題に対するアルゴリズムの性能評価のために，並列度およびタスク実行時間の偏差を考慮した標準タスクグラフセット (STG Ver3) を提案すると共に，それを用いたヒューリスティックアルゴリズム FIFO, CP, CP/MISF と逐次最適化アルゴリズム DF/IHS，並列最適化アルゴリズム PDF/IHS の性能を評価した結果について述べた．性能評価の結果，STG Ver3 が各種アルゴリズムの並列度や正規化偏差に対する特徴をうまく捉えた評価が可能であることが確認できた．また，プロセッサ数を 2，4，8 台とした 8370 例において，FIFO，CP, CP/MISF の各ヒューリスティックアルゴリズムはそれぞれ 6.68 %，59.06 %，59.01 %の問題で最適解が得られた他，IBM p550Q 上で制限時間を 600 秒とした場合，1PE を用いた DF/IHS が 87.25 %，8PE を用いた PDF/IHS が 92.25 %の問題で最適解を得ることを確かめられた．

本稿で提案した STG Ver3 は早大笠原研究室 web ページ内の STG ページ
http://www.kasahara.cs.waseda.ac.jp/schedule/
にて最適解と併せて公開されており，多くの研究者に利用されることを期待する．

参 考 文 献

1) Coffman, E.G.: “Computer and Job-Shop Scheduling Theory”, John Willey & Sons (1976)
2) Kasahara, H. and Narita, S.: “Practical Multiprocessor Scheduling Algorithms for Efficient Parallel Processing”, IEEE Trans. Comput., Vol. C-33, No.11, pp.1023-1029 (1984).
3) Kasahara, H., H.H. and Narita, S.: “Parallel Processing of Near Fine Grain Tasks Using Static Scheduling on OSCAR”, Proc. IEEE ACM Supercomputing (1990).
4) Adam, T.L, Chandy, K.M and Dickson, J.R: “A Comparison of List Schedules for Parallel Processing Systems”, Comm. ACM, Vol.17, No.12, pp.685-690 (1974).
5) Ramamoorthy, C.V., Chandy, K.M. and Gonzalez, M.J.: “Optimal Scheduling Strategies in a Multiprocessor System”, IEEE Trans. Comput., Vol.C-21, No.2, pp.137-146 (1972).
6) 笠原 博徳，伊藤 敦，田中 久充，伊藤 敬介: “実行時間最小マルチプロセッサスケジューリング問題に対する並列最適化アルゴリズム”, 信学論, Vol.J74-D-I, No.11, pp.755-764 (1991).
7) 飛田 高雄, 笠原 博徳: “標準タスクグラフセットを用いたマルチプロセッサスケジューリングアルゴリズムの性能評価”, JSPP2000, pp.131-138 (2000).
8) 飛田 高雄, 笠原 博徳: “標準タスクグラフセットを用いた実行時間最小マルチプロセッサスケジューリングアルゴリズムの性能評価”, 情報処理学会論文誌, Vol.43, No.4(2002).
9) 松澤 能成, 坂井田 真也, 飛田 高雄, 笠原 博徳: “並列度を考慮した標準タスクグラフセットを用いた実行時間最小マルチプロセッサスケジューリングアルゴリズムの性能評価”, 情報処理学会研究報告, 2005-ARC-161,Vol.2005,pp.45-50 (2005).
10) Fernandez, E.B. and Bussel, B. : “Bounds on the Number of Processors and Time for Multiprocessor Optimal Schedule”, IEEE Trans. Comput., 22, No.8, pp.745-751 (1973).
11) Li, G.J. and Wah, B.W.: “Coping with anomalies in parallel branch-and-bound algorithms”, IEEE Trans. Comput., No.6, pp.568-573(1986).