スケジューリング問題における探索方式

2K−1

湯上　伸弘　　原　裕貴
(株) 富士通研究所

1　はじめに

我々は、現在スケジューリング問題向けのエキスパートシェルの研究をおこなっている。ここで扱うスケジューリング問題とは、ジョブの集合および各ジョブを構成する手順の集合、各手順を処理する資源、処理時間、開始可能時刻、締切時刻、および手順間の先行関係が与えられたとき、各資源が、その処理すべき手順を、どういう順序で処理するかを決定する問題である。決定された順序を“計画”という。このような問題はＯＲで解くこともできるが、より複雑な制約を扱うことのできるシステムのための準備として、線形方程式、線形不等式処理部をそなえた探索方式の試作・評価をおこなったので、報告する。なお今回は制約充足のみを考え、最適化は考慮しない。

2　スケジューリング問題の解法

従来のシステムでは、各手順の開始時刻を順番に決定していくことによって、探索をおこなっていた。この方法では、例えば、計画期間の中間におこなわれる手順の開始時刻から決定しようとすると、その決定が誤りであったとき、開始時刻をどれだけずらすかについて、時刻の連続性から無限の選択肢があために、探索が有限の時間で終了しないおそれがある。この問題を避けるためには、各資源について、最初におこなう手順の開始時刻から決めていく（前詰め）か、最後におこなう手順の開始時刻から決め（後詰め）ていけばよい。しかし、前詰め（後詰め）では、計画期間の中間にジョブ（手順）が集中しているような場合でも計画期間の最初（最後）から計画をたてていかなければならず、効率的な探索ができない場合があった。

A Search Method in Scheduling Problems
Nobuhiro Yugami , Hirotaka Hara
FUJITSU LABORATORIES Ltd.

それに対して、今回、計画期間の任意の部分から探索を開始でき、かつ探索が有限時間で終了することが保証される探索法として、次のような探索方式を考えた。すなわち、同一の資源で処理される2つの手順の前後関係を仮定していくことにより探索をすすめるという方法である。この方法では、探索の各ノードは手順間の順序に関する仮説の集合で表現される。この仮説集合の無矛盾性は、各仮説を表現する不等式系の可解性を調べることによってチェックされる。

ここで、前後関係を表現する不等式とは、例えば、仮説

“手順1-1は手順2-1の前である”

に対しては、

手順1-1の開始時刻＋手順1-1の処理時間
≦　手順2-1の開始時刻

である。

この前後関係に関する仮説集合は、どの手順間の前後関係から仮定したかによらないから、計画の任意の部分から探索をおこなうことができる。また、この前後関係の集合の数は明らかに有限だから、探索は有限時間で終了する。

3　探索木について

前節で述べたように、探索は、2つの手順の前後関係を仮定していくことによって進められるが、できるだけ無駄な仮説集合がつくられるのを防ぐために、以下のように探索をおこなうことにした。

すなわち、仮説集合にあらわれる手順のうち、同一資源で処理される手順については、その全順序が完全に仮定されているようにする。

具体的には、既に仮定されている手順の全順序

の、隣接する２個の手順の間にあらたな手順を割り込ませていくことにより、探索をすすめる。

このときの探索木のノード数は、例えば、資源１個、手順数N個のときは、

$$\sum_{m=0}^{N} m!$$

で与えれる。それに対して、前詰め（後詰め）の場合の探索木のノード数は、

$$\sum_{m=o}^{N} N!／(N-m)!$$

である。この両者を比較すると、本方式のほうが、ノード数が１／２（N＝５）から１／ｅ（N＝∞）と少なくなることがわかる。図１は、N＝３の場合の前詰め法と本探索法の探索木である。

探索木の葉の数は同じであるから、この差は中間ノードの差である。すなわち、あるノードの下にある葉の数の平均は、本方式のほうが多い。そのため、１個のノードの矛盾により枝刈りされる葉の数が多くなり、効率的な枝刈りがおこなわれる可能性がある。ただし、それだけ１個のノードのもつ自由度が大きいわけであるから、あるノードが矛盾となる確率は小さくなる。このどちらが実際の探索時間に効くかは、問題の性質に依存する。

4　不等式の可解性

この探索方式をおこなうためには、不等式系の可解性を高速に調べる必要がある。今回は、Sup-Inf法を用いた。ただし、扱う不等式が、

$$T_i - T_j \leqq \text{Const.}$$

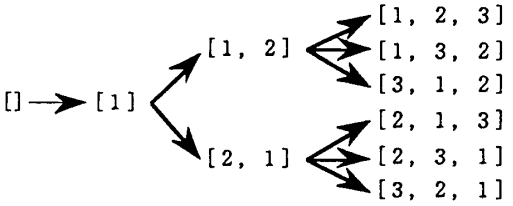
という形に限定されるから、それにあわせて特殊化して用いている。

5　評価

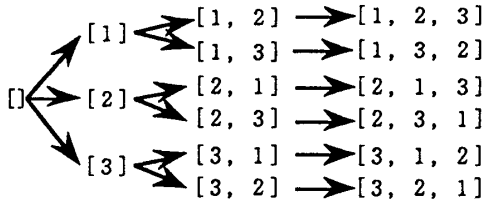
図２は、本探索方式と後詰め法とで、全解探索をおこなうのに必要な時間を比較したグラフである。問題としては、資源数３、ジョブ数４（ジョブ１個は手順３個からなる）の場合について、ジョブの締切時刻、所要時間などを変化させたものを与えている。

解である確率（解の数／並べ方の数）が１であるとき、すなわち、全ての並べ方が解であるときは、ほとんど探索時間に差はないが、解の数が減少するにつれて、本方式のほうが、後詰めに比較して短時間で探索を終了している。これは、探索木の比較的上部で枝刈りが効率的におこなわれているためだと思われる。

なお、この探索時間の比較は、Sun-4上のQuintus Prologを用いておこなった。



(a) 本方式による探索木



(b) 前詰め法による探索木

図１　探索木の例

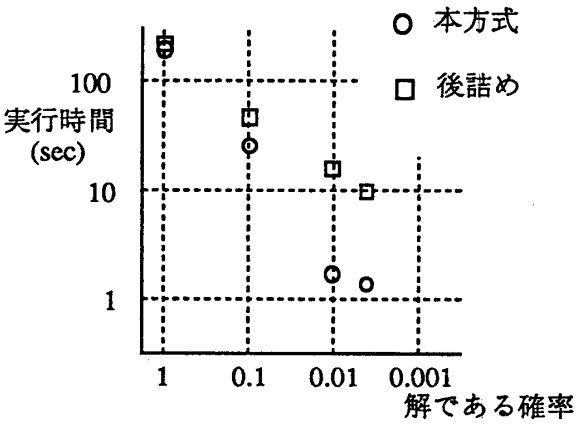


図２　全解探索の所要時間

参考文献

［１］溝口文雄、古川康一、J-L.Lassez
「制約論理プログラミング」　共立出版