日本オペレーションズ・リサーチ学会
2023 年春季研究発表会

RPA 手順の最適化問題—整数計画法を用いて

南山大学　吉田水紀　YOSHIDA Mizuki
01204223　南山大学　*鈴木敦夫　SUZUKI Atsuo

1. はじめに

RPA（Robotic Process Automation）は，コンピュータ上で繰り返し行われる定型業務を一度だけ記録し，繰り返し作業自体はロボットがその記録に基づいて行うことで，業務を自動化するという技術である．近年 DX 推進に伴い、多くの企業が定型業務を RPA によって自動化している。しかし現状では，自動化する作業手順は人手で行う順序のまま，すなわち逐次的であり、作業手順の順序の最適化は行われていない．このような作業手順のままで RPA を作成しても業務は自動化されるものの，それらの作業手順が最適化されるとは限らない．例えば，RPA を作成する作業手順を最適化すれば 1 日で済む RPA の所要時間が，実際には 2，3 日になっている可能性もある．そこで，ここでは，RPA を作成する前段階に RPA を作成する作業手順の最適化を行うことで，より効率的な RPA を作成できる可能性を示す．

Seguin らは，[1] で銀行業務を例として，RPA のロボット数の最適化問題，そのロボット数を所与としたロボットへの作業の割当問題を，それぞれ整数計画問題として定式化して解法を提案している．ここではそれとは異なり，手順の最適化とロボットへの割当，さらに RPA がアクセスする資源（データベースなど）の割当の最適化を１つの問題として定式化する．この問題をここでは RPA 最適化問題と呼ぶ．RPA 最適化問題を解くことによって RPA の効率を向上させることが期待できる．

2. RPA 最適化問題

RPA では，ロボットが処理できる作業は，同時に 1 作業のみである．つまり，あるロボットが一つの作業を行っている間は，そのロボットは別の作業を行うことはできない．また，ＲＰＡが外部のデータベースなどの資源を利用する場合は，その資源にアクセスできるのは，一つのロボットのみである．また，各作業には，複数の先行作業があり，それらの先行作業が終わらないとその作業を開始することはできない．

以上のことから，作業間の関係をＰＥＲＴ／ＣＰＭの手法を用いて整理し，ロボットへの作業の割当と資源へのアクセスの条件を満たす作業手順を求める．図 1 は作業の関係をあらわすＰＥＲＴのアローダイアグラムの一例である．ただし，資源へのアクセスは省略している．

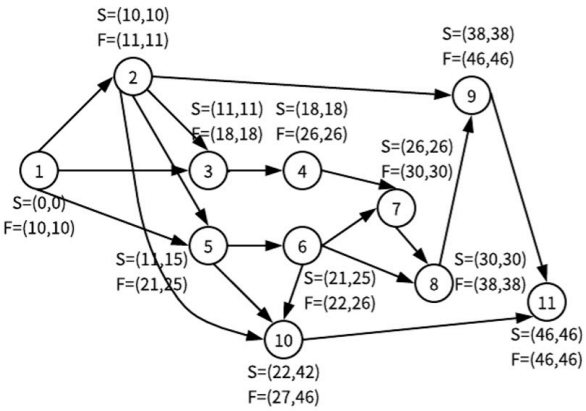

図 1: アローダイアグラムの一例
S，F はそれぞれ開始・終了時刻，カッコ内はそれぞれ最早・最遅時刻をあらわす

問題の定式化は [2] を参考にした．

3. 問題の定式化

定式化には以下の記号を用いた．ただし，作業１と作業 $m+1$ は開始と終了のダミー作業である．

定数

I：作業の集合 $I = \{1, \cdots, m+1\}$
d_i：作業 i の所要時間
P_i：作業 i の直前の作業の集合
R：リソースの集合 $(j \in R)$
n ：ロボットの数

$$r_{ij} = \begin{cases} 1: & \text{作業 } i \text{ がリソース } j \text{ を利用するとき} \\ 0: & \text{その他} \end{cases}$$

変数

x_i ：作業 i の開始時刻

$$s^1_{ii'} = \begin{cases} 1: & \text{作業 } i \text{ の開始時刻が,} \\ & \text{作業 } i' \text{ の終了時刻より早いとき} \\ 0: & \text{その他} \end{cases}$$

$$s^2_{ii'} = \begin{cases} 1: & \text{作業 } i' \text{ の開始時刻が,} \\ & \text{作業 } i \text{ の終了時刻より早いとき} \\ 0: & \text{その他} \end{cases}$$

$$t^1_{ii'} = \begin{cases} 1: & \text{作業 } i' \text{ の終了時刻が,} \\ & \text{作業 } i \text{ の開始時刻より早いとき} \\ 0: & \text{その他} \end{cases}$$

$$t^2_{ii'} = \begin{cases} 1: & \text{作業 } i \text{ の終了時刻が,} \\ & \text{作業 } i' \text{ の開始時刻より早いとき} \\ 0: & \text{その他} \end{cases}$$

$$u_{ii'} = \begin{cases} 1: & \text{作業 } i \text{ と作業 } i' \text{ の作業時間が} \\ & \text{重なっているとき} \\ 0: & \text{その他} \end{cases}$$

目的関数

$$\min. \quad x_{m+1} \tag{1}$$

制約条件

$$x_i \geq \max_{i \in P_i}\{x_{i'} + d_{i'}\}, \ (i \in I \backslash \{1\}) \tag{2}$$

$$x_i - x_{i'} - d_{i'} + M s^1_{ii'} > 0, \ (i, i' \in I, i \neq i') \tag{3}$$

$$x_{i'} - x_i - d_i + M s^2_{ii'} > 0, \ (i, i' \in I, i \neq i') \tag{4}$$

$$x_{i'} + d_{i'} - x_i + M t^1_{ii'} \geq 0, \ (i, i' \in I, i \neq i') \tag{5}$$

$$x_i + d_i - x_{i'} + M t^2_{ii'} \geq 0, \ (i, i' \in I, i \neq i') \tag{6}$$

$$s^1_{ii'} + t^1_{ii'} = 1, \ (i, i' \in I, i \neq i') \tag{7}$$

$$s^2_{ii'} + t^2_{ii'} = 1, \ (i, i' \in I, i \neq i') \tag{8}$$

$$u_{ii'} = (s^1_{ii'} + s^2_{ii'}) - 1, \ (i, i' \in I, i \neq i') \tag{9}$$

$$\sum_{i' \in I, i \neq i', r_{ij} = r_{i'j}} u_{ii'} \leq 0, \ (i \in I) \tag{10}$$

$$\sum_{i' \in I, i \neq i'} u_{ii'} \leq n - 1, \ (i \in I) \tag{11}$$

$$x_i \geq 0, \ (i \in I) \tag{12}$$

$$s^1_{ii'}, s^2_{ii'}, t^1_{ii'}, t^2_{ii'}, u_{ii'} \in \{0, 1\}, \ (i, i' \in I) \tag{13}$$

ここでは M は大きな数である．

(1) は終了時刻最小化の目的関数，(2) は 作業 i の開始時刻が作業 i の先行作業の終了時刻の後という制約，(3)，(4)，(5)，(6)，(7)，(8)，(9) は作業 i と i' の実行時刻が重複しているかどうかを判別するための制約式，(10) は作業が重なっているときに同じ資源にアクセスすることができない制約，(11) はロボット数の制約，(12) は x_i の非負制約，(13) は $s^1_{ii'}, s^2_{ii'}, t^1_{ii'}, t^2_{ii'}, u_{ii'}$ のバイナリ制約である．

4. 小規模の RPA 最適化問題についての計算機実験

仮想的な例を乱数を用いて多数作成し，先行作業の数，ロボット数などの条件により RPA の所要時間がどのように変化するかを，RPA 最適化問題を解いた場合と従来通り逐次的の場合を比較して，RPA の総作業時間の短縮率のどの程度であるかを数値的に比較・検証する．その結果，ロボット数 3，リソース数 3 の場合，作業数 20 の時は約 35%，作業数 100 の時は約 50%の作業時間を短縮できた．これは作業数の増加とともに並行して行える作業も増えたためだと考えられる。リソース数とロボット数が増えたときは短縮率も増加した．先行作業数が増えた場合は大幅な短縮は見られなかったが，計算時間内に最適な解を出せていない可能性も考えられるため，今後検証する必要がある．詳細については当日報告する．実験に使用した計算機の仕様は，CPU:Apple M1, RAM:16GB, OS: Big Sur，最適化計算に使用したソフトウェアは，Gurobi9.5.2 である．ロボット数 3，リソース数 3，作業数 100 の場合の RPA 最適化問題を解くための計算時間は，約 1000 秒であった．計算結果から RPA 最適化問題を解いてから RPA を作成することで総作業時間が短縮すると考えられる．

参考文献

[1] Sara Seguin and Imene Benkalai：Robotic Process Automation (RPA) Using an Integer Linear Programming Formulation, *Cybernetics and Systems*, Vol. 51, No. 4, pp. 357-369, 2020.

[2] 関根智明：PERT・CPM, 日科技連出版社，東京，1965.