グリッドコンピューティングにおける制約条件付きスケジューリング問題のACOによる解法

目　　次

1．まえがき

2．問題の記述

3．ACOの適用

3.1　ACO適用の概要

3.2　処理順ノード空間

3.3　配置順ノード空間

3.4　割当てノード空間

3.5　ガントチャート

3.6　ノードの選択確率

3.7　蓄積フェロモンの更新

4．全体の処理手順

4.1　タスク処理順の決定

4.2　タスク配置順の決定

4.3　処理マシンの決定

4.4　ガントチャートへの配置

4.4　ガントチャートへの配置

5．数値実験

5.1　実験条件

5.2　実験結果

　5.3　蒸発率変更の効果

5.４　アリ数の差

　5.5　蒸発率の差

　5.6　容量制約の影響

　5.7　先行制約の影響

6．あとがき

参考文献

付録1　蒸発率変更ACOのプログラムの概要

付録2　蒸発率変更ACOのソースコード

1．まえがき

マシンのCPU，メモリ，ディスク，ソフトウェアなどを複数つなぐことと，それらを統合的に利用することとの間には，大きなギャップが存在する[1][2]．上記複数のリソースをネットワークで接続することはグリッドの要件ではあるが，それだけではリソースを有効に活用することは難しい．多数のタスクから構成される大規模なジョブ群に対して，多数のリソースの統合的活用を検討するのが，本報告で検討する計算グリッドにおけるスケジューリング問題である．

ここでは，

①ネットワークには，一つのスケジューラと多数のマシンが接続されている．

②ネットワークには，複数のジョブが同時に投入される．

③各ジョブは，複数のタスクから構成されている．

④各タスクはネットワーク上のいずれか一つのマシンで処理される．

⑤各マシンには，処理できるタスクに関して容量制約がある．

⑥各ジョブのタスク間には処理順に関する先行制約[3][4]が存在する場合がある．

⑦スケジューラは，すべてのタスクに対して，処理マシンと処理開始時刻を決定する．

⑧処理時間は整数とする．

との想定で，スケジューリング問題を考える．

ガントチャートによるスケジュール表現を想定すると，上記スケジューリング問題は，ジョブ内タスクの先行制約を考慮した全タスクのガントチャートへの配置順問題と全タスクのマシンへの割当て問題から構成される．

本研究では，このような問題を組み合わせ最適化問題として捉え，最適解探索の手法としては，組み合わせ最適化手法の一手法であり，幅広い適用範囲を持つアントコロニー最適化手法 (ACO) [5][6]の適用を提案する．ACOはメタヒューリスティクスの一つであり[7]，大まかな処理手順は決まっているが，解候補を表現するノード空間の表現方法，制約条件の実現方法，蓄積フェロモンの更新方法などの具体的な適用方法は問題に応じて考案しなければならない．また，通常のACOでは，ノード空間上のすべてのノード上のフェロモン蓄積量を一世代終了するごとに一定の蒸発率を用いて蒸発させる．しかし，後述するように，提案するACOでは制約を考慮するノード空間では解候補の選択対象から除外されるノードが存在する．そこで，選択される可能性のあるノードに対してのみ蒸発を行うACOを提案する．

2．問題の記述

図2.1に示すグリッド環境を考える．すなわち，マシンを，同時に処理するジョブを，ジョブを構成するタスクをとする．マシンの処理速度を，タスクの処理量をとする．タスクがマシンで処理されるときの処理時間は

　　　 　　　　　　　　　　(1)

である．マシンの容量を，タスクの処理に必要なマシン容量をとする．もし

　　　　　 　　　　　　　　　　(2)

なら，タスクはマシンに割当てることができない．タスクの処理完了時刻をとすると，すべてのジョブの処理が完了する時刻は，すべてのタスクの処理が完了する時刻

　　　　　　　　　　　　　　　 (3)

となる．このとき，対象とするスケジューリング問題は「評価値(3)，すなわち，すべてのジョブの中で最も遅い処理完了時刻が最早となるように，すべてのタスクについて処理マシンとそのマシンでの処理開始時刻を決めること」となる．

図2.1　グリッド環境

3．ACOの適用

3.1　ACO適用の概要

ACOは，組み合わせ最適化手法の一つで，アリがフェロモンに基づいて餌を探索する行動を模した多点探索法である．解候補の空間をノード空間とし，複数のアリエージェントが，ノード上に蓄積されたフェロモン量をたよりに最適解の探索を行う．フェロモンはアリが探索した解の評価に応じてノード空間に散布され，蓄積されるフェロモン量は変化する．このことによって，高評価の解が選択されやすくなる．複数のアリによる解の探索とフェロモンの散布を1サイクルとし，あらかじめ設定しておいたサイクル数だけ繰り返す．

スケジューリング問題にACOを適用するときの基本的考え方はつぎのとおりである．まず，解候補を表現するノード空間として，ジョブ内先行制約を踏まえたジョブ内タスクの処理順を決めるノード空間（処理順ノード空間），処理順を踏まえてガントチャートへのすべてのタスクの配置順を決めるノード空間（配置順ノード空間），容量制約を考慮してすべてのタスクに対して割当てマシンを決めるノード空間（割当てノード空間）を準備する．処理順ノード空間と割当てノード空間の数は（ジョブ数），配置順ノード空間の数は1とする．そして，ガントチャート上に先行制約を考慮してタスクを前詰め配置し，各タスクの処理開始時刻を得るものとする．

3.2　処理順ノード空間

　まず，ジョブ内先行制約を踏まえたジョブ内タスクの処理順を決める処理順ノード空間の構成について記す．各ジョブの処理順ノード空間は，図3.1に示すようなノード空間で，ジョブにおけるノードは，番目に割当てられるタスクがであることを意味する．したがって，順序ノード空間はの正方形の空間となる．なお，一度選択されたタスク番号は，それ以降の処理において，選択候補から除外する．また，アリが選択できるのは，先行制約を満たすタスク番号のみである．

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | ･･･ |  | ･･･ |  |
| タスク番号 | 1 | 1 | 1 | ･･･ | 1 | ･･･ | 1 |
| 2 | 2 | 2 | ･･･ | 2 | ･･･ | 2 |
|  |  |  | ･･･ |  | ･･･ |  |
|  |  |  | ･･･ |  | ･･･ |  |

図3.1　ジョブの処理順ノード空間

アリが選択できるノードをタスク選択行列で表わす．たとえば，図3.2に示すようなジョブの先行制約を考える．6個のタスクが3層より構成されており，上下層のタスク間では先行制約が存在するが，同一層のタスク間では先行制約は存在しないものとする．GA，ACOなどの多点探索法はこのような基本構成の繰り返しである．最初のタスクはのみが選択可能で，2番目から5番目までのタスクとしてはのいずれかが選択可能，そしてがすべて選択された後にが選択可能とする．

このとき行列の初期状態は図3.3のようになる．まず，(a)先行制約条件にしたがって，選択できるノードを1，他は0とする．(b)最初に選択できるノードはだけである．(c)つぎに選択可能なノードは(2,2)-(5,2)である．(3,2)が選択されたとすると，(3,3)以降の値を0に変更する．(d)3番目に選択可能なノードは(2,3)(4,3)(5,3)である．(5,3)が選択されたとすると，(5,4)以降の値を0に変更する．以降，同様に(e)(f)(g)を得る．

　　　　　　第1層

　　　 第2層

　　　　　　第3層

図3.2　ジョブの先行制約の例

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 0 |
| 3 | 0 | 1 | 1 | 1 | 1 | 0 |
| 4 | 0 | 1 | 1 | 1 | 1 | 0 |
| 5 | 0 | 1 | 1 | 1 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(a) 初期選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 0 |
| 3 | 0 | 1 | 1 | 1 | 1 | 0 |
| 4 | 0 | 1 | 1 | 1 | 1 | 0 |
| 5 | 0 | 1 | 1 | 1 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(b) 最初の処理タスク決定後の選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 0 |
| 3 | 0 | ② | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | 1 | 1 | 0 |
| 5 | 0 | 1 | 1 | 1 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(c) 2番目の処理タスク決定後の選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 0 |
| 3 | 0 | ② | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | 1 | 1 | 0 |
| 5 | 0 | 1 | ③ | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(d) 3番目の処理タスク決定後の選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 0 |
| 3 | 0 | ② | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | ④ | 0 | 0 |
| 5 | 0 | 1 | ③ | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(e) 4番目の処理タスク決定後の選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | ⑤ | 0 |
| 3 | 0 | ② | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | ④ | 0 | 0 |
| 5 | 0 | 1 | ③ | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

(f) 5番目の処理タスク決定後の選択行列

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 処理順序 | | | | | |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 |
| タスク番号 | 1 | ① | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 1 | ⑤ | 0 |
| 3 | 0 | ② | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | ④ | 0 | 0 |
| 5 | 0 | 1 | ③ | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | ⑥ |

(g) 最後の処理タスク決定後の選択行列

図3.3　タスク選択行列の推移 (○印内の数字は割当て順を示す)

以上より，アリがノード空間を左から右へ移動することにより，先行制約を満たすタスクの処理順序が決まる．ただし，アリがノード空間を左から右へ移動するのに伴って選択行列の要素の値は動的に変化し，しかも，変化の様子はアリごとに異なることに注意．

3.3　配置順ノード空間

各ジョブの先行制約を満たしつつ作業のガントチャートへの配置順を決めるノード空間は，図3.4に示すようなノード空間とする．ノードは，番目に配置されるタスクがジョブに属していることを意味する．配置順ノード空間はの横に非常に長い空間となる．ここに， である．すなわち，はスケジュールの対象となるジョブに含まれるタスクの総数である．また，ジョブごとにタスクリストを準備し，3.2で決定された処理順にタスクを登録する．

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 配置順 | | | | | |
|  |  | 1 | 2 | ･･･ |  | ･･･ |  |
| ジョブ番号 | 1 | 1 | 1 | ･･･ | 1 | ･･･ | 1 |
| 2 | 2 | 2 | ･･･ | 2 | ･･･ | 2 |
|  |  |  | ･･･ |  | ･･･ |  |
|  |  |  | ･･･ |  | ･･･ |  |

図3.4　配置順ノード空間

ノード空間とタスクリストを使った作業の配置順決定はつぎのようである．まず，ノード空間でノードが選択されると，配置するタスクが属するジョブが決まる．つぎに，当該ジョブのタスクリスト中の先頭タスクを取り出し，配置するタスクとする．同時に，当該タスクはタスクリストから削除する．この手続きを繰り返すことにより，各ジョブに対して，タスク間の先行制約を満たすタスクが順次取り出せる．なお，タスクリストが空となったジョブは，空となった時点以降では選択されないものとする．このジョブの選択可能性を記述する変数としてジョブ選択ベクトルを導入する．ジョブが選択可能であるとき，選択できないときとする．

3.4　割当てノード空間

　タスクの割当てマシンを決めるノード空間は，図3.5に示すようなノード空間で，ジョブにおけるノードは，タスクをマシンに割当てることを意味する．したがって，割当てノード空間はの長方形の空間となる．なお，アリが選択できるのは，容量制約を満たすマシン番号のみである．すなわち，もし，なら，タスクはマシンに割当てることができないので，ノードの初期フェロモン量を0，それ以外の割当て可能なノードの初期フェロモン量は適当な正の値に設定する．これにより，ノードの初期選択確率は0となり，以降のサイクルにおいても選択されることは無い．

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | タスク番号 | | | | | |
|  |  | 1 | 2 | ･･･ |  | ･･･ |  |
| マシン番号 | 1 | 1 | 1 | ･･･ | 1 | ･･･ | 1 |
| 2 | 2 | 2 | ･･･ | 2 | ･･･ | 2 |
|  |  |  | ･･･ |  | ･･･ |  |
|  |  |  | ･･･ |  | ･･･ |  |

図3.5　ジョブの割当てノード空間

以上より，アリがノード空間を左から右へ移動することにより，容量制約を満たす各タスクの割当てマシンがタスク番号順に決まる．

3.5　ガントチャート

上記のノード空間から解候補が得られると，縦軸をマシン番号，横軸を時刻とするガントチャート上に既配置済みのタスクに対して先行制約を考慮して前詰めでタスクを配置する．すなわち，セミアクティブスケジュール[8]となるように各タスクを配置する．これにより，式(3)の評価値が決まる．

3.6　ノードの選択確率

アリは，ノード上に蓄積しているフェロモン量を参照し，ルーレット選択によって移動するノードを選択する．すなわち，処理順ノード空間，配置順ノード空間および割当てノード空間の蓄積フェロモン量をそれぞれおよびとすると，処理順ノード空間，配置順ノード空間および割当てノード空間のノードの選択確率およびはそれぞれ式(4)-(6)で求められる．ただし，はサイクル数を表す．

　　　(4)

　　　　　 (5)

　　　　　(6)

3.7　蓄積フェロモンの更新

通常のACOでは，一世代終了するごとに，一定の蒸発率を用いて，処理順ノード空間，配置順ノード空間および割当てノード空間のノード上のフェロモン蓄積量を蒸発・減少させることになる．ただし，本研究では，処理順ノード空間では先行制約のため選択の対象外となるノードが存在する．また，配置順ノード空間でも各ジョブの選択回数制約から選択の対象外となるノードが存在する．

そこで，制約上除外されるノードに対しては蓄積フェロモンの蒸発を行わず，選択される可能性のあるノードに対してのみ蒸発を行うという蒸発率の変更を考えた．除外ノードはアリの行動によって動的に変化する．そのため，処理順ノード空間では一サイクルの蒸発率を

　　　　　　　　　　(7)

とする．ここに，選択行列の値はアリごとに異なることを明示してとしている．選択行列の値がのときは，アリに対してノードの蓄積フェロモンは蒸発しないが，のときは，アリに対してだけ蒸発する．同様に，配置順ノード空間では一サイクルの蒸発率を，ジョブ選択ベクトルを用いて

　　　　　　　　　　　(8)

と変更する．なお，蒸発率を一定値とする通常のケースは，式(7)(8)において，とした場合に相当することがつぎの近似式

，　　　　　　(9)

よりわかる．

つぎに，アリの評価値に応じたフェロモン量をアリが通過したノード上に撒く．そして，すべてのアリがフェロモンを撒き終わると，処理順ノード空間，配置順ノード空間および割当てノード空間のノード上の蓄積フェロモン量は

　　(10)

　　(11)

 　(12)

となる．ここに，はアリ数，はアリが

 　　 　 (13)

　　　　　　(14)

　　　　　(15)

である．また，は適当な正の係数で，評価値が大きい (すなわち式(3)より評価が悪い) 場合に，フェロモンの散布量を少なくするために導入するものである．そして，式(10)-(12)をつぎのサイクルにおける式(4)-(6)のノード選択のためのフェロモンとする．

4．全体の処理手順

4.1　タスク処理順の決定

まず，処理順に関する先行制約を実現するための選択テーブル生成のアルゴリズムを示す．ジョブの先行制約は図4.1のように与えられているものとする．すなわち個のタスクが個のグループ

　　(16)

に階層的に属するものとする．そして，上下層のタスク間では先行制約が存在するが，同一層のタスク間では先行制約は存在しないものとする．

このとき，ジョブのタスクの処理順決定および選択行列生成のアルゴリズムはつぎのA1- A10となる．

A1：とする．

A2：ジョブのタスク選択行列の初期値として図4.2に示すブロック対角行列を作る．

A3：とする．

A4：ジョブの処理順ノード空間の列目のノードの選択確率を式(4)で計算する．

A5：ジョブの処理順ノード空間の列目のノード（タスク）をルーレット選択する．選択されたタスクをとする．

A6：選択されたタスクの行の列以降の選択行列の要素をすべて0にする．

A7：とする．

A8：もしならA4に戻る．ならA9へ進む．

A9：とする．

A10：もしならA2に戻る．なら4.2の配置順処理へ進む．

　　　 　 　第1層

　　 　　　第2層



　　　　 第層

図4.1　先行制約の階層表現

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | … |  |
|  |  | 0 | … | 0 |
|  | 0 |  |  |  |
|  |  |  |  | 0 |
|  | 0 | … | 0 |  |

図4.2　ジョブのブロック対角行列

4.2　タスク配置順の決定

　各ジョブでの先行制約を考慮したタスクの処理順が決まったので，ここでは，すべてのタスクのガントチャートへの配置順を決めるアルゴリズムをB1-B13に示す．なお，以下では番目に配置するタスクをと表記する．

B1：とする．

B2：ジョブのタスクリストに4.1で決定した処理順にタスクを登録する．

B3：ジョブ選択ベクトルをとする．

B4：とする．

B5：もしならB2に戻る．ならB6へ進む．

B6：とする．

B7：配置順ノード空間の列目のノードの選択確率を式(5)で計算する．

B8：配置順ノード空間の列目のノード（ジョブ）をルーレット選択する．選択されたジョブをジョブをとする．

B9：選択されたジョブのタスクリスト中の先頭タスクを取り出し，番目に配置するタスクとする．

B10：ジョブのタスクリストからB9で取り出したタスクを削除する．

B11：もしタスクリストが空となれば，ジョブ選択ベクトルをとし，B12へ進む．空とならなければB12へ進む．

B12：とする．

B13：もしならB7に戻る．なら4.3の処理マシン決定処理へ進む．

4.3　処理マシンの決定

各タスクを処理するマシンを決定するアルゴリズムは下記のC1-C9である．なお，以下では4.2で決めた番目に配置するタスクを処理するマシンをと表記する．

C1：とする．

C2：とする．

C3：ジョブの割当てノード空間の列目のノードの選択確率を式(6)で計算する．

C4：ジョブの割当てノード空間の列目のノードをルーレット選択する．選択された処理マシンをとする．

C5：とする．

C6：もしならC3に戻る．ならC7へ進む．

C7：とする．

C8：もしならC2に戻る．なら，タスクの配置順に対応して処理マシンもタスクの配置順に並べ，とする．

C9：4.4のガントチャートへの配置へ進む．

4.4　ガントチャートへの配置

上記A1-C8ですべてのタスクのガントチャートへの配置順と処理マシンが決まったので，つぎにガントチャートでの先行制約を考慮したタスクの前詰め配置のアルゴリズムについて示す．なお，はジョブの層に属するタスク番号の最小値である．

ガントチャートへの配置順と処理マシンの対を用いると，アルゴリズムはつぎのD1-D15となる．

D1：ジョブの層の処理終了時刻をとする．

D2：マシンの処理終了時刻をとする．

D3：ジョブの層番号をとする．

D4：ジョブの配置済みタスク数をとする．

D5：ジョブの仮処理終了時刻をとする．

D6：とする．

D7：番目に選ばれたタスクのジョブ番号をとする．

D8：タスクの処理時間を式(1)で計算する．

D9：タスクの処理終了時刻をとする．（処理開始時刻はである．）

D10：マシンの処理終了時刻をとする．

D11：ジョブの仮処理終了時刻をとする．

D12：ジョブの配置済みタスク数をとする．

D13：もしなら，ジョブの層の処理終了時刻を，ジョブの層番号をとし，D14へ進む．ならD14へ進む．

D14：とする．

D15：もしならD6に戻る．なら式(3)の評価値を各ジョブの最終層の処理終了時刻の中で最も遅い時刻とし，ガントチャート処理を終了する．

4.5　探索アルゴリズムの全容

　探索アルゴリズムの全容は4.1-4.4で示した処理を内部に含む形でつぎのE1-E12となる．

E1：グリッド環境，ACO，ジョブ（タスク）に関するパラメータを設定する．

E2：サイクル数をとする．

E3：すべての処理順ノード空間に初期フェロモンを散布する．

E4：配置順ノード空間に初期フェロモンを散布する．

E5：すべての割当てノード空間に容量制約式(2)を考慮して初期フェロモンを散布する．

E6：アリ番号をとする．

E7：A1-D15を実行する．

E8：とする．

E9：もしならE7に戻る．ならE10へ進む．

E10：すべての処理順ノード空間，配置順ノード空間およびすべての割当てノード空間の蓄積フェロモンを式(10)-(12)を用いて更新する．

E11：とする．

E12：（最大サイクル数）ならE6に戻る．ならを最良アリとし，最良アリの移動ルートから最良スケジュールを得る．

5．数値実験

5.1　実験条件

グリッドコンピューティングのモデルとして，つぎの表5.1，5.2に示すモデルを想定した．

表5.1　ジョブの条件

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ジョブ番号 | 階層番号 | | | タスク番号 | 処理量 | 必要容量 |
| Case１ | Case 2 | Case 3 |
| 1 | 1 | 1 | 1 | 1 | 100 | 50 |
| 2 | 110 | 60 |
| 2 | 3 | 120 | 70 |
| 2 | 4 | 130 | 80 |
| 3 | 5 | 140 | 90 |
| 6 | 150 | 100 |
| 2 | 3 | 4 | 7 | 160 | 50 |
| 8 | 170 | 60 |
| 5 | 9 | 180 | 70 |
| 4 | 10 | 190 | 80 |
| 6 | 11 | 200 | 90 |
| 12 | 210 | 100 |
| 2 | 同　上 | | | | | |
| 3 | 同　上 | | | | | |

表5.2　マシンの条件

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| マシン番号 | 処理速度 | 容　量 | | |
| Case A | Case B | Case C |
| 1 | 100 | 100 | 80 | 60 |
| 2 | 90 | 100 | 85 | 70 |
| 3 | 80 | 100 | 90 | 80 |
| 4 | 70 | 100 | 95 | 90 |
| 5 | 60 | 100 | 100 | 100 |
| 容量不足率 | | 0 | 0.2 | 0.33 |

ただし，表5.2の容量不足率とは，タスクのマシン必要容量に対してマシンの容量が不足している比率のことで

　　　　　　(17)

で求めたものである． ACOの条件を表5.3に示す．

表5.3　ACOの条件

|  |  |  |
| --- | --- | --- |
| アリ数 | | 50，20 |
| 世代数 | | 2000 |
| 初期フェロモン ，， | | 100 |
| 蒸発率 | | 0.1，0.2 |
| 散布式係数 |  | 100 |
|  | 2 |

5.2　実験結果1

　アリ数（）を50と20，蒸発率（）を0.1と0.2に設定し，蒸発率に変更式(7)(8)を用いた場合と式(9)を用いた場合を，Cases 1A，2A，3A，1B，2B，3B，1C，2C，3Cについて各10回試行した最良解（min）と最悪解（max）の世代平均の世代推移をそれぞれ図5.1-5.9に示す．