社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－MPS－67 (11)
2007－BIO－11 (11)
2007／12／20

資源と作業順序の制約を拡張したスケジューリング問題に対する下界値の計算法

草部 博輝†, 中森 眞理雄†,

† 東京農工大学工学府

資源制約付プロジェクトスケジューリング問題 (Resource Constrained Project Scheduling Problem : RCPSP) は，job-shop のようなモデルの一般形としてよく知られるモデルである．本稿は，RCPSP の先行制約と資源制約を変化させた RCPSP/τ+ モデルを取り扱い，その下界値の計算法を提案する．評価は比較的小規模のインスタンスを用い，最適解との比較を行うことにより行う．

Lower Bounding Method for the RCPSP/τ with Time Lags

Hiroaki KUSAKABE†, Mario NAKAMORI†,

† Graduate School of Engineering, Tokyo University of Agriculture and Technology

Resource-constrained project-scheduling problem (RCPSP) is a general model of several classical scheduling models like a job-shop. In this paper, we suggest the scheduling model RCPSP/τ+, which is added the time windows to the model of RCPSP/τ having the changing of limit of renewable resources in project term and of requirement of renewable resources in each activity's processing time. We present a lower bounding method for the RCPSP/τ+ and evaluate the lower bound accuracy comparing the oplitmal solution.

1 はじめに

資源制約付きプロジェクトスケジューリング問題 (Resource Constrained Project Scheduling Problem : RCPSP)[1] は，job-shop スケジューリングの一般形として知られるモデルである．本章では，RCPSP の変形モデルである RCPSP/τ+ について説明する．

プロジェクトは n 個のアクティビティから構成され，各アクティビティには，$0, 1, \ldots, n-1$ の番号が与えられている．2つのアクティビティ $0, n-1$ はそれぞれ，プロジェクトの開始と完了を表すダミーである．アクティビティの処理実行のために，一般的に複数種類の資源が提供される．これらの資源は再生型資源であり，何度でも再利用可能である．各資源は，時刻毎で利用可能量が異なることがある．各アクティビティが要求する資源量は処理開始からの経過時間によって変化することがある．さらに，作業順序が与えられたアクティビティ間に，2種類のタイムラグに関する制約が課されることがある．これらはそれぞれ，次のような制約である．2つのアクティビティ i, j を考える．これらには，i が j に先行するという作業順序が与えられている．アクティビティ j は i の処理完了後，ある猶予時間が経過するまでに処理を開始しなければならない．これを**猶予制約** (within constraint) と呼ぶ．また，アクティビティ j は i の処理完了後，ある待機時間が経過してからでないと処理を開始できない．これを**待機制約** (after constraint) と呼ぶ．アクティビティの処理時間，各時刻における利用可能資源量，アクティビティの処理中に占有される資源量，作業開始の順序関係，猶予時間および待機時間は全て既知である．これらの条件の下でプロジェクトの完了に必要な時間，すなわちアクティビティ $n-1$ の完了時刻を最小化する．

2 0-1 整数問題への定式化

RCPSP/τ+ は，次のような 0-1 整数計画問題に定式化できる．

minimize

$$z = \sum_{t=0}^{t_{max}-1} t x_{n-1t} \tag{1}$$

subject to

$$\sum_{t=0}^{t_{max}-1} t x_{st} \leq \sum_{t=0}^{t_{max}-1} t x_{jt} + p_j + w_{js}, \quad j \in J,\ s \in S_j \tag{2}$$

$$\sum_{t=0}^{t_{max}-1} tx_{st} \geq \sum_{t=0}^{t_{max}-1} tx_{jt} + p_j + a_{js}, \quad j \in J,\ s \in S_j \tag{3}$$

$$\sum_{j=0}^{n-1} \sum_{u=0}^{\min(t,p_j-1)} d_{jru} x_{j(t-u)} \leq l_{rt}, \quad t \in T, r \in R \tag{4}$$

$$\sum_{t=0}^{t_{max}-1} x_{jt} = 1, \quad j \in J \tag{5}$$

$$x_{jt} \in \{0,1\}, \quad j \in J,\ t \in T \tag{6}$$

各記号の意味は，以下の通りである．

- n：全アクティビティ数．
- m：全資源種類数．
- J：全アクティビティの集合．$J = \{0, \ldots, n-1\}$
- R：全資源種類の集合の集合．$R = \{0, \ldots, m-1\}$
- t_{max}：プロジェクト期間．
- T：離散で定義される時刻の集合．$T = \{0, \ldots, t_{max}-1\}$
- p_j：アクティビティjの処理時間．
- S_j：アクティビティjの後続アクティビティの集合．
- w_{js}：アクティビティ$s \in S_j$の処理開始の猶予時間．
  アクティビティsは，jの処理完了後w_{js}単位時間経過するまでに処理を開始しなければならない．
- a_{js}：アクティビティ$s \in S_j$の処理開始の待機時間．
  アクティビティsは，jの処理完了後a_{js}単位時間経過しなければ処理を開始できない．
- d_{jru}：アクティビティjが，処理開始後，u単位時間経過時に要求する資源rの量．
- l_{rt}：時刻tにおける，資源rの利用可能量．
- x_{jt}：0－1変数．アクティビティjが時刻tに処理を開始されるならば1，それ以外は0．

3 下界値の計算

本章ではRCPSP/τ+に対する下界値の計算方法を示す．提案する手法は，猶予制約，待機制約および資源制約をラグランジュ緩和する手法と，資源制約の違反に対して待機制約を追加する手法である．また単純な方法として，制約(2)を無視し，複数のアクティビティ間で資源を同時利用できるよう，資源制約を緩和した問題を解くというものがある．この緩和問題をRと呼ぶことにする．

3.1 ラグランジュ緩和による下界値計算

本節では，ラグランジュ緩和を用いた下界値の計算法を示す．制約(2)および(3)をラグランジュ緩和すると，ラグランジュ乗数の更新において振動を起こす原因になりやすい[2]．ラグランジュ乗数を更新する際の振動を防止するため，待機制約，猶予制約を次の不当式で置き換える．

$$\rho_{ijt} + \sigma_{jt} \leq 1, j \in J, i \in I_j, t \in T \tag{7}$$

$$\rho \in \{0,1\} \tag{8}$$

$$\sigma \in \{0,1\} \tag{9}$$

各記号は次の通りである．

- I_j：アクティビティjの先行アクティビティの集合．
- ρ_{ijt}：0－1変数．(i,j)に関して時刻tが，$0 \leq t \leq c_i + a_{ij}$または$c_i + w_{ij} + p_j < t$を満たせば1，それ以外は0．
- σ_{jt}：0－1変数．時刻tが$st_j \leq t \leq c_j$を満たせば1，それ以外は0．

ここでst_j, c_jはそれぞれ，アクティビティjの処理実行開始時刻および完了時刻である．

$$\sum_{u=0}^{\min(t,p_j-1)} d_{jru} x_{j(t-u)} \leq l_{rt}, \quad j \in J, t \in T, r \in R \tag{10}$$

これらを踏まえ，制約(4)，(7)をそれぞれ，乗数π_{rt}, μ_{ijt}を用いてラグランジュ緩和して定数項を取り除くと次の目的関数を得る．

minimize

$$\begin{aligned} L = & \sum_{t=0}^{t_{max}-1} tx_{(n-1)t} \\ & + \sum_{j=0}^{n-1} \sum_{t=0}^{t_{max}-1} \Big(\sum_{r=0}^{m-1} \pi_{rt} \sum_{u=0}^{min(t,p_j-1)} d_{jru} x_{j(t-u)} \\ & + \sum_{s \in S_j} \mu_{jst} \rho_{jst} + \sum_{i \in I_j} \mu_{ijt} \sigma_{jt} \Big) \end{aligned} \tag{11}$$

この緩和問題をLRと呼ぶことにする．目的関数式(11)はアクティビティレベルの子問題に分解可能であるので，最小化は容易である．LRを解いた後，ラグランジュ乗数を更新し，再度LRを解く．この処理を繰り返し，得られた目的関数値のうち，最大のものを下界値として出力する．

3.2 待機制約の追加による下界値計算

本節では，Rの解における制約(4)の違反に対し，待機制約を一時的に追加して再度Rを解くことにより，下界値を上昇させる．Rの解において，2つのアクティビティj,kがオーバーラップし，j,k間で制約(4)を違反する場合，これを解消するために，問題をR_{jk}とR_{kj}に分割する．R_{jk}にjがkに先行する待機制約を，R_{kj}にはその逆の待機制約を追加する．待機時間は，制約(4)を満たすことができる処理開始時刻が存在するような，最小の値にする．ただしRの解でのjとkのオーバーラップにおけるタイムラグより大きな値を用いる．R_{kj}に対するタイムラグも同様に求める．R_{jk}とR_{kj}の目的関数値のうち，小さい方の値を下界値とする．

4 数値実験

本章では，これまでに述べた手法の数値実験の結果を示す．最適値との比較により，得られる下界値の精度を検証する．対象は，webページPSPLIB[1]にて公開されている，$n = 12, 22, 32, 42$のRCPSPインスタンスをそれぞれ10個ずつ，ランダムに拡張したインスタンスである．それぞれ$t_{max} = 200, m = 4$である．実行環境は次の通りである．

- OS : Windows XP SP2
- 言語 : C++
- CPU : Pentium4 2.4GHz
- メモリ : 768MB

“R”，“LR”および“分割”はそれぞれ，R，LR，待機制約の追加による手法から得た結果を示す．“LR”での繰返し回数の上限は100回である．“LR”，“分割”それぞれにおいて，下界値の暫定値が最適値と一致した時点で，アルゴリズムは終了する．表1に，$n = 12, 22, 32, 42$それぞれのインスタンスにおける誤差の平均を示す．

表2に，$n = 12, 22, 32, 42$それぞれのインスタンスでの平均実行時間を示す．これらの結果から，下界値の精度と計算時間の両面で良好な結果となったのは“分割手法”であることが分かった．しかし，

[1] http://129.187.106.231/psplib/

表1: 誤差の平均 [%]

n	12	22	32	42
R	12.94	19.13	15.31	15.30
LR	10.91	17.80	13.80	14.16
分割	10.14	13.56	12.81	11.88

表2: 平均実行時間 [sec]

n	12	22	32	42
R	0.70	1.60	5.54	10.54
LR	1.20	3.42	11.35	18.50
分割	0.75	1.71	5.59	10.97

アクティビティ数の増加にともない，Rから下界値を改善したインスタンス数は少なくなっている．これは，一時的に待機制約を課されるアクティビティが1組のみであるため，nが大きくなると追加された制約が及ぼす効果が小さくなることが原因だと思われる．どのような待機制約を追加すれば下界値が上昇しやすいかという点についても今後調査が必要である．これを今後の課題とする．

5 おわりに

本稿において，我々はRCPSP/τ+モデルを定式化し，ラグランジュ緩和法および子問題への分割に基づく下界値の計算方法を提案した．$n = 12$から42のインスタンスを用いて，最適値との比較により，得られた下界値の精度を評価した．その結果，“分割”手法が優れた結果を残すことが分かった．最適値との差の平均は10%強という結果を得た．

参考文献

1) S. Hartmann. *Project Scheduling under Limited Resources Models, Methods, and Applications.* Springer-Verlag Berlin, Heidelberg, 1999.

2) Debra J. Hoitomt. A practical approach to job-shop scheduling problems. *IEEE Trans. Robotics and Automation*, Vol. 9, pp. 1–13, 1993.