成蹊大学理工学研究報告
J. Fac. Sci.Tech., Seikei Univ.
Vol.49 No.1 (2012)pp.1-12

タスクスケジューリングにおける過剰な通信遅延の排除と通信可能域の走査に関する経験則の導入

宇都宮　雅彦[*1]，甲斐　宗徳[*2]

Applying heuristics using scanning of communication term and elimination of ineffective commu-nication overhead to Task Scheduling

Masahiko Utsunomiya[*1], Munenori KAI[*2]

ABSTRACT : Task scheduling is one of core technologies to improve the efficiency of parallel processing. A schedule is a solution of task scheduling problem, and make to cooperate the performance of parallel machines. The shorter length of schedule (makespan) is reduced, the more efficient parallel machines run. But, task scheduling problem is combinatorial optimization problem that has strongly NP hard computational complexity. Accordingly, it is necessary that to design algorithm taking account of more unforeseeable disposition in order to reduce makespan. Concerning to solve the problem, the disposition is considered with communication overhead between machines. In this paper, the authors propose new heuristic algorithms to optimize communication overhead. These are applied the focus to critical path of dependences and earliest allocation pattern of communications. The authors describe characteristics and processes of these algorithms, and our experiments show the effectiveness of them

Keywords : Task scheduling, Combinatorial optimization problem, Strongly NP hard, Communication overhead

(Received　March 31, 2012)

1．はじめに

タスクスケジューリングは，並列処理環境におけるタスク並列の中核となる最適化技術の一つである。

並列化技術の普及に伴い，実行時間最小化を目的としたタスクスケジューリングアルゴリズムの需要が高まっている。タスク並列は，大規模で計算量の多い処理を複数の計算装置に分割することで並列化を行うが，そのスケジュールを求める問題は強 NP 困難な計算複雑度を持つことが知られており，任意の問題に対し多項式時間で最小の優越解集合を得ることは不可能に近い[1] 。そこで，性能限界を持つ理論的手法から実装ベースの手法まで，多くの近似アルゴリズムが提案されている[6] [7] 。

タスクスケジューリング問題において最も複雑な条件の 1 つに，タスク間の実行順序に制約を与える依存関係が存在する。タスクスケジューリング問題の強 NP 困難な計算複雑度を持つ性質から，問題に最も強い複雑性を与える依存関係を最適化することが，組み合わせの結果であるスケジュールの長さと算出時間に大きく影響を与えると考えられる。しかし，依存関係による通信の組み合わせは高い計算複雑度の部分問題となることが多く，アルゴリズムが通信を考慮する際には，その計算量を充分に考慮しなければならない。

本稿では，依存関係に着目した仮説とその検証実験から，計算量を抑えて依存関係の最適化を図る 2 つのヒューリスティックアルゴリズムを提案する。これらのアルゴリズムを用いて，問題規模の増加に伴う組み合わせ爆発を抑えつつ，より最適化されたスケジュールの導出が可能となることを示す。

[*1] ：理工学研究科理工学専攻博士前期学生
[*2] ：理工学研究科理工学専攻教授 (kai@st.seikei.ac.jp)

2．タスクスケジューリング

2.1.　タスクスケジューリング問題の定義

筆者は，タスクスケジューリング問題を“性能の等しいM個の処理装置（Processing Element，以下PE）を用いて，任意の順序依存と任意の処理時間を持つN個のタスクの全てを割り込み無しで処理するとき，各PEが一度に高々一つまでの処理・依存情報の送信・受信を可能とする条件下で，依存間の通信の組み合わせが生じる遅延を考慮した実行時間の最小化を目的関数とした，組み合わせ最適化問題”と定義し，短時間に良質な近似解を導出する近似アルゴリズムの研究を行っている。

本章の以降では，上記の各定義に関して解説を行う。

2.2.　タスク

タスクは，スケジューリング問題における分割不可能な処理単位である。任意のタスクをスケジュールの中に割り当てるためには，そのタスクごとに定められたコストに基づいて係数倍した単位時間を払い，1つのPEを占有する必要がある。

2.3.　依存関係

任意のタスク間の依存関係として定められた通信は，先行依存側のタスクが終了してから後続依存側のタスクが実行を開始するまでの任意の時間に割り当てる必要がある。タスクは自身の先行関係に相当する全タスクの処理が完了するまで処理が開始できない。依存関係を解決するときには，先行タスクを処理したPEの送信ポートと後続タスクを処理するPEの受信ポートが，同時かつ通信のコスト以上の時間アイドル状態でなければならない。ただし，依存関係を持つタスクのペアが同一のPEで処理される場合，先行タスクの処理結果に相当する情報は既に対象のPEが保持しているため，依存関係に定められたコストを支払う必要がないものとする。

2.4.　処理装置

処理装置PEはタスクを消費する主体である。各PEの性能は等しいものとする。本稿では依存解決の逐次性（2.7節参照）を考慮しているため，ある時刻における，あるPEの稼動状態としては，1つのタスクの処理，1つの依存情報の受信及び，1つの依存情報の受信のみが可能である。

2.5.　タスクグラフ

本稿で取り扱うタスクスケジューリング問題は始点と終点のノードを各々1つとするDirected Acyclic Graphで表現できる。この条件は元の問題の始点と終点にコストが0のダミーノードを付与することで達成でき，一般性を失わない。本研究の対象とするタスクグラフの例をFigure 1に示す。ノード内の数字がタスク番号，ノードの左側の数字がタスクの処理時間，エッジ横の<x>が依存間で要する通信コストxを示している。なお，グラフ中のノード，エッジの各コストは非負整数とする。

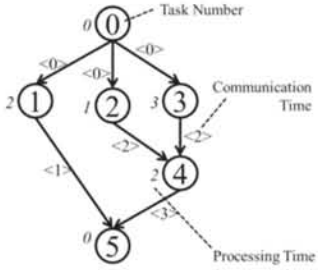

Figure 1 タスク数6のタスクグラフの例

2.6.　スケジュール

タスクスケジューリング問題の解をスケジュールと呼ぶ。また，あるスケジュールにおいて最も実行の遅いタスクの完了時刻をそのスケジュールのmakespanと呼ぶ。実行時間の最小化が目的のタスクスケジューリング問題では，このmakespanの値が最も小さくなるスケジュールが厳密解に相当する。

2.7.　依存解決の逐次性

通信遅延を考慮するモデルは複数存在するが，最も汎化されたモデルは，通信の重複に制限のないFully Coupled Modelである。しかし，情報の転送のための通信ポートが無制限に用意されていることは現実的ではないため，本研究ではより実用的なモデルとして，各PEが情報の送信ポートと受信ポートを1つずつ用意されたLoosely Coupled Modelを用いる。Loosely Coupled Modelでは，通信順序の組み合わせが生じ，最適化のためには複数の組み合わせを考慮する必要がある。したがって，問題の計算複雑度は強NP困難から更に複雑化する。

3．スケジュールの改善に向けた戦略　～通信の最適化～

本章より後で示す具体的な最適化アルゴリズムの構築に向けて，本稿が採った依存関係の最適化の方針を示す。

3.1.　既存手法の分類

タスクスケジューリング問題は組み合わせ最適化問題

であるため，その計算量と解の厳密性にはトレードオフの関係が存在する。問題の計算複雑度は強 NP 困難に分類されていることから，このトレードオフの関係は通信の組み合わせを考慮した各アルゴリズムの計算量に応じた指数関数によって表すことができる。各アルゴリズム次第でそのレートは変動するが，計算を行うほど考慮する組み合わせが増えるため makespan は短くなる傾向にある。一方で，計算量は各アルゴリズムの考慮する組み合わせによって指数的に増加してしまう。この関係から，各スケジューリングアルゴリズムは計算量と厳密性のどちらに重きを置いているかで分類が可能である。

計算量を重視し，タスクや通信の割り当て時に考慮する組み合わせを可能な限り減らしたアルゴリズムとしては，CP（Critical Path）[8] や CP/DT/MISF（Critical Path / Data Transfer / Most Immediate Successors First）[2] といったアルゴリズムが過去に提案されている。これらはリストスケジューリング[3] に基づく手法である。

makespan の質（厳密性）を重視したアルゴリズムとしては，ECT（Earliest Completion Time）に基づく HLFET（Highest Levels First with Estimated Times）[5] や ETF（Earliest Task First）[4] などの手法が提案されている。ECT では最も早期に完了できるタスクと PE の組み合わせを考慮するため，スケジューリング中に元の問題規模に応じた探索問題が生じる。

3.2. 方針の策定

前章で述べたタスクスケジューリング問題の定義及び前節で述べたトレードオフの関係から，通信を考慮したスケジューリングの性能向上のための方針として，通信の最適化を挙げる。

解の厳密性に重きを置いたアルゴリズムでは，複雑な計算を伴う探索が発生していることから，これらのアルゴリズムでは計算量が問題規模に応じて指数的に増加してしまう。通信を考慮したスケジューリングにとっての複雑さは，最適なスケジュールに要する通信を予測しきれない点にあることから，通信を最適化する手法をスケジューリングの流れの一部に加えることで，解の厳密性の向上を試みるという方針を採った。

スケジュール上の通信を最適な状態とするためには，組み合わせの選択，ないし割り当ての過程で通信の総量を操作することが必要となる。ただし，通信の発生を操作する機構は通信の発生量のバランスを考慮する必要がある。

以上より，筆者は通信の最適化を「よりスケジュールが短くなるように通信の発生を操作すること」と捉え，通信を最適化するための経験則を前処理とスケジュールへの割り当てに加えることで厳密性の向上を行うアルゴリズムの構築を行った。

4．過剰な通信遅延の排除を行うアルゴリズムの設計と開発

本章では，前処理において通信の最適化のために有効な手段を実験に基づいて考察し，特定の通信を組み合わせから排除するための「仮割り当て」機構と，依存のクリティカルパスに着目した新たなスケジューリングアルゴリズム CPD を提案する。

4.1. 通信に関する仮説の提案

通信の組み合わせの探索は強 NP 困難な計算複雑度を持つ部分問題となるため，可能な全ての組み合わせを探索した場合，スケジューリングに要する時間は問題規模に指数的に比例してしまう。この部分問題を解くことで得られる組み合わせに近い解をより少ない計算量で得られれば，通信を考慮したスケジューリングに対し，従来よりも解の相対性能を著しく低下させずに計算量を抑えた手法を与えることが可能となる。

探索によって求めたい組み合わせは，探索の候補となる実行可能タスクの集合と PE の組み合わせで生じる通信コストの総量がより少ないものであるから，相対的にコストの大きい通信は探索の結果得られる組み合わせに含まれにくいと考えられる。そこで，タスクグラフ中の相対的にコストの大きい依存関係に対し「スケジューリングにおいて組み合わせを冗長に増やすものの，より良質な解であるほど，通信としてコストを支払う場合が少ない」とする仮説を立て，検証実験を行った。

4.2. 検証実験

前節の仮説に基づき，ECT による探索を行う HLFET に対して，タスクグラフ中の相対的にコストの大きい通信がスケジュールの生成に与える影響を調査した。スケジューリング中の部分問題の探索時に探索候補から除くタスクと PE の組み合わせを，通信のコストの降順に 0～10%まで，1%ずつ変化させて計測を行った結果を Figure 2 に示す。グラフの値にはタスク数 300，CCR（Communication to Computation Ratio, グラフ上の通信遅延の総コストをタスクの総処理時間で除算した比）を 1.0 とした 180 種の異なるタスクグラフに対して計測した平均値を用いた。

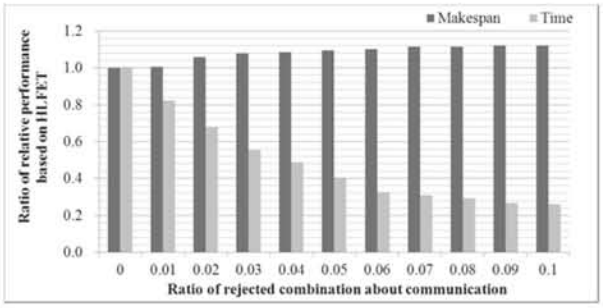

Figure 2 通信の排除率に伴う性能調査

実験結果から，考慮する組み合わせの減少に伴うmakespanの増加は緩やかであるのに対し，計算量は急激に減少している傾向が確認された。特に通信のコストが上位10%を占める組み合わせを考慮しない場合では，全ての組み合わせを考慮する場合と比較して，1.11倍程度のmakespanの増加に対し4.10倍以上の大幅な実行時間の短縮が得られている。

4.3. 仮割り当て

前項の実験結果から，アルゴリズムに対して特定の通信の組み合わせを強制することで，計算量と相対性能が効率的なレートでトレードできると考えられる。そこで，アルゴリズムによるタスクやPEの選択に依存せずに通信の組み合わせを省略するための前処理として，「仮割り当て」操作を提案する。

まず，スケジューリングを行う前の段階で，通信コストとその評価指標に基づき，通信を省略すると判断したタスクの先行依存に相当するタスクの番号を該当タスクの付加情報として与えておく。そして，スケジューリング時に該当タスクが選択されたとき，アルゴリズムによるPEの選択を行わずに，事前に与えられた付加情報に基づき先行タスクの割り当てられたPEに該当タスクを割り当てる。従って，割り当て前のタスクは割り当て先のPEが一意に決定されるがスケジューリングは行われていない状態となる。以降ではこの手法を仮割り当てと定義する。

4.4. CPDアルゴリズム

仮割り当てによって排除する通信の選択を，任意の問題に対して効果的に適用するために，依存関係のクリティカルパスを応用したヒューリスティックアルゴリズムCPD（Critical Path of Dependences）を提案する。

依存関係のクリティカルパスとは，元の問題のタスクグラフから通信コストのみを抽出した状態のグラフ構造から算出する最長コスト経路を指す。タスク数7のタスクグラフを例にとり，依存関係のクリティカルパスの例をFigure 3に示す。

依存関係のクリティカルパスは，対象の問題をスケジュールした際に最も通信コストを必要とする，スケジュールにとってボトルネックとなる依存関係のパスを示している。従って，この依存関係のクリティカルパスに対して仮割り当てを行うことで，少なくとも最長コスト分の通信が発生するケースを排除することが可能となる。そのため，依存関係のクリティカルパスを仮割り当ての判断基準として用いれば，タスクの優先順序を変更することなく非効率なタスク割り当ての組み合わせを自動的に回避することが可能と考えられる。

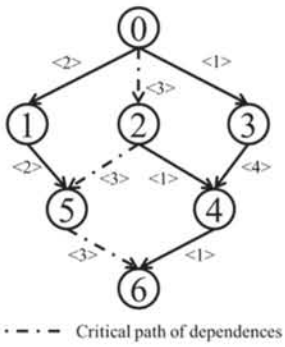

Figure 3 依存関係のクリティカルパスの例

以下にCPDの計算手順を示す。なお，リストスケジューリングに基づくヒューリスティックスの計算量の少なさを活かすため，タスクのプライオリティの決定にはCPと同様に優先順序付けを用いた。

I. 元の問題のタスクグラフから，依存関係のクリティカルパスを算出する
II. 依存関係のクリティカルパスの経路上にある全てのタスクを，経路上の直接先行タスクに従って仮割り当て
III. 割り当ての優先順序を決定するためにタスクのプライオリティをスタティックレベルから算出し，タスクのプライオリティリストを作成する
IV. 割り当て先のPEの選択に仮割り当てを用いて，リストスケジューリングを行う

4.5. CPDの単体評価の方法

本章で提案したCPDの性能を評価するために，関連する従来手法との比較によって計算量と解の厳密性のトレードオフに関する評価実験を行った。関連する従来手法には，CPDにおけるタスクのプライオリティの決定のベースとなっているCP，及び4.3節の仮割り当ての有効性を調査する検証実験で用いたHLFETを選択した。

実験に用いるタスクグラフは，タスクスケジューリング分野のベンチマークとして提案されている標準タスクグラフセット[9] を用い，タスク数 50，100，300 とし，通信コストを CCR=1.0 として各依存関係にランダムにコストを与えた。実験値には，各試行における 180 種類のタスクグラフによる実行結果の平均値を用いた。各問題を解く際の PE 数は，事前に 2～20 までの PE でスケジューリングを行い，最も良い makespan が得られた値を使用した。

実験環境を以下に示す。

OS ： Fedora 15 （Lovelock）

Memory ： 2GB

CPU ： Intel Core 2 Duo E8500 @ 3.16GHz × 2

4.6. CPD の単体評価の実験結果

CP，CPD，HLFET の 3 種のアルゴリズムを用いて，タスク数 50，100，300 の規模のスケジューリングを 180 種類ずつ解いた際の makespan の比較を行った実験の結果を Figure 4 に示す。図の横軸は問題の規模（タスク数）を表す。図の縦軸（makespan の比）は，各問題の規模ごとに CP を用いてスケジューリングした際の makespan を基準点（1.0）とし，同じ規模の問題に対してアルゴリズムを変えてスケジューリングを行った場合の結果を比で表す。

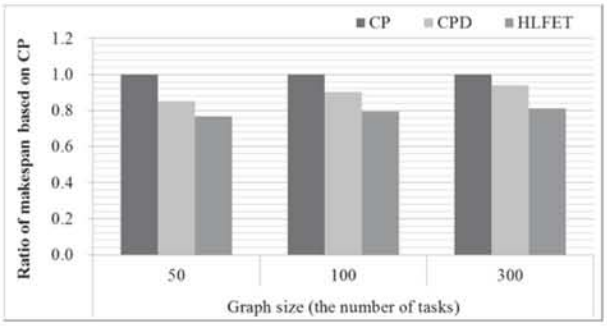

Figure 4 CPD と従来手法の makespan の比較

各タスク数で CP によるスケジュールの makespan を基準とした CPD の相対性能は，タスク数 50 のとき約 85%，タスク数 100 のとき約 90%，タスク数 300 のとき約 93%となり，いずれの問題規模の実験においても makespan が改善された。

提案した CPD は，makespan の性能について，いずれの問題規模においても CP と HLFET の中間程度の結果となった。

同条件で実行時間の比較を行った実験の結果を Figure 5 に示す。図の横軸は問題の規模（タスク数）を表す。図の縦軸（実行時間の比）は，各問題の規模ごとに CP を用いてスケジューリングした際の実行時間を基準点（1.0）とし，同じ規模の問題に対してアルゴリズムを変えてスケジューリングを行った場合の結果を比で表す。

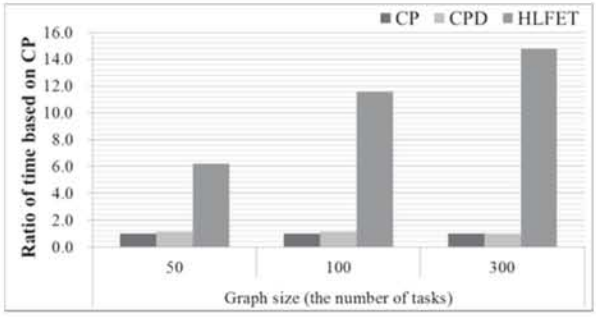

Figure 5 CPD と従来手法の実行時間の比較

各タスク数の問題規模で，CP によるスケジューリングの実行時間を基準とした CPD の相対性能は，タスク数 50 のとき約 16%，タスク数 100 のとき約 12%，タスク数 300 のとき約 2%となり，平均で 10%程度計算量が増加した。

HLFET を含む 3 種のアルゴリズムの対比としては，提案した CPD は，計算量の性能について，いずれの問題規模においても CP 以上 HLFET 未満の計算量を要する結果となった。

4.7. CPD の評価実験に対する考察

makespan の実験において，makespan の推移が CP, CPD, HLFET の順に短縮傾向にある点について考察を行う。タスクの選択順序を静的な解析情報から一意に決定する CP では，通信が考慮したヒューリスティックが用いられていないため，CPD が依存関係のクリティカルパスを仮割り当てで排除したことによる影響が CP との makespan の差として現れているものと考えられる。一方で，HLFET との対比では，CPD が仮割り当てによって通信を最適化した場合の組み合わせを含む全ての組み合わせを探索しているため，より厳密にスケジュールが短くなるタスクと PE の組み合わせが算出され，CPD との makespan の差を生じる要因になっていると考えられる。

makespan に関して以上の結果が得られた一方で，実行時間については，CPD の計算量がほぼ CP と変わらない結果となった。したがって，CPD は，より計算量の少ない CP や，計算量を多く払ってでもより厳密なスケジュールを求めようとする HLFET に対し，スケジューリングの計算量と厳密性のバランスに関する比較で，非常に効率的なスケジューリングが行われていると言える。

5．通信の割り当てを高速に行うアルゴリズムの設計と開発

本章では，アロケーションの工程において通信の最適化のために有効な手段を考察し，通信の割り当て先を効率的に走査するための割り当てアルゴリズム CEA を提案する。

対象の通信が割り当て可能な領域は暫定のスケジュール上に複数存在する場合があり，特定の組み合わせで生じる最適な割り当て位置をより少ない計算量で算出し，計算量を抑えたうえで厳密性の向上が可能となることを示す。

5.1.　割り当てステップの最適化の必要性

タスクの割り当て以上に，通信の割り当ては複雑である。なぜなら，通信の割り当ては，タスクの割り当てと同様に依存関係を考慮するだけでなく，割り当ての度に依存情報の送信側と受信側で共通した時間帯に目的の通信コスト分のアイドル時間が各々のスロットで連続していることを検査する必要があるためである。

既存のアルゴリズムの多くは，タスク及び通信がスケジュールに割り当てられる際の割り当てのタイミングを一意に決定している。その割り当てタイミングには，割り当て先の PE の対象スロットが最後にアイドル状態となった時刻以降で，依存関係の整合性を満たす最短の時刻が選択されていた。この選択方法を取るメリットとしては，タスク及び通信のスケジュールへの割り当て処理における計算量を最小限に抑えられる，ということが考えられる。しかし，通信を考慮したスケジューリングにおいてこの割り当て方法を用いた場合，スケジュールに冗長なアイドル時間が多く発生し，結果として makespan も長くなってしまうという深刻な問題がある。

前述の検査を用いて，通信の割り当て可能かつ最短の実行開始時刻を保証するアイドル領域 EAIT （Earliest Allocatable Idle Term）を特定するには，暫定解のスケジュールに既に割り当てられたタスク及び通信の間にあるアイドル領域を逐次に走査する必要がある。通信の発生量は問題規模の増加に比例するため，この走査に要する計算量はスケジューリングにおけるボトルネックとなることが予想される。したがって，この走査に関する計算量を可能な限り低減させるためには，効率的な割り当てアルゴリズムが必要となる。

5.2.　共通アイドル時間の導入

最小の計算量で対象の通信の最適な割り当て先 EAIT を特定するために，新たな定義を導入する。

先行タスク T_P と後続タスク T_S を順序付ける依存関係 D_{PS} がコスト α を持ち，T_P を処理した処理装置 PE_P と T_S を処理する処理装置 PE_S が異なるとき，D_{PS} による通信の割り当て走査対象となるアイドル領域は，T_P の処理が完了した時刻よりも後で，PE_P の送信スロットと PE_S の受信スロットが同時に単位時間以上の連続したアイドル時間を持つ全てのアイドル領域に相当する。ここで，割り当て対象の通信の先行依存タスクの実行完了後に，2つのスロットが同時にアイドル状態となるタイミングとその連続するアイドル時間を持つアイドル領域を，PE_P から PE_S への通信の「共通アイドル領域（Shared Idle Term）」と定義し，SIT_{PS} と表記する。

5.3.　共通アイドル時間を用いた EAIT の特定

前述の定義によって対象の通信の EAIT が高速に導出可能となることを，PE_P と PEs の送受信スロットの具体例を用いて示す。

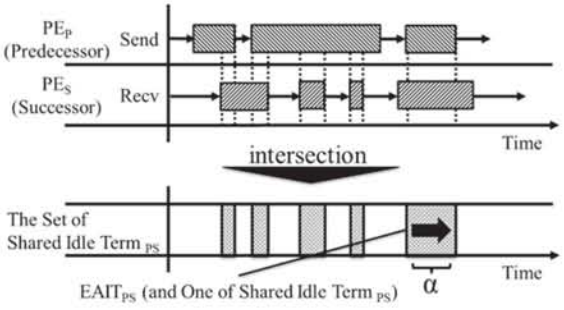

Figure 6 共通アイドル領域の抽出

Figure 6 の上部は，コスト α を持つ依存関係 DPS の通信を割り当てるとき，その先行依存タスクの実行完了以降の，PE_P の送信スロットと PE_S の受信スロットのアイドル領域を表している。これらのスロット内の矢印は DPS 以外の依存によって発生した通信のためにスロットが使用できない状態を示しており，斜線の帯の部分が現在の各スロットのアイドル領域である。

これらのアイドル領域の積から抽出される，共通アイドル領域の集合を Figure 6 の下部に示す。D_{PS} による通信の割り当てのために走査が行われる際は，共通アイドル領域を時刻の早いものから順にコスト α 以上のアイドル時間を持つか判定していく。発見されたコスト α 以上のアイドル時間を持つアイドル領域の集合が，D_{PS} の割り当て可能なアイドル領域の集合となる。

割り当て可能なアイドル領域の集合のうち最も開始時刻の早いアイドル領域 EAIT は，抽出された共通アイドル領域の集合を走査することで取り出しが可能である。

任意のスロットにおけるアイドル領域の集合が時系列順にリスト化された状態を保つことから，共通アイドル領域の集合は時系列順にリスト化されている。また，全ての共通アイドル領域は依存上の割り当て開始時刻の条件を満たしている。したがって，時系列順に整理された共通アイドル領域のリストを順方向に走査すれば，最小の計算量でEAITを得られる。

5.4. CEAアルゴリズム

共通アイドル領域を用いた通信の割り当て候補の限定操作を応用し，EAIT を高速に導出する割り当てアルゴリズムCEA（most Compact and Earliest Allocation）を提案する。

CEAは，依存情報の送信側の送信スロットと受信側の受信スロットが同時期に必要な通信コスト以上の時間に連続してアイドル状態となる部分にしか割り当てられない，という通信の特性を利用し，両スロットのアイドル領域が重複している部分だけを走査対象として通信の割り当てステップの効率化を実現している。

加えて，CEAを用いた割り当てでは，通信の組み合わせの選択が同一であっても，従来の割り当てタイミングの算出方法から得られるスケジュール以上に良質である（makespanが短い）ことが保証される。

以下にCEAの割り当てアルゴリズムを示す。

I. 割り当てる通信を任意の方法で1つ選択する
II. 先行タスクの処理が完了した時刻以降に存在する，先行タスクを処理したPEの受信スロットと後続タスクを処理することになるPEの送信スロットのアイドル領域の積集合を取り，共通アイドル領域の集合を得る
III. アイドル状態の開始時刻の昇順に共通アイドル領域を1つ選択する
IV. 選択したアイドル領域のアイドル時間の長さが手順Iで選択した通信のコスト以上ならば，手順VIへ
V. アイドル時間が通信コスト未満の場合，開始時刻の昇順に次のアイドル領域を選択し，手順IVに戻る
VI. 選択した共通アイドル領域がEAITに相当し，そのアイドル領域の開始時刻に対象の通信を割り当て
VII. 手順Iで選択した通信の割り当てが完了となる

5.5. CEAの単体評価の方法

本章で提案したCEAの性能を評価するために，既存アルゴリズムであるCPの通信の割り当てタイミングに，従来の方法を用いた場合とCEAを用いた場合の比較によって，計算量と解の厳密性のトレードオフに関する評価実験を行う。

CEAは割り当てPE数の変化によって大きく性能が変化する事が予想されるため，PE数は2～12PEまで，2つずつPEの数を増やしながら，CPにCEAを適用した場合と適用しなかった場合の実行時間とmakespanの比較を行った。

実験に用いるタスクグラフは，タスクスケジューリング分野のベンチマークとして提案されている標準タスクグラフセット[9]を用い，通信コストをCCR=1.0として各依存関係にランダムにコストを与えた。実験値には，各試行における180種類のタスクグラフによる実行結果の平均値を用いた。

実験環境は4.5節と同様である。

5.6. CEAの単体評価の実験結果

以下の実験結果では，従来の通信の割り当て方法を用いた場合のCPを「CP」，CEAを適用したCPを「CP+CEA」と表記する。

CP，及びCP+CEAを用いて，タスク数100，300の規模のスケジューリングを180種類ずつ解いた際のmakespanの比較を行う実験の結果をFigure 7に示す。図の横軸は問題の規模（タスク数）を表す。図の縦軸（makespanの比）は，PE数を2としてCPでスケジューリングした際のmakespanをタスク数の規模別の基準点（1.0）とし，同じ問題に対してアルゴリズムないしPE数の異なるスケジューリングを行った場合のmakespanを比で表す。

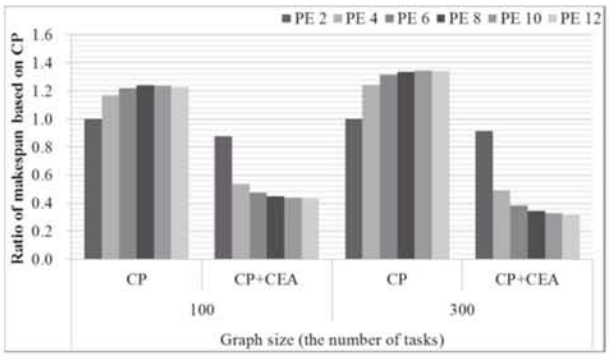

Figure 7 CPとCP+CEAのmakespanの比較

いずれのPE数，いずれのタスク数の問題でも，CEAを適用した場合にmakespanの短縮傾向が得られた。短縮傾向が最も強かったのは，PE数12でタスク数300の問題に対してCP+CEAでスケジューリングを行った場合で，同じ問題を解いたCPのmakespanと比較して約76%短縮されたmakespanを得た。短縮傾向が最も弱かったのは，同じ場合のPE数12の試行で，同じ問題を解いた

CP の makespan と比較して約 10%短縮された makespan を得た。

PE 数の増加に伴う変化では，CP+CEA は PE の数が増える毎に makespan を短縮するスケジュールを算出した。一方，CP では PE の数が増える毎に makespan が長くなる冗長なスケジュールが算出された。

同条件で実行時間の比較を行った実験の結果を Figure 8 に示す。図の横軸は問題の規模（タスク数）を表す。図の縦軸（実行時間の比）は，PE 数を 2 として CP でスケジューリングした際の実行時間をタスク数の規模別の基準点（1.0）とし，同じ問題に対してアルゴリズムないし PE 数の異なるスケジューリングを行った場合の実行時間を比で表す。

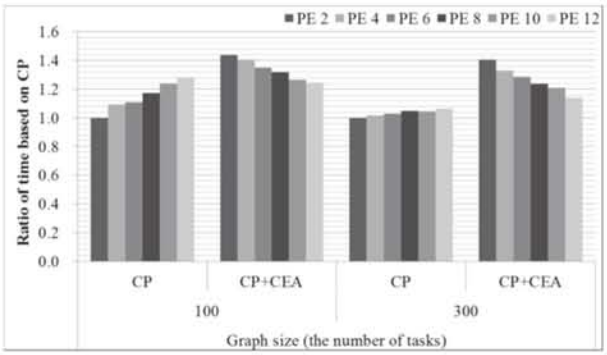

Figure 8 CP と CP+CEA の実行時間の比較

いずれのタスク数の試行においても，PE の増加に従って CP+CEA の実行時間に短縮傾向が得られた。ただし，同じ PE 数で行う CP の実行時間より短くなることはなかった。一方で CP を用いた結果では実行時間の増加に従って実行時間が長くなった。

同じ規模の問題を同じ PE 数で解いたとき，CP に対する CP+CEA の実行時間の差が最も小さかったのは，タスク数 300 の問題を PE 数 12 で解いた場合であり，このとき同じ問題を解いた CP に対し 14%程度実行時間が長くなった。また，平均では約 19%実行時間が長い結果となった。

5.7. CEA の実験結果及び設計に関する考察

makespan に関する全ての実験で，CEA によってスケジュールが改善されたことについて考察を行う。CEA は従来の割り当て方法以上の makespan を生成することが理論上保証されており，実験でもこの性質が確認された。特に makespan が短くなったケースでは，従来の割り当て手法と比較して約 76%もの makespan の短縮に成功していることから，makespan の改善に関して CEA が有効であるといえる。

実行時間の比較実験の結果から，CEA の相対的な計算量について考察を行う。これらの実験結果において CEA は，タスク数の増加に伴う計算量の爆発的増加を起こさず，PE 数の増加に対しては計算量の低下が見られたことから，非常に少ない計算量で EAIT を発見できているといえる。したがって，アイドル領域によるスロット情報の管理，及び TEA に通信を割り当てるヒューリスティックが通信を考慮したタスクスケジューリングの通信の割り当てに対して有効であると考えられる。

加えて，実行時間の比較実験で生じた，PE 数の増加に伴う CP+CEA の計算量の低下現象について考察する。対象の計算量低下現象は，PE 数の増加に伴う共通アイドル領域の変化に要因があると考えられる。PE 数を 2 としてスケジューリングを行う場合，一方の PE の送信スロットともう一方の PE の受信スロットは全く同一の割り当て状態となる。この場合，これらの PE のいずれが依存情報を送信・受信しても，その共通アイドル領域には各スロットのアイドル領域の全てが相当してしまう。したがって，PE 数が少ない試行では，相対的に計算量が増加していると考えられる。各 PE のスロットの状態は PE の増加に伴って，その他のより多くの PE と関連を持つことになるため，共通アイドル領域が絞りこまれ，EAIT を走査する計算量が少なくなる傾向にあると考えられる。

以上の実験結果と考察から，CEA は，その特徴として適用対象のアルゴリズムを幅広く選択可能であると考えられる。CEA は通信の選択順序を規定していないため，既存の多くのスケジューリングアルゴリズムとの併用が可能である。割り当て処理が従来の割り当て方法以外のアルゴリズムで規定されている場合には，そのアルゴリズムとの間で，厳密性と計算量のトレードオフを考慮し，割り当てアルゴリズムの選択を行うことが有効である。CEA を用いることで，既存のアルゴリズムのタスクや通信の選択に関する特性を生かしたまま，スケジュールのみを前倒しにすることで makespan の短縮効果が得られるため，CEA と処理が重複しないスケジューリングアルゴリズムにとって非常に有効な手段であると考えられる。

6. 複合評価

6.1. 提案手法

本章では，前述の 2 つのヒューリスティックスを用いたスケジューリングアルゴリズムを提案手法と呼称し（図中では CPD+CEA と表記），既存手法との比較による評価実験と考察を示す。

6.2. 評価方法

提案手法の評価は，計算量と厳密性のトレードオフに着目し，実行時間と解の相対性能を比較することで行った。実行時間は各手法がスケジュールを求めるまでに要する時間である。解の相対性能は各手法による makespan を CP との比によって示す。

比較対象として選択した既存手法は，CP, CP/DT/MISF, HLFET, ETF の 4 種のスケジューリングアルゴリズムである。

提案手法はアロケーションの工程で CEA を用いるために，割り当て PE 数の変化によって大きく性能が変化すると予想し，PE 数は 2～12PE まで 2 つずつ PE の数を増やしながら，実行時間と makespan の計測を行った。

実験に用いるタスクグラフは，タスクスケジューリング分野のベンチマークとして提案されている標準タスクグラフセット[9] を用いて，通信コストを CCR=1.0 として各依存関係にランダムにコストを与えた。実験値には，各試行における 180 種類のタスクグラフによる実行結果の平均値を用いた。

実験環境は 4.5 節と同様である。

6.3. 複合評価の実験結果

提案手法をはじめとする各手法を用いて，PE 数を 2～12 まで 2PE ずつ変化させた際に，各アルゴリズムがタスク数 300 の規模のスケジューリング問題を 180 種類ずつ解いた実行時間を計測した結果を Figure 9 に示す。図の横軸は各アルゴリズムを表す。図の縦軸（実行時間の比）は相対グラフとなっており，CP でスケジューリングした際の実行時間を PE 数別の基準点（1.0）とし，同じ PE 数の同じ問題に対して異なるアルゴリズムを用いてスケジューリングを行った場合の実行時間を比で表す。

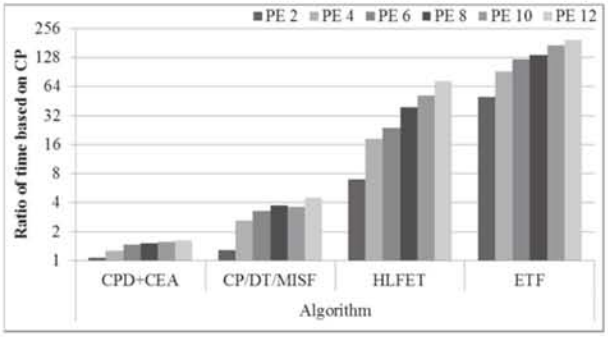

Figure 9 各アルゴリズムの平均実行時間の比較

提案手法は，CP との比較において最も計算量の増加が緩やかであった。最も CP との実行時間比が小さかったのは PE 数 2 でスケジューリングした場合で，このとき同条件の CP で計測した時間と比べて約 1.07 倍の計算時間を要した。最も実行時間比が大きかったのは PE 数 12 でスケジューリングした場合で，CP と比べ約 1.62 倍の計算時間を要した。

CP/DT/MISF，HLFET，ETF 及び提案手法の 4 種の手法で，CP と比較したときの実行時間の加速度が最も小さいのは提案手法であった。スケジューリング中に組み合わせの探索を行う HLFET や ETF は，問題規模や PE 数が増えることで実行時間が大幅に増加し，その増加の加速度が最も大きいのは ETF であった。特に，ETF を用いてタスク数 300 の問題を PE 数 12 で解いたとき，CP と比較して 195.30 倍の実行時間を要した。

同条件で makespan を計測した結果を Figure 10 に示す。図の横軸は各アルゴリズムを表す。図の縦軸（実行時間の比）は CP でスケジューリングした際の makespan を PE 数別の基準点（1.0）とし，同じ PE 数の同じ問題に対して異なるアルゴリズムを用いてスケジューリングを行った場合の makespan を比で表す。

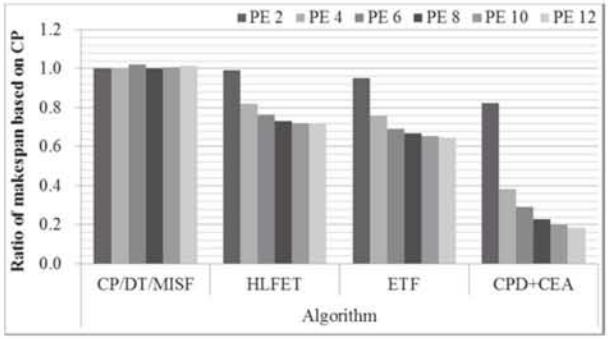

Figure 10 各アルゴリズムの平均 makespan の比較

HLFET，ETF，及び提案手法は PE 数の増加に伴い，makespan の改善傾向がみられた。この改善傾向は提案手法で最も強く，HLFET で最も緩やかな傾向であった。また，上記の 3 つのアルゴリズムのうち，提案手法は問題規模（タスク数）の増加によっても makespan の改善傾向がみられた。

全ての試行で提案手法が最も良い makespan を算出した。最も makespan が短縮されたのは提案手法を用いて PE 数 12 で解いた場合で，同条件の CP の makespan と比較して約 18 .6%まで短縮された。

6.4. 考察

PE 数の増加に伴う提案手法の計算量の増加速度が，CP 以外の比較対象のうち最も少ない点について考察を行う。提案手法では通信を最適化するために CPD 及び

CEA のアルゴリズムをスケジューリング中に用いていることから，問題規模の増加に伴う計算量の爆発的な増加を抑えることに成功していると考えられる。また，この結果と考察から，少量の計算量で通信の最適化を実現する CPD と CEA を組み合わせたことで，提案手法は問題規模に依らず少ない計算量を維持していると考えられる。

CP と提案手法の計算量の差については，CEA によるアロケーションの工程の計算量が主な要因として考えられる。その理由としては，一度のスケジューリングにおける CPD と CEA の使用回数の違いが挙げられる。一度のスケジュールにおいて，CPD は一度だけ使用され，CEA はアロケーションの工程で発生する通信の量に応じて複数回使用される。ここで，CPD と CEA の個別の評価実験において計算量への影響は CEA がより強かったことから，CEA による通信の割り当て処理が CP と提案手法の計算量の差を生じる主な原因として考えられる。

makespan に関しては，提案手法の makespan の推移が PE 数の増加に伴って最も強い短縮傾向を示した点に着目し，考察を行う。PE 数の増加にともなう makespan の短縮傾向は，暫定解のアイドル領域の発生量に影響を受けているのではないかと考えられる。PE 数とアイドル領域の発生量の関係についての考察は CEA 法の評価の際にも述べたが，提案手法については，それに加えてタスクの処理と通信時間のバランスによる影響が考えられる。提案手法では，CEA による割り当ての前に CPD による前処理が加えられているため，依存関係のクリティカルパス上に存在する通信が排除されている。パス上のタスクは仮割り当てによって同一の PE に割り当てられるため，対象の PE のタスク処理スロットはほぼ無駄なく使用されている状態となり，通信コストの支払いのためにスケジュールが伸びる，という組み合わせの発生を減少させているものと考えられる。また，仮割り当てによって通信の発生総量が減少する CPD 単体の特性も，提案手法の makespan の短縮に寄与していると考えられる。

以上の考察と，makespan を調査する全ての試行で，提案手法が最良のスケジュールを生成したことから，提案手法が用いる通信の最適化は問題規模に依らず機能していると考えられる。したがって，CPD と CEA の組み合わせは makespan の改善について相乗効果をもたらし，提案手法の makespan に関する性能を高く維持する要因になっていると考えられる。

7．おわりに

本稿では，依存関係を最適化する 2 つのヒューリスティックス CPD と CEA を提案し，その評価実験の結果と考察について報告した。

過剰なコストを持つ依存関係に対しては，仮割り当てという前処理を導入することで，対象の通信をスケジュール上に発生させないための「仮割り当て」という機構を提案した。さらに，依存関係を任意のタスクグラフから抽出するアルゴリズム CPD を構築し，対象の通信が発生する組み合わせを排除することで，計算量と makespan のトレードオフを非常に低いレートに抑えた。

通信の割り当てステップにおける makespan 短縮のための探索処理に対して，割り当て候補領域の管理方法と処理装置確定後の走査対象を限定する CEA を構築し，計算量及び makespan の改善を図った。結果，対象の通信が割り当て可能な最短時刻を非常に少ない計算量で算出可能となった。

さらに，これらの複合利用についても評価実験を行い，計算量及び厳密性の両面からその有用性を示した。

今後は，より makespan を短縮するためのヒューリスティックの発見と応用が望まれる。

謝辞

本研究の一部は，文部科学省戦略的研究基盤形成支援事業の補助を受けて行ったことをここに記し，謝意を表します。

参考文献

[1] Bernstein, D. Rodeh, and M. Gertner, I, “On the complexity of scheduling problems for paral-lel/pipelined machines”, IEEE Trans, Vol. 38 No. 9 pp.1308–1313, 1989.

[2] H. Kasahara, H. Honda, and S. Narita, “Parallel processing of near fine grain tasks using static scheduling on oscar”, IEEE Supercomputing ’90, 1990.

[3] Edward G. Coffman, “Computer and Job-shop Scheduling Theory”, John Wiley and Sons, 1976.

[4] J.J. Hwang, Y.C. Chow, F.D. Anger, and C.Y. Lee, “Scheduling precedence graphs in systems with in-terprocessor communication times”, SIAM Journal of Computing 18 （2）, 1989.

[5] Kwok, Y.-K. and Ahmad, I. “Benchmarking the task graph scheduling algorithms”, IPPS/SPDP, 1998.

[6] Masahiko U., Ryuji S., Munenori K., "Heuristic search based on branch and bound method for task scheduling considering communication overhead", IEEE Pacific Rim Con., 2011.

[7] Xiaohong K., Jun S., Bin Y. and Wenbo X., "An Efficient Quantum-Behaved Particle Swarm Optimization for Multiprocessor Scheduling", ICCS 2007, Volume 4487/2007 pp.278-285, 2007.

[8] T.C. Hu, "Parallel sequencing and assembly line problems", Operations Research, Novem-ber/December 1961 vol. 9 no. 6 pp.841-848, 1961.

[9] Standard Task Graph Set, http://www.kasahara.elec.waseda.ac.jp/schedule/, Kasahara Lab., Waseda Univ., 24 January 2012.