2005－ARC－161　(9)
2005／1／21

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

松澤能成[†]　坂井田真也[†]
飛田高雄[††]　笠原博徳[†]

本論文では，実行時間最小化ノンプリエンプティブマルチプロセッサスケジューリングの公平な性能評価を可能とするために開発中のベンチマークタスクグラフを用いたヒューリスティックアルゴリズム，実用的逐次型最適化アルゴリズム及び並列最適化アルゴリズムの性能評価について述べる．本論文で用いる標準タスクグラフセット (STG) では，タスクグラフの並列度とスケジューリング対象プロセッサ台数の関係が，最適解求解率へ影響を与えることに注目し，タスク数規模は 50, 100, 300, 500, 700, 1000 ，タスクグラフの並列度 $para$ は $1.5 \leq para < 20.5$ の範囲で，3078 例のタスクグラフを生成した．この STG を用いて，2〜16 台のプロセッサに割り当てる際のヒューリスティックスアルゴリズム FIFO (First In First Out), RTRS (Ready Task Random Selection), CP (Critical Path), CP/MISF (CP / Most Immediate Successor First), 実用的逐次型最適化スケジューリングアルゴリズム DF/IHS (Depth First / Implicit Heuristic Search) 及び，その並列化アルゴリズム PDF/IHS (Parallelized DF/IHS) の性能評価を行った．この結果，全 12312 例において，FIFO で 15.14 %, RTRS で 14.63 %, CP で 65.80 %, CP/MISF で 65.85 %, DF/IHS で 87.79 %, PDF/IHS で 91.62 % の最適解求解率が得られた．また，探索上限時間を 6 時間とした場合，SUN 4PE WS Ultra80 上で，PDF/IHS は DF/IHS に比べタスク割り当て対象プロセッサ台数 2 の時平均 554.6 倍，4 の時平均 461.8 倍と非常に高い加速率を得ることができた．さらに，$para$ とプロセッサ台数が近い時，各アルゴリズムにおいて求解率が急激に低下し，プロセッサ台数 4 の時においては，CP などのヒューリスティックアルゴリズムでは "$para$ > プロセッサ台数" の時においても求解率が低下し，最適解求解率が約 60 % であるが，DF/IHS では 約 90 %，PDF/IHS では約 100 % という高い求解率が得られることが確認された．

Performance Evaluation of Minimum Execution Time Multiprocessor Scheduling Algorithms Using Standard Task Graph Set Which Takes into Account Parallelism of Task Graphs

Takanari Matsuzawa,[†] Shinya Sakaida,[†] Takao Tobita[††]
and Kasahara Hironori [†]

This paper evaluates performance of heuristic and optimization algorithms using benchmark task graphs named Standard TaskGraph Set (STG) for the minimum execution time nonpreemptive multiprocessor scheduling problem. In the standard task graph set used in this paper, in addition to the relationship between parallelism of task graphs and "*the number of processors*" which is the number of processors used in the scheduling problem, the scale of task graphs like 50, 100, 300, 500, 700, 1000 tasks, and parallelism "$para$" of $1.5 \leq para < 20.5$ affects optimal solution rate. This paper evaluates perfomance of heuristic algorithms, practical sequential optimization algorithm DF/IHS (Depth First / Implicit Heuristic Search) and practical parallel optimization algorithm (Parallelized DF/IHS) using this STG also for 2 to 16 processors. The evaluation shows for the total 12312 tested problems, FIFO gives us optimal solutions for 15.14 % of the problems, RTRS for 14.63 %, CP for 65.80 %, CP/MISF for 65.85 %, DF/IHS for 87.79 % and PDF/IHS for 91.62 %. Also, it was confirmed that the parallel algorithm PDF/IHS gave us 554.6 times speed up against the sequential algorithm DF/IHS for 2 processors scheduling problems and 461.8 times for 4 processors scheduling problems. When $para$ is close to *the number of processors*, each algorithm gives us low optimal solution rate, in addition to that, when *the number of processors* is 4 and "$para$ > *the number of processors*", heuristic algorithms like CP gives us low optimal solution rate (60 %) and however, DF/IHS and PDF/IHS give us high optimal solution rate such as 90 % and 100 % respectively.

† 早稲田大学理工学部 コンピュータ・ネットワーク工学科
Dept. of Computer Science, Waseda University

†† （株）ソニー・コンピュータエンタテインメント
Sony Computer Entertainment Inc.

1. はじめに

マルチプロセッサシステム上で，効率よく並列処理を行うには，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)~3)]．

この問題は，一般的に極めて難しい最適化問題であり，特にタスク間の先行制約が任意形状で，タスク処理時間やプロセッサ台数が任意である問題は，共有メモリ型マルチプロセッサを仮定し，共有メモリアクセスが各タスクの処理時間の中に含まれるとした場合，すなわちプロセッサ間のデータ転送オーバーヘッド (以下，データ転送時間と記述) はタスク処理時間の中に含まれているとした場合でも，強 NP 困難となってしまう難しい (intractable) 問題である．すなわちこの問題は，P $\neq$ NP ならば，擬多項式時間最適化アルゴリズム，及び両完全多項式時間近似スキームの構築も不可能であることを意味しており[4)]，組み合わせ最良化問題の中でも最も難しい問題のグループに入っている．

これらの問題を解くために，種々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おり，HLFET (Highest Levels First with Estimated Times)[5)]，CP (Critical Path)[1)]，CP/MISF (Critical Path / Most Immediate Successors First)[2)] などが提案されている．また，最適解を求める実用的な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を 2，タスク数を約 40 として最適解を得ている[6)] ほか，タスク数数百の問題の多くを数分で求解することができる実用的逐次型最適化アルゴリズム DF/IHS (Depth First / Implicit Heuristic Search)[2)] と，その並列化アルゴリズム PDF/IHS (Parallelized DF / IHS)[7)] を提案している．

データ転送を考慮したスケジューリングアルゴリズムとしては，CP/DT/MISF (Critical Path / Data Transfer / Most Immediate Successors First)[3)]，DSC (Dominant Sequence Clustering)[8)], DCP (Dynamic Critical Path)[9)] や TDS (Task Duplication based Scheduling)[10)] などが提案されている．

これらのスケジューリングアルゴリズムの性能評価も行われており[11)]，筆者等は公正なアルゴリズム評価のためのベン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[12)]．これまでに，タスクの処理時間の決定方法，タスクグラフ形状などによる各スケジューリングアルゴリズムの性能評価が行われてきたが，並列度 $para$ の値がアルゴリズムの最適解求解率にどのような影響を与えるのかは，ほとんど議論されることがなかった．本論文では，並列度 $para$ が最適解求解率に与える影響[12)] に注目し，タスク数 50, 100, 300, 500, 700, 1000 のランダムタスクグラフ 3078 例を用いての評価例として，FIFO, RTRS (Ready Task Random Select), CP, CP/MISF, DF/IHS, PDF/IHS の性能評価を行い，並列度 $para$ が最適解求解率にどのように影響を与えるのか検証する．

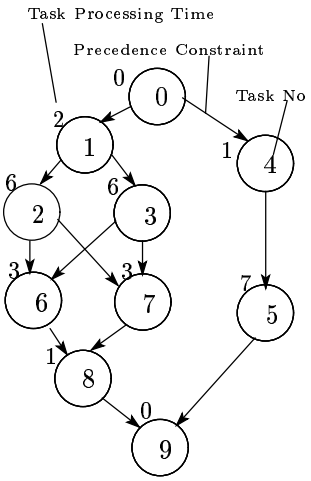

図 1 タスクグラフ例

2. 対象スケジューリング問題の定義

本論文で扱うマルチプロセッサスケジューリング問題は，能力の等しい m 台のプロセッサで，任意の n 個のタスク (処理時間は任意，先行制約は任意形状，各タスク間のデータ転送時間を無視できるとする) からなるタスク集合 $T = \{T_1, T_2, \ldots, T_n\}$ をノンプリエンプティブ (処理割り込み無し) に並列処理する際に，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．この時，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 は有向エッジの集合) 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，出口ノードから到達できるものとする．もし，問題となるタスクグラフが複数の出入口ノードを有する場合，ダミーノードを付加することにより上記の形に変換できるため，一般性を失うことはない．図 1 に 8 個のタスク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:unit time，以下 [u.t.]) 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．

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，本論文で性能評価を行うヒューリスティックアルゴリズム FIFO, RTRS, CP, CP/MISF, DF/IHS 及び 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

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．しかしながら，タスクグラフ作成時に番号が若いタスクが出口ノードからのパス長が長いという関係があると，FIFO は CP 法に近

い値を出す可能性があるため，空きプロセッサにレディキューのタスクを割り当てる際に，レディキュー内のタスクをランダムに選び出すようにしたアルゴリズムが，RTRS (Ready Task Random Select) である．CP 法は，CP 長が長いタスクから順に割り当て優先度リストを作成し，レディキュー内の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スケジューリングアルゴリズムである．また，CP 法では同一 CP 長を持つタスクが複数ある場合にタスクの優先順位を一意に決定できないため，優先度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ものが CP/MISF である[2]．

3.2 逐次最適化アルゴリズム DF/IHS

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 の割り当て優先度の高い順，すなわち大きい 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ことにより，探索木の左に CP/MISF 的に良い解を集め，探索部では DF/FIFO (Depth First / First In First Out) 方式で探索木の左側から探索を行うことにより，ヒューリスティック的に良い解から見つけることができる．また，初期解が CP/MISF 解となり，非常に良い暫定解を得られる可能性が高く，探索上界値を小さな値に設定できるため，効果的に限定操作（枝切り）を行うことができ，探索時間を大幅に削減できる．

この限定操作は，上界値とその枝を伸ばして探索した場合に得られる可能性のある解の下界値を比較し，上界値より良い解が存在する可能性がある場合は，その枝を伸ばして探索し，分枝により分解された部分問題に対する下界値として 3 つのもの使用する[2]．Fernández によって拡張された Hu の下界[13] については，下界値の精度は上がるが計算量が多いので，他の 2 つの下界値により限定できなかった場合のみ計算する[2]．

また DF/IHS は，SP (Selection Pointer) 値と呼ばれるポインタを使用することにより，領域複雑度を $O(n^2+mn)$ に抑えることができる[2]．

3.3 並列最適化アルゴリズム PDF/IHS

PDF/IHS[7] 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 (リーダ PE) は DF/IHS と同様に探索木の左から右に優先探索を行い，初期解が下解と一致しないことを他の PE_j $(j=2,\ldots,k)$(スレーブ PE) に通知後，スレーブ PE はリーダ PE の探索経路上のノードを根ノードとする部分探索木を右から左へ探索する．マスター PE，スレーブ PE による左右からの階層的深さ優先挟み撃ち探索により，探索木の右側にある最適解を効率良く見つけることができ，スレーブ PE は割り当て領域の探索を終了するかリーダ PE と出会うまでは独立に探索できるので，PE 間の負荷均等化が図れる．また，リーダ PE が DF/IHS と同様の探索を行うため，減速異常は生じない[14]．

4. 評価用標準タスクグラフセット

本論文では，標準タスクグラフセット生成ツールを用いて，以下のように生成したランダムタスクグラフを用いる．また，タスクグラフの性質を表す指標として，タスクグラフの並列度 $para$ を以下の式で定義する．

$$para = \frac{\text{タスク処理時間の総和}}{\text{タスクグラフの } CP \text{ 長}}$$

問題規模の変化が性能に与える影響を知るため，タスク数は 50, 100, 300, 500, 700, 1000 の 6 種類を用意し，タスクグラフの形状は（各タスク間の接続方法）は，$samepred, layrpred, linkpred$ の 3 種類を用意した．

第一の形状 $samepred$ は，各タスクの平均直接先行タスク数 pre を指定して，以下の規則により各タスク間にエッジを生成する．

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} a(i,j) = pre \cdot n$$

ここで，$a(i,j)$ (ただし $0 \le i \le n+1, 0 \le j \le n+1$, n はタスク数) はタスク接続行列 A の各要素を表し，$a(i,j)=1$ のときにタスク T_i は タスク T_j に先行する．すなわち T_i は T_j の開始前に実行が終了している必要があることを示し，$a(i,j)=0$ ならば，T_i から T_j へのエッジは存在しないことを示している．

第二の形状 $layrpred$ は，まずタスクグラフ中に作成する層（レイヤ）数を決定し，次に各層に独立なタスクをランダムに配置した後，$samepred$ のように各タスクの平均直接先行タスク数 pre を指定して，同様の規則で各タスク間にエッジを生成する．

第三の形状 $linkpred$ は，タスクグラフの中からタスクをランダムに 2 つ選びだし，番号の小さい方から大きい方へエッジを生成する．

各タスクの処理時間（processing time）決定方法は一様乱数，指数乱数，正規乱数の 3 種類を用い，平均値は 5, 10, 20 の 3 種類を使用した．正規乱数での標準偏差 SD は，以下の範囲を満たすように一様乱数で決定した．

$$1 \le SD \le \lceil \frac{\text{タスク処理時間の平均値}+1}{2} \rceil$$

これらの条件を使用して各タスク数ごとに莫大な数のタスクグラフを作成し，$para$ の値が $1.5 \le para < 2.5, \cdots, 19.5 \le para < 20.5$ の各範囲 (計 19 種類) において，形状別，処理時間別，処理時間決定方法別にタスクグラフが一つずつ分布するようにランダムに取り出す．以上から，タスク数ごとに，19($para$ の種類) * 3(形状) * 3(平均タスク処理時間) * 3(処理時間決定方法数) = 513 通り のタスクグラフが生成される．本論文では，6 種類のタスク数，計 3078 例のタスクグラフを使用し，3 章で述べたアルゴリズムに適用して性能を評価する．

5. 並列度 $para$ を考慮した標準タスクグラフセットを用いた性能評価

本章では，4 章で述べた並列度 $para$ を考慮した標準タスクグラフセットを用いて FIFO, RTRS, CP, CP/MISF, DF/IHS, PDF/IHS の性能評価を行った結果について述べる．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 の並列探索に使用される主記憶共有型マルチプロセッサシステム SUN Ultra80 (以下 Ultra80) のプロセッサを “PE” と表記する．また Ultra80 は，450MHz UltraSPARC IIs (一次キャッシュはデータ/命令ともに 16KB, 二次キャッシュは 1MB) を 4 台，メインメモリ 1GB を搭載している．

5.1 性能評価条件

本評価では，生成した 3078 例のランダムタスクグラフを 2, 4, 8, 16 プロセッサにスケジュールする問題合計 12312 例に対して，Ultra80 の 1PE 上で FIFO, RTRS, CP, CP/MISF を適用し，求解を行った．同様に，ランダムタスクグラフ 3078 例を，2, 4, 8, 16 プロセッサにスケジュールする問題を最適化アルゴリズム DF/IHS (PE を 1 台使用)，PDF/IHS (PE を 4 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．

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 のように，FIFO は全問題の 14.63 %，RTRS は 15.14 %，CP は 65.80 %，CP/MISF は 65.86 % に最適解を得られることが確かめられた．また，探索時間 600 秒以内に DF/IHS では 87.79 %，4PE を用いた PDF/IHS では 91.62 % に最適解が得られた．この最適解求解数では，下界または DF/IHS, PDF/IHS により得られた最適解と一致する解が得られた場合を “最適解が求まった” としており，探索上限時間内に最適解であると確認できていなくても得られた解が最適解である可能性があるため，各アルゴリズムの真の最適解求解率は 表 1 より高い可能性がある．表 1 から分かるように，タスク数の増加と共に各アルゴリズムの求解数が減少していることが分かる．このことは，本スケジューリング問題の求解計算複雑度が強 NP 困難であり，$O(C^n)$ (C は定数，n はタスク数) になるためである．

また，割り当てプロセッサ台数別の最適解求解率を表 2 に示す．表 2 のように，DF/IHS においては，プロセッサ台数が 2 から 8 においてプロセッサ台数の増加に伴い求解率が 93.76 %, 86.32 %, 83.14 % と減少し，PDF/IHS においても，プロセッサ台数が 2 から 8 においてプロセッサ台数の増加に伴い求解率が 95.58 %, 91.88 %, 88.82 % と減少している．また，プロセッサ台数が 8 から 16 においては，プロセッサ台数の増加に伴い，DF/IHS では求解率が 83.14 % から 87.95 % に増加し，PDF/IHS でも 88.82 % から 90.19 % に増加していることが分かる．それ以外の 4 種類のヒューリスティックアルゴリズムに関しては，プロセッサ台数が 2 から 4 においてプロセッサ台数の増加に伴い，FIFO が 8.35 % から 1.17 %，RTRS が 10.49 % から 0.94 %，CP が 68.42 % から 52.96 % , CP/MISF が 68.13 % から 53.25 % と求解率が減少し，プロセッサ台数が 4 から 16 においてはプロセッサ台数の増加と共に FIFO が 1.17 %, 12.44 %, 36.55 %，RTRS が 0.94 %, 12.74 %, 36.39 %，CP が 52.96 %, 58.71 %, 83.11 %，CP/MISF が 53.25 %, 59.45 %, 82.81 % と求解率が増加していることが分かる．

次に，各アルゴリズムの並列度 $para$ と最適解求解率の関係をプロセッサ台数別に図 2〜5 に示す．なお，以下の図 2〜5 においては $1.5 \leq para < 2.5$ のものを $para = 2$，$2.5 \leq para < 3.5$ のものを $para = 3, \cdots$ と表すものとする．図 2〜5 から，FIFO, RTRS においては，プロセッサ台数が 2 のときには $para$ の増加と共に求解率に改善が見られるが，プロセッサ台数が 4 以上になると，$para$ が増加しても，ほとんど最適解が得られないことが分かる．CP, CP/MISF においては，図 3 のプロセッサ台数 4 の場合の $para$ 4 の時のように，プロセッサ台数と $para$ の値が近い時には最適解求解率は大幅に低下するが，図 3,4 のそれぞれ $para$ 4, $para$ 8 以上の時のように $para$ の値がプロセッサ台数よりも大きくなるにつれ，最適解求解率の改善が見られる．DF/IHS, PDF/IHS においては，プロセッサ台数と $para$ の値が近い時には求解率が大幅に低下するが，プロセッサ台数が 2, 4 の時に至っては，DF/IHS で 90 % 以上，PDF/IHS でほぼ 100% の最適解を得るという高い最適解求解率が得られている．このように図 2〜5 のどの場合においても，上記 4 つのヒューリスティックアルゴリズムに比べてこれらの 2 つのアルゴリズムの性能の高さを確認することができる．

全てのアルゴリズムにおいて，“$para <$ プロセッサ台数” の場合に最適解が得られやすくなっている理由としては，ある時点におけるレディタスク数よりも実行可能プロセッサ台数の方が多いため，クリティカルパス上に存在するタスクを，割り当てプロセッサ待ちさせることなく実行できる可能性があることが考えられる．また，$para$ の増加に伴い，最適解求解率が増加している理由としては，ある時点における実行可能プロセッサ台数よりもレディタスク数が多いため，どのタスクからプロセッサに割り当てても，最終的なスケジュール長にあまり影響を及ぼさなくなるということが考えられる．

次に，PDF/IHS により最適解が得られた問題に対し，DF/IHS に比べ，どの程度のスピードアップが図れているのか検討する．ここで，加速率 α は以下で表される．

$$\alpha = \frac{\text{DF/IHS を用いた場合の探索時間}}{\text{PDF/IHS を用いた場合の探索時間}}$$

表 3 に PDF/IHS により最適解が得られた問題に対する DF/IHS, PDF/IHS それぞれの探索時間を示す．DF/IHS において最適解が得られなかった場合は，探索時間を 600 秒として換算している．ここで，求解に用いたマルチプロセッサワークステーション Ultra80 の PE 数が 4 台であるので，$NoSpeedup$ は $\alpha \leq 1$ のもの，$LinearSpeedup$ は $1 < \alpha \leq 4$ のもの，$SuperLinearSpeedup$ は $\alpha > 4$ のものとしている．多くの問題が，$NoSpeedup$ であるが，これは DF/IHS, PDF/IHS の初期解 (CP/MISF 解) が探索上界値

表 1　タスク数別最適解求解数

Task 数	問題数	RTRS	FIFO	CP	CP/MISF	DF/IHS	PDF/IHS
50	2054	341	329	1410	1412	1844	1910
100	2054	329	323	1444	1469	1885	1996
300	2054	328	311	1371	1361	1814	1914
500	2054	304	299	1284	1306	1762	1848
700	2054	284	270	1338	1306	1773	1832
1000	2054	278	269	1254	1255	1731	1780
total	12312	1864	1801	8101	8109	10809	11280
%	100.00	15.14	14.63	65.80	65.86	87.79	91.62

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	RTRS	FIFO	CP	CP/MISF	DF/IHS	PDF/IHS
2	3078	10.49	8.35	68.42	68.13	93.76	95.58
4	3078	0.94	1.17	52.96	53.25	86.32	91.88
8	3078	12.74	12.44	58.71	59.45	83.14	88.82
16	3078	36.39	36.55	83.11	82.81	87.95	90.19
total	12312	1864	1801	8101	8109	10809	11280
%	100.00	15.14	14.63	65.80	65.86	87.79	91.62

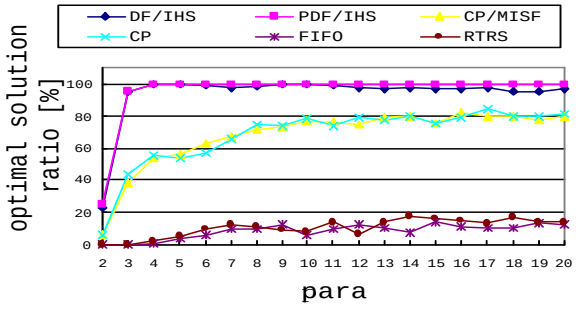

図 2　アルゴリズム別最適解求解率（プロセッサ台数 2）

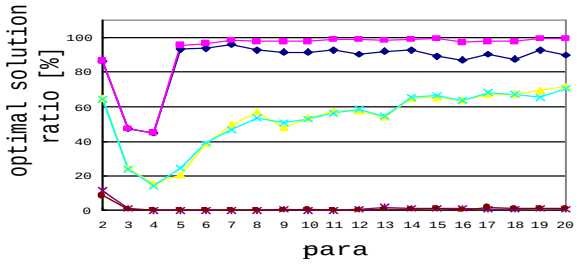

図 3　アルゴリズム別最適解求解率（プロセッサ台数 4）

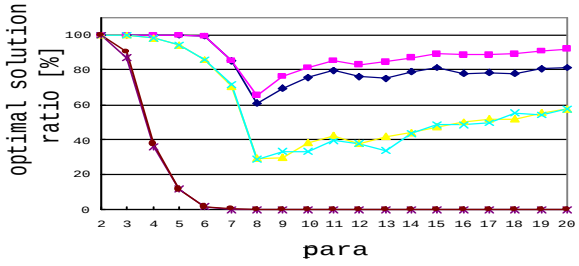

図 4　アルゴリズム別最適解求解率（プロセッサ台数 8）

と一致して最適解となった問題や，探索木の左側に最適解が存在し，1 ～ 2 秒程度の短時間で最適解が求まってしまう場合がほとんどであり，実用上スピードアップの有無はユーザにとってほとんど問題にならない．また，*SuperLinearSpeedup* の問題に対しては，探索木の右側に最適解が存在し，探索木を左から右へ探索する DF/IHS では長時間を要するが，右から左へ探索する PE を持つ PDF/IHS ではより短時間で最適解を見つけることができる場合である．この場合，DF/IHS では最適解を求めるのに平均 400 秒以上要する問題に対しても，PDF/IHS を用いれば，8 秒未満で最適解を求めることを可能にしており，実用性の高さを確認できる．表 3 全体 11280 例の平均としては，6.90 倍の加速率を得ることに成功しており，PDF/IHS が *SuperLinearSpeedup* を可能にしていることが確認できる．

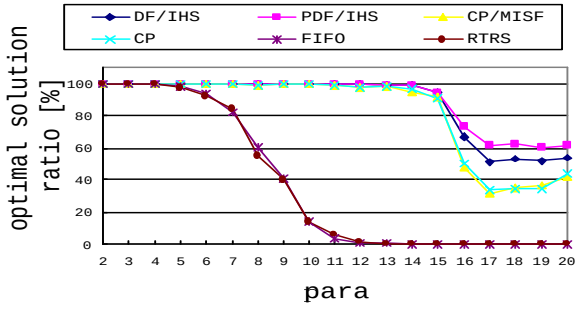

図 5　アルゴリズム別最適解求解率（プロセッサ台数 16）

表 4 に PDF/IHS により最適解が得られた問題に対するプロセッサ台数別の加速率を示す．DF/IHS において最適解が得られなかった場合は，図 3 と同様に探索時間を 600 秒として換算している．表より，プロセッサ台数が 2 台の時は 2942 例の平均として 22.22 倍，4 台の時は 2828 例の平均として 17.68 倍，8 台の時は 2734 例の平均として 4.63 倍と，16 台の時以外は全て *SuperLinearSpeedup* が図れていることが分かる．

表 5 に PDF/IHS により最適解が求まったが，DF/IHS では求まらなかったものに対して，制限時間を 6 時間として，最適解を求めさせた時の加速率を示す．現在得られているデータが プロセッサ台数が 2, 4 の時のみであるが，各プロセッサ台数において，554.6 倍，461.8 倍という加速率が得られており，DF/IHS で最適解を求めるのに長時間を要する問題に対しても，PDF/IHS が短時間で最適解を得ることを可能にしていることが確認できる．

表 3 探索時間と加速率

分類	問題数		DF/IHS	PDF/IHS
NoSpeedup	10054	合計時間 (秒)	18486.15	21765.64
		平均時間 (秒)	1.84	2.16
		加速率	1.00	0.85
Linear Speedup	542	合計時間 (秒)	39253.90	21636.49
		平均時間 (秒)	72.42	39.92
		加速率	1.00	1.81
Super Linear Speedup	684	合計時間 (秒)	277631.70	5228.33
		平均時間 (秒)	405.89	7.64
		加速率	1.00	1.00
total	11280	合計時間 (秒)	335371.80	48630.45
		平均時間 (秒)	29.73	4.31
		加速率	1.00	6.90

表 4 探索時間と加速率 (プロセッサ台数別, 600 秒)

プロセッサ台数	問題数		DF/IHS	PDF/IHS
2	2942	合計時間 (秒)	40694.55	1831.59
		平均時間 (秒)	13.83	0.62
		加速率	1.00	22.22
4	2828	合計時間 (秒)	135444.80	7661.80
		平均時間 (秒)	47.89	2.71
		加速率	1.00	17.68
8	2734	合計時間 (秒)	111792.80	24127.70
		平均時間 (秒)	40.89	8.83
		加速率	1.00	4.63
16	2776	合計時間 (秒)	47439.62	15009.36
		平均時間 (秒)	17.09	5.41
		加速率	1.00	3.16
total	11280	合計時間 (秒)	335371.80	48630.45
		平均時間 (秒)	29.73	4.31
		加速率	1.00	6.90

表 5 探索時間と加速率 (プロセッサ台数別, 6 時間)

プロセッサ台数	問題数		DF/IHS	PDF/IHS
2	2942	合計時間 (秒)	1015805	1831.59
		平均時間 (秒)	345.3	0.62
		加速率	1.00	554.6
4	2828	合計時間 (秒)	3538260	7661.80
		平均時間 (秒)	1251.2	2.71
		加速率	1.00	461.8

6. ま と め

本論文では，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において，“並列度を考慮した標準タスクグラフセット” を用いて評価を行う手法を提案すると共に，それを用いてヒューリスティックアルゴリズム FIFO, RTRS, CP, CP/MISF と逐次型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，タスク数 50〜1000, プロセッサ数 2〜16 の 12312 例において，RTRS, FIFO, CP, CP/MISF の各ヒューリスティックアルゴリズムでは，それぞれ 15.14%, 14.63%, 65.80%, 65.86% の問題に最適解が得られたほか，SUN 4PE WS Ultra80 上で 600 秒以内に 1PE を用いた DF/IHS が 87.79%, 4PE を用いた場合の PDF/IHS が 91.62% の問題に最適解を与えることが確かめられた．各ヒューリスティックアルゴリズムにおいては，“*para* < プロセッサ台数” の場合には，求解率が悪くなってしまうが，DF/IHS, PDF/IHS においてはそのような場合においても高い求解率が得られた．また，PDF/IHS で求解できた問題に対し，探索上限時間を 600 秒とした場合，PDF/IHS は DF/IHS に対して全 11280 例の平均として 6.90 倍の 加速率 (*SuperLinearSpeedup*) が得られ，探索上限時間を 6 時間とした場合，割り当て対象プロセッサ台数が 2, 4 の時はそれぞれ 554.6 倍，461.8 倍の加速率が得られ，本論文のような強 NP 困難な最適化問題に対しても，大規模問題を含め，実用的な時間内での求解が可能であることが確かめられた．

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 (1988).
3) Kasahara, H., H. H. and Narita, S.: Parallel Processing of Near Fine Grain Tasks Using Static Scheduling on OSCAR, *Proc. IEEE ACM Supercomputing '90* (1990).
4) Garey, M. R. and Johnson, D. S.: A Guide to the Theory of NP-Completeness, *CA: Free-man* (1979).
5) Adam, T.L., C. K.M. and Dickson, J.R.: A Comparison of List Schedules for Parallel Processing Systems, *Communications of the ACM*, Vol. 17, No. 6, pp. 685–690 (1974).
6) Ramamoorthy, C. V., C. 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).
7) 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, No. 11, pp. 755–764 (1991).
8) Yang, T. and Gerasoulis, A.: DSC:Scheduling Parallel Tasks on an Unbounded Number of Processors, *IEEE Transaction on parallel and Distributed systems, Vol.9*, Vol. 5, No. 9 (1994).
9) Kwok, Y. and Ahmad, I.: Dynamic Critical-Path Scheduling, *IEEE Trans. on Parallel and Distributed Systems*, Vol. 7, No. 5, pp. 506–521 (1996).
10) Darbha, S. and Agrawal, D. P.: Optimal Scheduling Algorithm for Distributed-Memory Machines, *IEEE Trans. Parallel and Distributed Systems*, Vol. 9, No. 1, pp. 87–95 (1998).
11) Kwok, Y. and Ahmad, I.: Benchmarking and Comparison of The Graph Scheduling Algorithms, *Journal of Parallel and Distributed Computing* (1999).
12) 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 (2001).
13) B., F. E. and Bussel, B.: Bounds on the number of processors and time for multiprocessor optimal schedules, *IEEE Trans. Comput.*, No. 8, pp. 745–751 (1973).
14) 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).